1. Crowd Documentation . . . 5
1.1 Crowd 101 . . . 5
1.2 Crowd User Guide . . . 9
1.2.1 Introduction to Crowd . . . 11
1.2.2 Logging in to Crowd . . . 12
1.2.3 Logging out of Crowd . . . 13
1.2.4 Changing or Resetting your Password . . . 14
1.2.4.1 Changing your Password . . . 14
1.2.4.2 Resetting your Password . . . 15
1.2.5 Updating your User Profile . . . 16
1.2.6 Viewing your Group Membership . . . 17
1.2.7 Viewing your Role Membership . . . 17
1.2.8 Viewing your Applications . . . 18
1.2.9 Crowd User's Glossary . . . 19
1.2.9.1 Alias (Glossary Entry) . . . 19
1.2.9.2 Authorisation to Use Crowd (Glossary Entry) . . . 19
1.2.9.3 Crowd Administrator (Glossary Entry) . . . 20
1.2.9.4 Crowd-Connected Application (Glossary Entry) . . . 20
1.2.9.5 Directory (Glossary Entry) . . . 20
1.2.9.6 Self-Service Console (Glossary Entry) . . . 20
1.2.9.7 Single Sign-On (Glossary Entry) . . . 20
1.3 Crowd Administration Guide . . . 20
1.3.1 Getting Started . . . 22
1.3.1.1 Concepts . . . 22
1.3.1.2 Supported Applications and Directories . . . 23
1.3.1.3 About the Crowd Administration Console . . . 24
1.3.2 Managing Directories . . . 25
1.3.2.1 Using the Directory Browser . . . 25
1.3.2.2 Adding a Directory . . . 26
1.3.2.2.1 Configuring an Internal Directory . . . 28
1.3.2.2.2 Configuring an LDAP Directory Connector . . . 29
1.3.2.2.3 Configuring a Custom Directory Connector . . . 65
1.3.2.2.4 Configuring a Delegated Authentication Directory . . . 66
1.3.2.3 Configuring Caching for an LDAP Directory . . . 71
1.3.2.4 Configuring Relaxed DN Standardisation . . . 76
1.3.2.5 Specifying Directory Permissions . . . 77
1.3.2.6 Importing Users and Groups into a Directory . . . 78
1.3.2.6.1 Importing Users from Atlassian Confluence . . . 80
1.3.2.6.2 Importing Users from Atlassian JIRA . . . 81
1.3.2.6.3 Importing Users from Atlassian Bamboo . . . 82
1.3.2.6.4 Importing Users from Jive Forums . . . 84
1.3.2.6.5 Importing Users from CSV Files . . . 85
1.3.2.6.6 Importing Users from One Crowd Directory into Another . . . 91
1.3.3 Managing Applications . . . 93
1.3.3.1 Using the Application Browser . . . 93

1.3.3.2 Adding an Application . . . 95
1.3.3.2.1 Integrating Crowd with Atlassian Bamboo . . . 98
1.3.3.2.2 Integrating Crowd with Atlassian Confluence . . . 102
1.3.3.2.3 Integrating Crowd with Atlassian CrowdID . . . 107
1.3.3.2.4 Integrating Crowd with Atlassian Crucible . . . 109
1.3.3.2.5 Integrating Crowd with Atlassian FishEye . . . 110
1.3.3.2.6 Integrating Crowd with Atlassian JIRA . . . 115
1.3.3.2.7 Integrating Crowd with Acegi Security . . . 121
1.3.3.2.8 Integrating Crowd with Apache . . . 128
1.3.3.2.9 Integrating Crowd with Jive Forums . . . 132
1.3.3.2.10 Integrating Crowd with Spring Security . . . 137
1.3.3.2.11 Integrating Crowd with Subversion . . . 145
1.3.3.2.12 Integrating Crowd with a Custom Application . . . 147
1.3.3.3 Configuring the Google Apps Connector . . . 148
1.3.3.4 Mapping a Directory to an Application . . . 153
1.3.3.4.1 Specifying the Directory Order for an Application . . . 155
1.3.3.4.2 Specifying an Application's Directory Permissions . . . 157
1.3.3.5 Specifying which Groups can access an Application . . . 160
1.3.3.6 Viewing Users in Directories Mapped to an Application . . . 161
1.3.3.7 Specifying an Application's Address or Hostname . . . 162
1.3.3.8 Testing a User's Login to an Application . . . 164
1.3.3.9 Enforcing Lower-Case Usernames, Groups and Roles for an Application . . . 165
1.3.3.10 Managing an Application's Session . . . 166
1.3.3.11 Deleting or Deactivating an Application . . . 168
1.3.3.12 Configuring Caching for an Application . . . 169
1.3.3.13 Overview of SSO . . . 171
1.3.3.14 Configuring Options for an Application . . . 174
1.3.4 Managing Users, Groups and Roles . . . 174
1.3.4.1 Using the User Browser . . . 175
1.3.4.2 Adding a User . . . 176
1.3.4.3 Deleting or Deactivating a User . . . 177
1.3.4.4 Managing a User's Session . . . 179
1.3.4.5 Editing a User's Details and Password . . . 180
1.3.4.6 Specifying a User's Attributes . . . 181
1.3.4.7 Specifying a User's Aliases . . . 182
1.3.4.8 Granting Crowd Administration Rights to a User . . . 185
1.3.4.9 Granting Crowd User Rights to a User . . . 186
1.3.4.10 Managing Groups and Roles . . . 187
1.3.4.10.1 Adding a Group or Role . . . 189
1.3.4.10.2 Deleting or Deactivating a Group . . . 190
1.3.4.11 Managing Group Members . . . 191
1.3.4.11.1 Automatically Assigning New Users to Groups . . . 192
1.3.4.11.2 Adding Users to a Group . . . 194
1.3.4.11.3 Removing Users from a Group . . . 197
1.3.4.11.4 Nested Groups in Crowd . . . 200
1.3.4.11.5 Adding a Sub-Group . . . 203
1.3.4.11.6 Removing a Sub-Group . . . 204
1.3.4.12 Case Sensitivity of Usernames, Groups and Roles . . . 206
1.3.4.13 Editing a User's Group and Role Membership . . . 207
1.3.5 System Administration . . . 208
1.3.5.1 Configuring Server Settings . . . 208
1.3.5.1.1 Deployment Title . . . 208
1.3.5.1.2 Domain . . . 209
1.3.5.1.3 Token Seed . . . 211
1.3.5.1.4 Session Configuration . . . 212
1.3.5.1.5 Authorisation Caching . . . 213
1.3.5.1.6 Compression of Server Output . . . 214
1.3.5.1.7 Licensing . . . 215
1.3.5.1.8 SSO Cookie . . . 217
1.3.5.2 Configuring your Mail Server . . . 218
1.3.5.3 Creating an Email Notification Template . . . 221
1.3.5.4 Configuring Trusted Proxy Servers . . . 222
1.3.5.5 Viewing Crowd's System Information . . . 223
1.3.5.6 Backing Up and Restoring Data . . . 225
1.3.5.7 Logging and Profiling . . . 226
1.3.5.7.1 Performance Profiling . . . 230
1.3.5.8 Overview of Caching . . . 230
1.3.6 Crowd Security Advisories and Fixes . . . 231
1.3.6.1 Crowd Security Advisory 2008-10-14 - Parameter Injection Vulnerability . . . 232
1.4 Crowd Installation & Upgrade Guide . . . 232
1.4.1 Crowd Release Notes . . . 233
1.4.1.1 Crowd Release Summary . . . 233
1.4.1.2 Crowd 2.0.3 Release Notes . . . 234
1.4.1.3 Crowd 2.0.2 Release Notes . . . 235
1.4.1.4 Crowd 2.0.1 Release Notes . . . 236
1.4.1.5 Crowd 2.0 Release Notes . . . 237
1.4.1.6 Crowd 2.0 Beta Release Notes . . . 247
1.4.1.7 Crowd 1.6.1 Release Notes . . . 249
1.4.1.8 Crowd 1.6 Release Notes . . . 250
1.4.1.9 Crowd 1.5.2 Release Notes . . . 255

1.4.1.10 Crowd 1.5.1 Release Notes . . . 255
1.4.1.11 Crowd 1.5 Release Notes . . . 256
1.4.1.12 Crowd 1.4.7 Release Notes . . . 262
1.4.1.13 Crowd 1.4.4 Release Notes . . . 262
1.4.1.14 Crowd 1.4.3 Release Notes . . . 263
1.4.1.15 Crowd 1.4.2 Release Notes . . . 263
1.4.1.16 Crowd 1.4.1 Release Notes . . . 264
1.4.1.17 Crowd 1.4 Release Notes . . . 264
1.4.1.18 Crowd 1.3.3 Release Notes . . . 268
1.4.1.19 Crowd 1.3.2 Release Notes . . . 269
1.4.1.20 Crowd 1.3.1 Release Notes . . . 269
1.4.1.21 Crowd 1.3 Release Notes . . . 270
1.4.1.21.1 Client API Changes . . . 277
1.4.1.21.2 Known Issues in Crowd 1.3 . . . 279
1.4.1.22 Crowd 1.3 Beta Release Notes . . . 280
1.4.1.23 Crowd 1.2.4 Release Notes . . . 284
1.4.1.24 Crowd 1.2.2 Release Notes . . . 284
1.4.1.25 Crowd 1.2.1 Release Notes . . . 285
1.4.1.26 Crowd 1.2 Release Notes . . . 286
1.4.1.27 Crowd 1.1.2 Release Notes . . . 292
1.4.1.28 Crowd 1.1.1 Release Notes . . . 293
1.4.1.29 Crowd 1.1.0 Release Notes . . . 295
1.4.1.30 Crowd 1.0.7 Release Notes . . . 300
1.4.1.31 Crowd 1.0.6 Release Notes . . . 300
1.4.1.32 Crowd 1.0.5 Release Notes . . . 301
1.4.1.33 Crowd 1.0.4 Release Notes . . . 302
1.4.1.34 Crowd 1.0.3 Release Notes . . . 302
1.4.1.35 Crowd 1.0.2 Release Notes . . . 303
1.4.1.36 Crowd 1.0.1 Release Notes . . . 304
1.4.1.37 Crowd 1.0.0 Release Notes . . . 304
1.4.1.38 Crowd 0.4.5 Beta Release Notes . . . 305
1.4.1.39 Crowd 0.4.4 Beta Release Notes . . . 305
1.4.1.40 Crowd 0.4.3 Beta Release Notes . . . 305
1.4.1.41 Crowd 0.4.2 Beta Release Notes . . . 305
1.4.1.42 Crowd 0.4.1 Beta Release Notes . . . 306
1.4.1.43 Crowd 0.4 Beta Release Notes . . . 306
1.4.1.44 Crowd 0.3.3 Beta Release Notes . . . 306
1.4.1.45 Crowd 0.3.2 Beta Release Notes . . . 307
1.4.1.46 Crowd 0.3 Beta Release Notes . . . 307
1.4.1.47 Crowd 0.2 Beta Release Notes . . . 307
1.4.2 Installing Crowd . . . 308
1.4.2.1 Supported Platforms . . . 308
1.4.2.1.1 Setting JAVA_HOME . . . 310
1.4.2.2 Installing Crowd and CrowdID . . . 311
1.4.2.2.1 Connecting Crowd to a Database . . . 312
1.4.2.2.2 Connecting CrowdID to a Database . . . 316
1.4.2.2.3 Installing Crowd and CrowdID WAR Distribution . . . 324
1.4.2.2.4 Specifying your Crowd Home Directory . . . 332
1.4.2.3 Running the Setup Wizard . . . 333
1.4.2.3.1 Troubleshooting your Configuration on Setup . . . 340
1.4.2.4 Configuring Crowd . . . 341
1.4.2.4.1 Important Directories and Files . . . 341
1.4.2.4.2 Changing the Port that Crowd uses . . . 346
1.4.2.4.3 Configuring Crowd to Work with SSL . . . 347
1.4.2.5 Installing Crowd as a Windows Service . . . 350
1.4.2.5.1 Specifying Startup Order of Windows Services . . . 351
1.4.2.5.2 Changing the User for the Crowd Windows Service . . . 351
1.4.2.5.3 Removing the Crowd Windows Service . . . 352
1.4.2.5.4 Troubleshooting Crowd as a Windows Service . . . 352
1.4.3 Upgrading Crowd . . . 353
1.4.3.1 Upgrade Notes . . . 355
1.4.3.1.1 Crowd 1.0 Upgrade Notes . . . 355
1.4.3.1.2 Crowd 1.1 Upgrade Notes . . . 355
1.4.3.1.3 Crowd 1.2 Upgrade Notes . . . 356
1.4.3.1.4 Crowd 1.3 Beta Upgrade Notes . . . 356
1.4.3.1.5 Crowd 1.3 Upgrade Notes . . . 357
1.4.3.1.6 Crowd 1.4 Upgrade Notes . . . 358
1.4.3.1.7 Crowd 1.5 Upgrade Notes . . . 358
1.4.3.1.8 Crowd 1.6 Upgrade Notes . . . 359
1.4.3.1.9 Crowd 2.0 Upgrade Notes . . . 359
1.4.4 Migrating Crowd between Servers . . . 360
1.5 CrowdID User Guide . . . 361
1.5.1 1. Getting started with CrowdID . . . 361
1.5.1.1 1.1 What is OpenID? . . . 362
1.5.1.2 1.2 What is CrowdID? . . . 362
1.5.1.3 1.3 What is an OpenID URL or identifier? . . . 362
1.5.1.4 1.4 Viewing the CrowdID page . . . 363
1.5.2 2. Logging in to a website using OpenID . . . 364
1.5.2.1 2.1 Does the website support OpenID? . . . 364
1.5.2.2 2.2 Entering your OpenID URL . . . 365

1.5.2.3 2.3 Logging in to CrowdID . . . 365
1.5.2.4 2.4 Allowing or denying a login . . . 366
1.5.2.5 2.5 Providing additional profile information to a website . . . 368
1.5.3 3. Viewing your always-approved websites . . . 368
1.5.4 4. Viewing your login history . . . 369
1.5.5 5. Updating your profile . . . 370
1.5.6 6. Using more than one profile . . . 371
1.5.6.1 6.1 Adding a profile . . . 372
1.5.6.2 6.2 Choosing a profile for a website . . . 373
1.5.6.3 6.3 Setting a default profile . . . 373
1.5.6.4 6.4 Deleting a profile . . . 374
1.5.7 7. Changing or resetting your password . . . 375
1.5.7.1 7.1 Changing your password . . . 375
1.5.7.2 7.2 Resetting your password . . . 376
1.6 CrowdID Administration Guide . . . 377
1.6.1 1. About CrowdID . . . 377
1.6.1.1 1.1 How CrowdID works with Crowd . . . 378
1.6.1.1.1 1.1.1 Determining the name of the CrowdID application . . . 378
1.6.1.1.2 1.1.2 Locating the Crowd Server that CrowdID is using . . . 379
1.6.1.2 1.1 How OpenID sites interact with CrowdID . . . 380
1.6.2 2. Allowing users to access CrowdID . . . 382
1.6.2.1 2.1 Granting CrowdID access rights to a user . . . 382
1.6.2.2 2.2 Granting CrowdID Administration Rights to a User . . . 383
1.6.3 3. Specifying the sites to which users can login . . . 383
1.6.3.1 3.1 Allowing all hosts . . . 384
1.6.3.2 3.2 Allowing all except specified hosts ('Blacklist') . . . 384
1.6.3.3 3.3 Allowing specified hosts only ('Whitelist') . . . 385
1.6.4 4. Configuring CrowdID system settings . . . 386
1.6.4.1 4.1 Specifying the CrowdID URL . . . 386
1.6.4.2 4.2 Enabling localhost authentication . . . 387
1.6.4.3 4.3 Enabling immediate authentication requests . . . 388
1.6.4.4 4.4 Enabling communication with stateless clients . . . 390
1.7 Crowd FAQ . . . 391
1.7.1 Deployment FAQ . . . 392
1.7.1.1 Finding the atlassian-crowd.log File . . . 392
1.7.1.2 Finding your Crowd Home Directory . . . 393
1.7.1.3 Recovering your Console application password . . . 393
1.7.1.4 Resetting the Domain Cookie Value . . . 395
1.7.1.5 Restarting the Setup Wizard from Scratch . . . 395
1.7.1.6 Self Signed Certificate . . . 395
1.7.1.7 Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX . . . 395
1.7.1.8 Using Crowd in a Cluster is Not Supported . . . 397
1.7.2 Guides, Hints and Tips . . . 397
1.7.2.1 Principals and Users . . . 397
1.7.2.2 Using Apache Directory Studio for Crowd LDAP Configuration . . . 397
1.7.2.2.1 Creating a Connection to your LDAP Directory . . . 397
1.7.2.2.2 Getting an LDIF Export of a User or Group . . . 402
1.7.2.2.3 Restricting LDAP Scope for User and Group Search . . . 402
1.7.3 Integration FAQ . . . 405
1.7.3.1 All Integrations . . . 406
1.7.3.1.1 If I delete a user from Crowd, how will this affect integrated applications? . . . 406
1.7.3.1.2 Passing the crowd.properties File as an Environment Variable . . . 406
1.7.3.2 Atlassian Product Integration . . . 406
1.7.3.2.1 Application Caching . . . 406
1.7.3.2.2 JIRA integration . . . 407
1.7.3.2.3 Public Signup Setup . . . 407
1.7.3.3 IBM Lotus Domino Integration . . . 407
1.7.3.4 IBM Websphere Integration . . . 407
1.7.4 Support Policies . . . 408
1.7.4.1 Bug Fixing Policy . . . 408
1.7.4.2 How to Report a Security Issue . . . 409
1.7.4.3 New Features Policy . . . 409
1.7.4.4 Patch Policy . . . 409
1.7.4.5 Security Advisory Publishing Policy . . . 410
1.7.4.6 Security Patch Policy . . . 410
1.7.4.7 Severity Levels for Security Issues . . . 410
1.7.5 Troubleshooting . . . 411
1.7.5.1 Finding Known Issues . . . 411
1.7.5.2 Characters in User or Group DN's that will cause problems when using Crowd . . . 411
1.7.5.3 Problems when Importing Users into MySQL . . . 412
1.7.5.4 Troubleshooting LDAP Error Codes . . . 412
1.7.5.4.1 Active Directory LDAP Errors . . . 412
1.7.5.5 Troubleshooting SSL certificates and Crowd . . . 412
1.7.5.6 How to Optimise Crowd Client Caching . . . 413
1.7.5.7 Troubleshooting Crowd Performance . . . 413
1.7.5.8 Troubleshooting SSO with Crowd . . . 414
1.7.5.8.1 Debugging SSO in environments with Proxy Servers . . . 415
1.7.5.9 Troubleshooting CrowdID . . . 416
1.8 Crowd Resources . . . 417
1.8.1 Tips of the Trade . . . 417

Crowd Documentation

Crowd 2.0.x

User's Guide

The Crowd User Guide is for project managers, developers, testers – anyone who uses Crowd. New to Crowd? Start with the introduction to Crowd. Try logging in, then explore your user profile, see the groups you belong to and the applications you can access. You can also use Crowd to change your password across all your applications.

Administrator's Guide

The Crowd Administration Guide is for people with Crowd administration rights. It will help you configure your email server and set up applications, directories, users and groups. Learn about integrating Crowd with JIRA, Confluence and other applications. Administrative tasks such as backup are also covered. You may also find the Knowledge Base, FAQ and Crowd forum useful.

Installation Guide

The Crowd Installation Guide is for people who are installing Crowd for the first time. Check the supported platforms, then download and install Crowd. Where to next? Crowd 101 will help you get started. If you are using other Atlassian products, take a look at the Integration Guide.

Upgrade Guide

The Crowd Upgrade Guide is for people who are upgrading their instance of Crowd to a newer version. Start by reading the latest release notes and version-specific upgrade notes for the version to which you are upgrading, then download Crowd and follow the main Upgrade Guide.

Developer Resources

These resources are for software developers who want to create their own plugins or extensions for Crowd. Take a look at the Development Hub and the API documentation. You may also find the Crowd Developers Forum useful. (Click here to subscribe.)

CrowdID User's Guide

Using CrowdID? Read the CrowdID User Guide to learn about managing your OpenID logins.

CrowdID Administrator's Guide

The CrowdID Administration Guide shows you how to allow users to access CrowdID, black list or white list external sites and configure your CrowdID server.

Crowd 101

Crowd 101

Thank you for choosing Crowd. To help you get up and running quickly, we have compiled some quick-start instructions on configuring and using Crowd with your JIRA and Confluence applications.

This quick-start guide assumes that you have installed and set up JIRA and/or Confluence and now wish to set up Crowd for user management in one or both of them.

- If you want to use JIRA or Confluence but have not yet installed them, please follow the instructions in JIRA 101 and/or Confluence 101 before configuring Crowd.
- If you want to use Crowd with other applications but not JIRA or Confluence, please follow the detailed Crowd installation and setup guide rather than this 'Crowd 101' guide.

Getting Started

1. Installing Crowd

First things first. If you have not already got Crowd up and running, carry out the following steps:

For Windows: (click to expand)

1. Go to the Atlassian download centre.
2. Download the 'Standalone (ZIP archive)' file.
3. Unzip the zip archive into a directory of your choice, avoiding spaces in the directory name.
4. Tell Crowd where to find its Crowd Home directory, by editing the `crowd-init.properties` file as described in the installation guide.
5. Set up your database as described in the database configuration guide.
   This quick-start page assumes that you have an existing JIRA or Confluence application. So we recommend that you connect Crowd to a production-ready database and not HSQLDB. But if you are evaluating Crowd, it is fine to use HSQLDB and then move to a different database later. In that case, you do not need to do anything in this step, because Crowd contains everything you need.
6. Start your Crowd server by going to the directory where you unzipped Crowd and running `start_crowd.bat`.
7. To access Crowd, go to your web browser and type this address: http://localhost:8095/crowd.
8. Follow the Setup Wizard. This will guide you through the process of setting up your Crowd server and creating an admin user.

For more help on the technical procedures in this section, please refer to the Crowd installation guide.

If you need assistance, please create a support ticket.

For Mac: (click to expand)

1. Go to the Atlassian download centre.
2. Click the 'Mac OS X' tab and download the 'Standalone (TAR.GZ archive)' file.
3. Unzip the archive into a directory of your choice, avoiding spaces in the directory name.
4. Tell Crowd where to find its Crowd Home directory, by editing the `crowd-init.properties` file as described in the installation guide.
5. Set up your database as described in the database configuration guide.
   This quick-start page assumes that you have an existing JIRA or Confluence application. So we recommend that you connect Crowd to a production-ready database and not HSQLDB. But if you are evaluating Crowd, it is fine to use HSQLDB and then move to a different database later. In that case, you do not need to do anything in this step, because Crowd contains everything you need.
6. Start your Crowd server by going to the directory where you unzipped Crowd and double-clicking `start_crowd.sh`.
7. To access Crowd, go to your web browser and type this address: http://localhost:8095/crowd.
8. Follow the Setup Wizard. This will guide you through the process of setting up your Crowd server and creating an admin user.

For more help on the technical procedures in this section, please refer to the Crowd installation guide.

If you need assistance, please create a support ticket.

For UNIX or Linux: (click to expand)

1. Go to the Atlassian download centre.
2. Click the 'Linux' tab and download the 'Standalone (TAR.GZ Archive)' file.
3. Unzip the archive into a directory of your choice, avoiding spaces in the directory name.
4. Tell Crowd where to find its Crowd Home directory, by editing the `crowd-init.properties` file as described in the installation guide.
5. Set up your database as described in the database configuration guide.
   This quick-start page assumes that you have an existing JIRA or Confluence application. So we recommend that you connect Crowd to a production-ready database and not HSQLDB. But if you are evaluating Crowd, it is fine to use HSQLDB and then move to a different database later. In that case, you do not need to do anything in this step, because Crowd contains everything you need.
6. Start your Crowd server by going to the directory where you unzipped Crowd and double-clicking `start_crowd.sh`.
7. To access Crowd, go to your web browser and type this address: http://localhost:8095/crowd.
8. Follow the Setup Wizard. This will guide you through the process of setting up your Crowd server and creating an admin user.

For more help on the technical procedures in this section, please refer to the Crowd installation guide.

If you need assistance, please create a support ticket.

2. Adding Users and Groups

Crowd is designed to help you manage users and groups across multiple applications. Your next step is to configure a user directory in Crowd to contain your JIRA and/or Confluence users and groups.

If you are starting out from scratch with a new JIRA and a new Confluence site: (click to expand)

1. Add a Crowd directory — Add a Crowd Internal directory to contain all your JIRA and Confluence users.
2. Add the Confluence groups — Add the 'confluence-users' and 'confluence-administrators' groups to your new directory.
3. Add the JIRA groups — Add the 'jira-users', 'jira-developers' and 'jira-administrators' groups to your new directory.
4. Import your users from a CSV file or add them manually.
5. Add the users to the groups — Use Crowd's bulk user management to add all the users to the 'confluence-users' and 'jira-users' groups. Also add any administrators to the administration groups and add the developers to the 'jira-developers' group. For more details about the permissions applicable to each group, refer to the Confluence and JIRA documentation.

▷ If you have existing JIRA and Confluence sites, each currently managing its own set of users internally: (click to expand)

If your JIRA users are currently managed via JIRA's internal management and your Confluence users are managed separately via Confluence's internal management, you can use Crowd to simplify and centralise your user and group management:

1. Add a Crowd directory — Use the Crowd Administration Console to add a Crowd Internal directory to contain all your JIRA and Confluence users.
2. Import the users and groups from Confluence — Use the Crowd importer to copy your users and groups from Confluence into the new Crowd directory. This process will also copy the group memberships into Crowd.
3. Import the users and groups from JIRA — Use the Crowd importer to copy your users and groups from JIRA into the same Crowd directory as the Confluence users. This process will add any additional users and groups from JIRA and update the existing Confluence users with their JIRA group memberships.
4. Check your users and groups in Crowd — Use Crowd's group browser to check that your users, groups and group memberships are available as expected in Crowd.

▷ If you have existing JIRA and Confluence sites, with all users currently managed internally in JIRA: (click to expand)

If your JIRA and Confluence users are currently all managed via JIRA's internal management, you can use Crowd to simplify and centralise your user and group management:

1. Add a Crowd directory — Use the Crowd Administration Console to add a Crowd Internal directory to contain all your JIRA and Confluence users.
2. Import the users and groups from JIRA — Use the Crowd importer to copy your users and groups from JIRA into the new Crowd directory. This process will also copy the group memberships into Crowd.
3. Check your users and groups in Crowd — Use Crowd's group browser to check that your users, groups and group memberships are available as expected in Crowd.

▷ If you have existing JIRA and Confluence sites, with all users currently managed in an LDAP directory: (click to expand)

If your users are in a corporate LDAP directory, you can choose whether you want to store your groups in LDAP or in Crowd.

- If you want to store your users and groups in LDAP:
    1. Add a Crowd LDAP directory connector — Choose the options for your specific version of LDAP, such as Microsoft Active Directory or Novell eDirectory. Crowd supports a number of LDAP flavours, as listed in the documentation.
    2. Check your users and groups in Crowd — Use Crowd's group browser to check that your users, groups and group memberships are available as expected in Crowd.
- If you want to store your users in LDAP and your groups in Crowd:
    1. Add a Crowd Delegated Authentication directory — Choose the options for your specific version of LDAP, such as Microsoft Active Directory or Novell eDirectory. Crowd supports a number of LDAP flavours, as listed in the documentation.
    2. Add the Confluence groups — Add the 'confluence-users' and 'confluence-administrators' groups to your new Crowd Delegated Authentication directory.
    3. Add the JIRA groups — Add the 'jira-users', 'jira-developers' and 'jira-administrators' groups to your new Crowd Delegated Authentication directory.
    4. Add the users to the groups — Use Crowd's bulk user management to add all the users to the 'confluence-users' and 'jira-users' groups. Also add any administrators to the administration groups and add the developers to the 'jira-developers' group. For more details about the permissions applicable to each group, refer to the Confluence and JIRA documentation.

▷ If none of the above scenarios matches your requirements: (click to expand)

Take the following steps, choosing your directory and other options as indicated in the linked documentation:

1. Add a Crowd directory — Choose the directory type you need to contain all your JIRA and Confluence users.
2. Add your users and groups either via Crowd's importer or manually:
    - Import your users and groups into Crowd.
    - Or do it manually:
        a. Add the users.
        b. Add the Confluence groups — Add the 'confluence-users' and 'confluence-administrators' groups to your new directory.
        c. Add the JIRA groups — Add the 'jira-users', 'jira-developers' and 'jira-administrators' groups to your new directory.
        d. Add the users to the groups — Use Crowd's bulk user management to add all the users to the 'confluence-users' and 'jira-users' groups. Also add any administrators to the administration groups and add the developers to the 'jira-developers' group. For more details about the permissions applicable to each group, refer to the Confluence and JIRA documentation.

If you have Confluence or JIRA, but not both, pick the scenario above that best matches your requirements, then just skip the steps for the application that you do not need.

3. Connecting the Applications

Crowd manages your users' access to your applications and makes single sign-on (SSO) possible. (More about SSO below.) For this to happen, you need to tell Crowd about the applications and to copy some Crowd libraries into the applications' installation folders.

1. Add Confluence — Add the Confluence application to Crowd, following the instructions in the Add Application Wizard.
    - Choose 'Confluence' as the application type.
    - In the 'Directories' step, choose the user directory you added for Confluence.
    - In the 'Authorisation' step, allow all users to authenticate.
2. Configure the Crowd libraries in Confluence — Copy the Crowd client libraries into your Confluence folders and configure the properties files as described on the Confluence integration page.
3. Now add JIRA — Add the JIRA application to Crowd, following the instructions in the Add Application Wizard.
    - Choose 'JIRA' as the application type.
    - In the 'Directories' step, choose the user directory you added for JIRA.
    - In the 'Authorisation' step, allow all users to authenticate.
4. Configure the Crowd libraries in JIRA — Copy the Crowd client libraries into your JIRA folders and configure the properties files as described on the JIRA integration page.

We will call these your 'Crowd-connected applications'.

Mastering the Basics

4. Examining your Crowd Server Setup

Go to the System Information screen in Crowd's Administration Console to find useful information about your Crowd server, such as the location of your Crowd Home directory, information about your database and JVM, and your license server ID.

5. Managing SSO

If you have configured single sign-on (SSO) when setting up your Crowd-connected applications (JIRA and Confluence) in step 3 above, your users will only need to log in or log out once, to Crowd or any Crowd-connected application. When they start another Crowd-connected application, they will be logged in automatically. Similarly, when they log out of Crowd or one of the Crowd-connected applications, they will be logged out of Crowd and the other application(s) at the same time.

- Overview of SSO — An overview of Crowd's SSO capabilities, plus links to detailed information.
- Configuring Trusted Proxy Servers — If you are running applications behind one or more proxy servers, you may find it useful to configure Crowd to trust the proxies' IP addresses.

Managing your Users' Experience of Crowd

Your users will need to access Crowd at **http://<Crowd machine name>:8095/crowd** (not http://localhost:8095/crowd).

6. Testing a User's Login

Why would I do this? (click to expand)

You may want to test a user's login to a specific application if the user has reported problems with logging in, or if you have just set up the first user to access a new application. The test verifies whether a user will be able to log in to a given application, based on the application, directory and group associations in Crowd.

How do I do this? (click to expand)

Go to the application's 'Authentication Test' tab in the Crowd Administration Console, as described in the documentation. The documentation also describes the possible error messages and the steps you can take to resolve any problems.

7. Changing or Resetting a User's Password

Why would I do this? (click to expand)

You may need to change or reset someone's password, if they have forgotten their password or if someone else has come to know the password.

Crowd users can change or reset their own passwords too. See the user documentation. To allow this, you need to grant them Crowd user rights, as described below.

How do I do this? (click to expand)

Go to the 'User Details' screen in the Crowd Administration Console, as described in the documentation.

If you have configured an email server and a notification template, Crowd will send the user an email about their new password.

8. Setting Up User Aliases

▷ Why would I do this? (click to expand)

Aliases are useful if the same person has different usernames in JIRA and Confluence. You can define the user just once in Crowd, and allocate one or more aliases for the different applications that the user can access.

▷ How do I do this? (click to expand)

The documentation has the full details. In summary:

1. Make sure that aliasing is enabled for JIRA and Confluence, on the application's 'Options' screen.
2. Add the appropriate alias for each user, on the user's 'Applications' screen.

9. Granting Crowd User Rights to Someone

▷ Why would I do this? (click to expand)

You can give your users access to Crowd's Self-Service Console, where they can edit their own profile, change their password and see the applications they are allowed to access. They can read the Crowd User Guide for guidance.

▷ How do I do this? (click to expand)

Make sure that the person's username is in a user directory where all users are authorised to use Crowd. Please refer to the documentation for details.

10. Granting Crowd Administrator Rights to Someone

▷ Why would I do this? (click to expand)

When you first set up Crowd, you will define a single Crowd administrator. It is advisable to give other people administration rights too, so that you do not run into problems when the single administrator is unavailable.

▷ How do I do this? (click to expand)

Make sure that the person is a member of the 'crowd-administrators' group. Please refer to the documentation.

Important Next Steps

11. Setting Up your Applications' Host Names

When you set up your applications in step 3 above, you will have specified an IP address for each application. If JIRA, Confluence or any Crowd-connected application resides on a server that passes Crowd a host name instead of an IP address, you will need to tell Crowd the host name. Please refer to the documentation.

12. Connecting to an External Database

If you decided to use the default HSQLDB database when you set up Crowd, you need to switch to a production-ready database before using Crowd as a production system. HSQLDB is provided for evaluation purposes only. Please refer to the documentation.

13. Backing Up your Crowd Data

To back up your Crowd data and establish processes for regular backups, please refer to the documentation.

Thank you for choosing Crowd.

We are always happy to help. Feel free to email or call us with any questions you may have.

Crowd User Guide

About Crowd

Atlassian's Crowd is a software application installed by the system administrator. The administrator will also connect one or more of your organisation's applications to Crowd. When you log in to a Crowd-connected application, Crowd will verify your password and login permissions.

Using Crowd for single sign-on (SSO), each person needs only one username and password to access all web applications. You can host your own OpenID provider to include external applications.

- You only need to log in once, to Crowd or a Crowd-connected application. When you start another Crowd-connected application, you will be logged in automatically.
- When you log out of Crowd or one of the Crowd-connected applications, you will be logged out of Crowd and the other application(s) at the same time.

Crowd also manages the information held about you as a user of other software applications:

- Your login permissions to various applications.
- The password you use to log in to those applications.
- The groups and roles you belong to, which are used by the applications to decide which functions you can perform within the applications.
- The user directories which hold your information.

Search the User Guide

About the User Guide

The ***Crowd User Guide*** contains information for people who use Crowd to update their user profiles and passwords and to view their groups, roles and applications.

If you need information about installing Crowd, configuring your Crowd server or using the Crowd Administration Console, please visit the Crowd documentation home page.

If you have a question about using Crowd that hasn't been answered here, please let us know.

Download

You can download the Crowd documentation in PDF, HTML or XML formats.

Getting Help

Support | Feature requests and bug reports | Forums | Knowledge base

Table of Contents

Introduction to Crowd

Logging in to Crowd

Logging out of Crowd

Changing or Resetting your Password

- Changing your Password
- Resetting your Password

Updating your User Profile

Viewing your Group Membership

Viewing your Role Membership

Viewing your Applications

Crowd User's Glossary

- Alias (Glossary Entry)
- Authorisation to Use Crowd (Glossary Entry)
- Crowd Administrator (Glossary Entry)
- Crowd-Connected Application (Glossary Entry)
- Directory (Glossary Entry)
- Self-Service Console (Glossary Entry)
- Single Sign-On (Glossary Entry)

Introduction to Crowd

This page gives a brief introduction to Crowd, for people who will view and update their login and user profile information in Crowd.

What is Crowd?

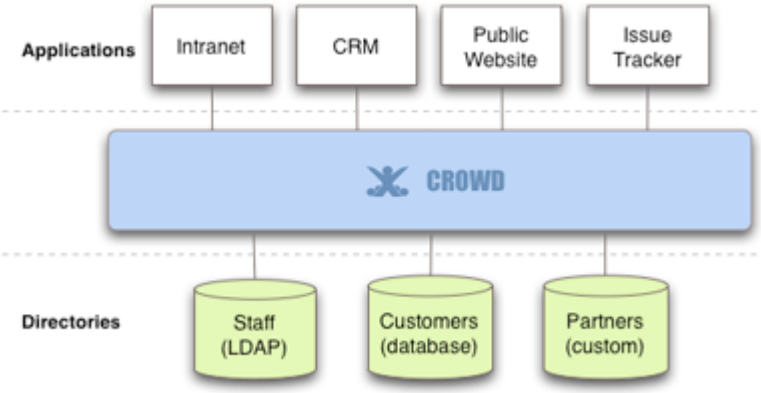
Atlassian's Crowd is a software application installed by the system administrator. The administrator will also connect one or more of your organisation's applications to Crowd. When you log in to a Crowd-connected application, Crowd will verify your password and login permissions.

Using Crowd for single sign-on (SSO), each person needs only one username and password to access all web applications. You can host your own OpenID provider to include external applications.

- You only need to log in once, to Crowd or a Crowd-connected application. When you start another Crowd-connected application, you will be logged in automatically.
- When you log out of Crowd or one of the Crowd-connected applications, you will be logged out of Crowd and the other application(s) at the same time.

Crowd also manages the information held about you as a user of other software applications:

- Your login permissions to various applications.
- The password you use to log in to those applications.
- The groups and roles you belong to, which are used by the applications to decide which functions you can perform within the applications.
- The user directories which hold your information.



Using Crowd

The Crowd administrator has access to Crowd's Administration Console, which provides the functions described in the Crowd Administration Guide.

Every authorised Crowd user has access to Crowd's Self-Service Console, where you can edit your user profile, change your password and view other information about your Crowd username. The Crowd User Guide describes this functionality.

Some Terminology

Here is a list of all entries in the glossary, plus the first few lines of content. Click a link to see the full text for each entry.

- Alias (Glossary Entry) — Crowd allows you to have different usernames in different applications. These different usernames are called 'aliases'. Your Crowd administrator can manage your aliases for the applications you are authorised to access.
- Authorisation to Use Crowd (Glossary Entry) — If you are authorised to use Crowd, you can log in to Crowd's Self-Service Console to update your user profile and view other information about your username. The Crowd administrator can grant people access to the Self-Service Console, as described in the Crowd Administration Guide. Basically, the administrator should ensure that your username is in a user directory which is mapped to the Crowd application.
- Crowd Administrator (Glossary Entry) — A Crowd administrator is a user who has access to the **Crowd Administration Console**, which provides the functions described in the Crowd Administration Guide. The first administrator is defined during the installation of Crowd. A Crowd administrator can grant administration rights to other users, as described in the Crowd Administration Guide.

- Crowd-Connected Application (Glossary Entry) — A 'Crowd-connected application' is a software application which has been defined to and integrated with Crowd. These applications pass all login requests to Crowd for authentication. Depending on the integration level, the application may also make use of the groups and roles defined in Crowd for authorisation purposes, and allow single sign-on across the Crowd domain. The Crowd Administration Guide tells you how to connect an application to Crowd.
- Directory (Glossary Entry) — Crowd uses the term 'directory', or 'user directory', to refer to a store of information about a user. Typically, a directory will hold your username, name, password, email address, and so on. Your Crowd administrator can define one or more directories internally in Crowd or connect one or more external directories to Crowd. The external directory may be a corporate directory such as Microsoft's Active Directory. To learn more about Crowd's directory management, please refer to the Crowd Administration Guide.
- Self-Service Console (Glossary Entry) — Authorised Crowd users can access the Crowd Console, even if they are not Crowd administrators. Non-administrators will see a subset of the Crowd Console functionality, which we call the 'Self-Service Console'. The Crowd User Guide describes this functionality. The Crowd Administration Console presents the full range of Crowd administration functionality to authorised Crowd administrators.
- Single Sign-On (Glossary Entry) — Single sign-on (SSO) is a feature offered by Crowd. Your Crowd administrator can choose to enable this feature for the Crowd-connected applications. If SSO is enabled, you will only need to log in or log out once. Specifically:

RELATED TOPICS

Logging in to Crowd
Crowd User Guide

Logging in to Crowd

If you are authorised to use Crowd, you can log in to Crowd's Self-Service Console to update your user profile and view other information about your username. The Crowd administrator can grant people access to the Self-Service Console, as described in the Crowd Administration Guide. Basically, the administrator should ensure that your username is in a user directory which is mapped to the Crowd application.

If your administrator has configured Crowd to allow single sign-on (SSO), then you only need to log in once. When you start another Crowd-connected application, you will be logged in automatically.

On this page:

- How to Log In
- User Aliases
- SSO and Google Apps

How to Log In

To log in to Crowd,

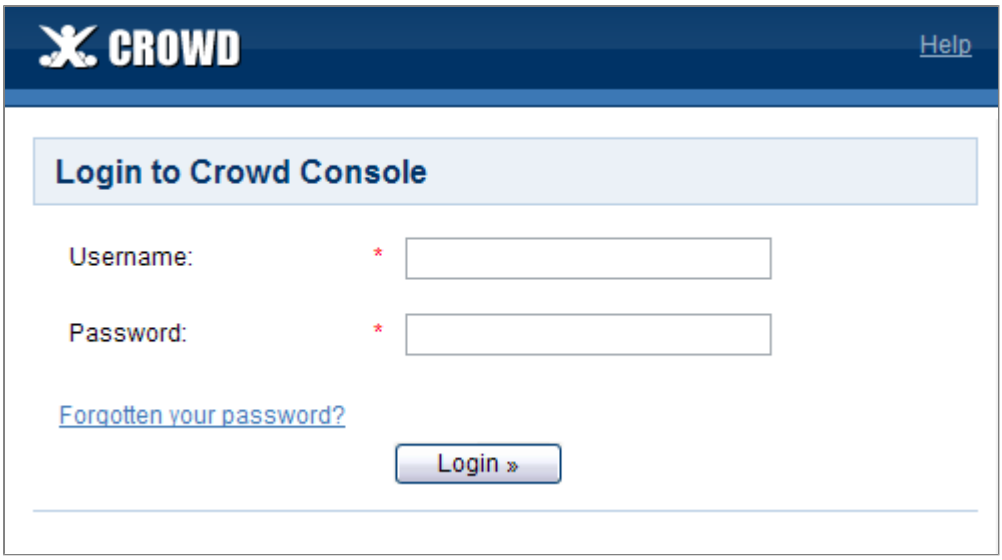
1. Open Crowd in your web browser. In most cases, you will do this by typing an address like this one into the browser's address bar:

   ```
   1.http://YOUR-CROWD-LOCATION:8095/crowd/
   ```

   Replace '`YOUR-CROWD-LOCATION`' with the address of your Crowd server. (Ask your Crowd administrator for this address.)

2. The Crowd login screen will appear, as shown in the screenshot below. Type in your username and password.
3. Click the '**Log In**' button.

Screenshot: Crowd login screen



If you have forgotten your password, you can click the '**Forgotten your password**' link. Crowd will email you a new password. Read more about resetting your password.

User Aliases

Crowd allows you to have different usernames in different applications. These different usernames are called 'aliases'. Your Crowd administrator can manage your aliases for the applications you are authorised to access.

- When you log in to Crowd itself, you must use your primary username i.e. the one registered in Crowd.
- If you choose to log in to another Crowd-connected application directly, such as Confluence or JIRA, instead of logging in via Crowd, then you must log in using the alias registered in that application (Confluence, JIRA, or whatever.)
- If SSO is enabled you will only need to log in or log out once, to Crowd or a Crowd-connected application. When you start another Crowd-connected application, you will be logged in automatically.

SSO and Google Apps

These notes are relevant if your Crowd administrator has enabled single sign-on between Crowd and Google Apps:

- Single sign-on (SSO) applies only to the applications within Google Apps. The Google Apps administration section (control panel) does not support SSO.
- When you sign out of Google Apps, you will also be signed out of Crowd and all Crowd-connected applications. This is the usual SSO behaviour.
- But when you sign out of Crowd, you will remain logged in to Google Apps even though you will be logged out of other Crowd-connected applications. (Reason: Google does not rely on a cookie, so there is no easy way for Crowd to tell Google you have signed out.)

  It would take some additional development to support single sign-out from Google Apps. If you would like to see this work undertaken, please vote for issue CWD-1238.
- If you go directly to a Google Apps application without logging in to Crowd, Google Apps direct you to a Crowd login screen.
- The Crowd login screen for Google Apps will not offer a 'Forgotten your password' link. You cannot change your Crowd password via Google Apps. Instead, if you need to change your password please log in to Crowd directly, by going to this URL: http://YOUR-CROWD-LOCATION:8095/crowd/

RELATED TOPICS

Logging out of Crowd
Resetting your Password
Crowd User Guide

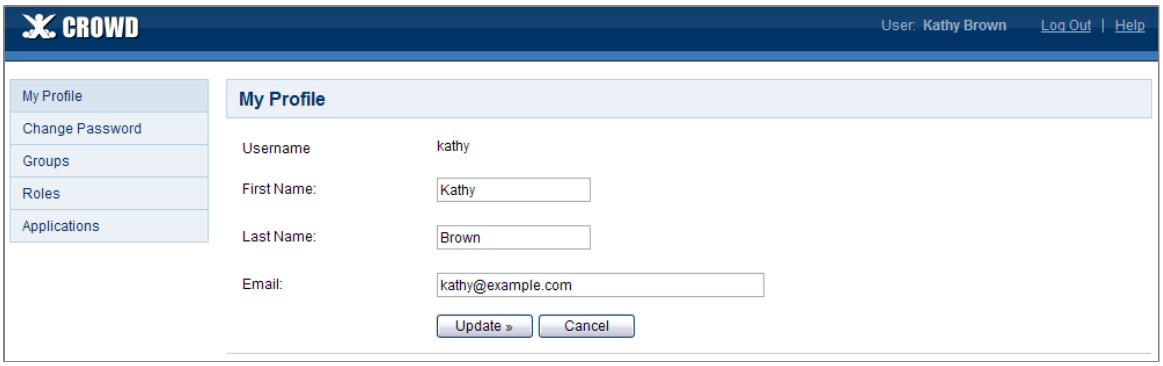
Logging out of Crowd

Logging out of Crowd is easy — just click the '**Log Out**' link at the top of the Crowd screen.

If your administrator has configured Crowd to allow single sign-on (SSO), then you will be automatically logged out of all Crowd-connected applications when you log out of Crowd.

This automatic logout will also happen if you log out of one of the other Crowd-connected applications — you will be logged out of Crowd and the other application(s) at the same time.

Screenshot: Crowd screen showing 'Log Out' link



SSO and Google Apps

- Single sign-on (SSO) applies only to the applications within Google Apps. The Google Apps administration section (control panel) does not support SSO.
- When you sign out of Google Apps, you will also be signed out of Crowd and all Crowd-connected applications. This is the usual SSO behaviour.
- But when you sign out of Crowd, you will remain logged in to Google Apps even though you will be logged out of other Crowd-connected applications. (Reason: Google does not rely on a cookie, so there is no easy way for Crowd to tell Google you have signed out.)

  It would take some additional development to support single sign-out from Google Apps. If you would like to see this work undertaken, please vote for issue CWD-1238.
- If you go directly to a Google Apps application without logging in to Crowd, Google Apps direct you to a Crowd login screen.
- The Crowd login screen for Google Apps will not offer a 'Forgotten your password' link. You cannot change your Crowd password via Google Apps. Instead, if you need to change your password please log in to Crowd directly, by going to this URL: http://YOUR-CROWD-LOCATION:8095/crowd/

RELATED TOPICS

Logging in to Crowd
Crowd User Guide

Changing or Resetting your Password

If you are authorised to use Crowd, you can log in to Crowd's Self-Service Console and change your password.

When attempting to log in to Crowd, you can also reset your password. This is useful if you have forgotten the old one.

Password change applies to one user directory only
In most cases, your username will be defined in one user directory only. But some organisations may have more than one user directory. For example, your username may be defined in Crowd for JIRA use, and also in another Crowd-connected directory (e.g. LDAP) for use in another application. If you change or reset your password, the new password will apply only in one directory — the directory mapped to the '**crowd**' application and defined as **first** in the directory sequence. Your Crowd administrator can define the order of the directories, as described in the Crowd Administration Guide.

RELATED TOPICS

Logging in to Crowd
Crowd User Guide

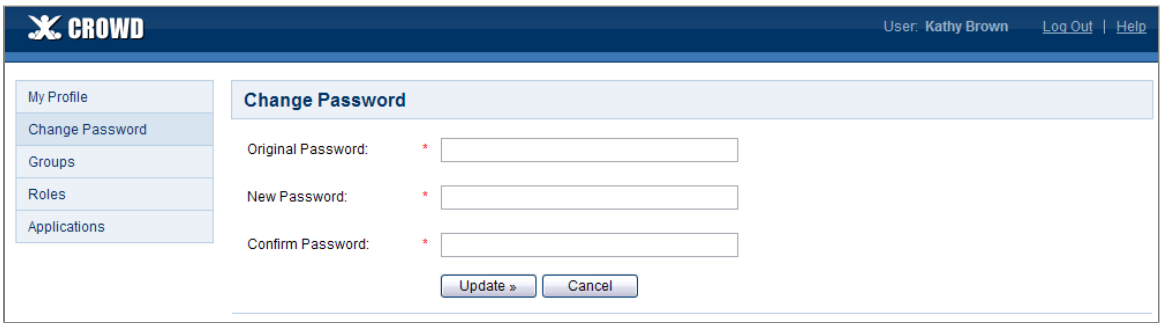
Changing your Password

If you are authorised to use Crowd, you can log in to Crowd's Self-Service Console and change your password.

To change your password,

1. Log in to Crowd.
2. If you are not a Crowd administrator, you can skip this step because you will go directly to the Crowd Self-Service Console.
    - If you are a Crowd administrator, the **Crowd Administration Console** will open. Click the '**My Profile**' link in the top navigation bar.
3. The **Crowd Self-Service Console** will open.
4. Click '**Change Password**' in the left-hand menu.
5. The '**Change Password**' screen will appear, as shown in the screenshot below. Enter the following information:
    - **Original Password** — Your current password.
    - **New Password** — Your new password.
    - **Confirm Password** — Your new password again, to verify that you typed it correctly the first time.
6. Click the '**Update**' button.
7. If the change is successful, a '**Password updated**' message will appear on the screen.

Screenshot: Crowd's Change Password Screen



Password change applies to one user directory only
In most cases, your username will be defined in one user directory only. But some organisations may have more than one user directory. For example, your username may be defined in Crowd for JIRA use, and also in another Crowd-connected directory (e.g. LDAP) for use in another application. If you change or reset your password, the new password will apply only in one directory — the directory mapped to the '**crowd**' application and defined as **first** in the directory sequence. Your Crowd administrator can define the order of the directories, as described in the Crowd Administration Guide.

RELATED TOPICS

Resetting your Password
Logging in to Crowd
Crowd User Guide

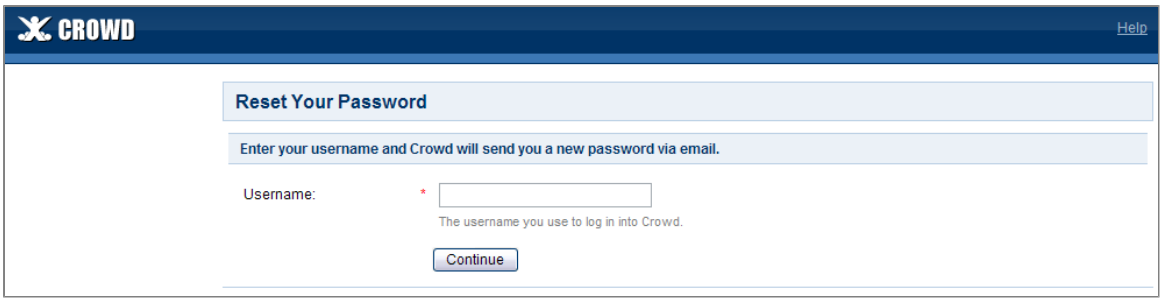
Resetting your Password

The Crowd '**Login**' screen allows you to reset your password. This is useful when you have forgotten the password.

To reset your password,

1. Open Crowd in your browser.
2. Click the '**Forgotten your password**' link on the Crowd login screen.
3. The '**Reset Your Password**' screen will appear, as shown in the screenshot below. Type in your Crowd username and click the '**Continue**' button.
4. A message will appear: '**Your new password is on the way!**' Click the '**Home**' link at the top of the screen.
5. You will receive an email message with your new password. Copy the password.
6. Log in to Crowd using the new password.
7. Change your password to one you can remember easily.

Screenshot: Crowd's Reset Your Password screen



Password change applies to one user directory only
In most cases, your username will be defined in one user directory only. But some organisations may have more than one user directory. For example, your username may be defined in Crowd for JIRA use, and also in another Crowd-connected directory (e.g. LDAP) for use in another application. If you change or reset your password, the new password will apply only in one directory — the directory mapped to the '**crowd**' application and defined as **first** in the directory sequence. Your Crowd administrator can define the order of the directories, as described in the Crowd Administration Guide.

RELATED TOPICS

Changing your Password
Logging in to Crowd
Crowd User Guide

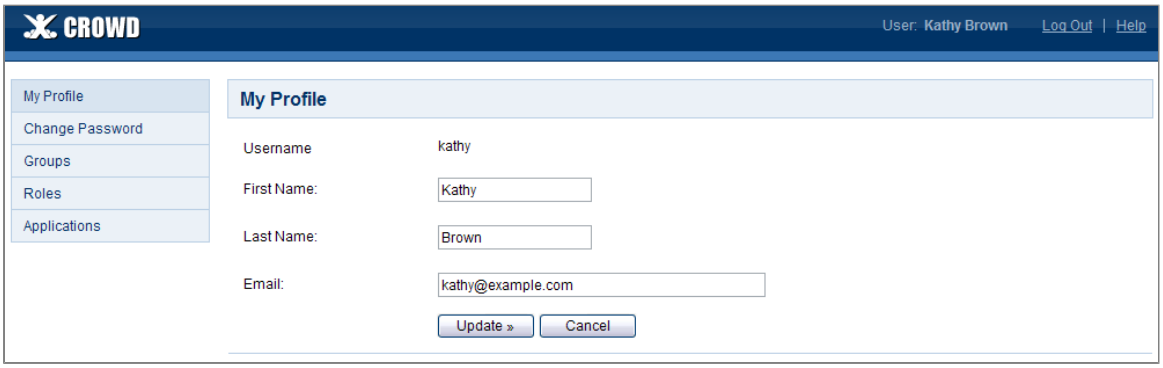
Updating your User Profile

Provided that you are authorised to use Crowd, you can change the profile information for your username.

To update your user profile,

1. Log in to Crowd.
2. If you are not a Crowd administrator, you can skip this step because you will go directly to the Crowd Self-Service Console.
    - If you are a Crowd administrator, the **Crowd Administration Console** will open. Click the '**My Profile**' link in the top navigation bar.
3. The **My Profile** screen will open, as shown in the screenshot below.
4. Update your profile information where necessary:
    - **First Name** — Your first name.
    - **Last Name** — Your last name or surname.
    - **Email** — Crowd will use this email address when sending you messages, such as a new password if you reset your password.

Screenshot: Crowd user profile



Which user directories are updated?
In most cases, your username will be defined in one user directory only. But some organisations may have more than one user directory. For example, your username may be defined in Crowd for JIRA use, and also in another Crowd-connected directory (e.g. LDAP) for use in another application. If you change your profile details, the change will be applied to **all** directories which the '**crowd**' application has permission to update. Your Crowd administrator defines the application permissions, as described in the Crowd Administration Guide.

RELATED TOPICS

Changing or Resetting your Password
Crowd User Guide

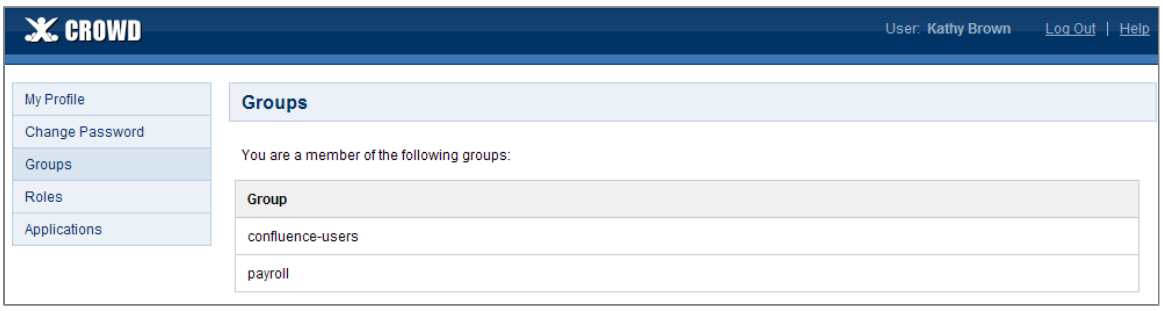
Viewing your Group Membership

Provided that you are authorised to use Crowd, you can see a list of the groups to which your username belongs.

To see which groups you belong to,

1. Log in to Crowd.
2. If you are not a Crowd administrator, you can skip this step because you will go directly to the Crowd Self-Service Console.
    - If you are a Crowd administrator, the **Crowd Administration Console** will open. Click the '**My Profile**' link in the top navigation bar.
3. The **Crowd Self-Service Console** will open. Click '**Groups**' in the left-hand menu.
4. The '**Groups**' screen will appear, as shown in the screenshot below.

Screenshot: Groups



Each group appears only once
Even if you are a member of the same group in more than one directory, the group name will appear only once on this screen. *More explanation:* In most cases, your username will be defined in one user directory only. But some organisations may have more than one user directory. For example, your username may be defined in Crowd as a Crowd administrator, and also in another Crowd-connected directory (e.g. LDAP). In addition, you may then be a member of the same group (e.g. 'confluence-users') in both directories. On the Crowd '**Groups**' screen, the group 'confluence-users' will appear only once.

RELATED TOPICS

Crowd User Guide

Viewing your Role Membership

Provided that you are authorised to use Crowd, you can see a list of the roles to which your username is assigned.

To see which roles you have been assigned,

1. Log in to Crowd.
2. If you are not a Crowd administrator, you can skip this step because you will go directly to the Crowd Self-Service Console.
    - If you are a Crowd administrator, the **Crowd Administration Console** will open. Click the '**My Profile**' link in the top navigation bar.
3. The **Crowd Self-Service Console** will open. Click '**Roles**' in the left-hand menu.
4. The '**Roles**' screen will appear, as shown in the screenshot below.

Screenshot: Roles

Each role appears only once
Even if you are a member of the same role in more than one directory, the role name will appear only once on this screen.
More explanation: In most cases, your username will be defined in one user directory only. But some organisations may have more than one user directory. For example, your username may be defined in Crowd as a Crowd administrator, and also in another Crowd-connected directory (e.g. LDAP). In addition, you may then be a member of the same role (e.g. 'hr-admin') in both directories. On the Crowd '**Roles**' screen, the role 'hr-admin' will appear only once.

RELATED TOPICS

Crowd User Guide

Viewing your Applications

Provided that you are authorised to use Crowd, you can see a list of the applications you are authorised to log in to.

More information about the applications listed:

- Crowd verifies all logins to these applications. Your Crowd administrator has defined them as Crowd-connected applications.
- Your username is authorised to log in to these applications. Your Crowd administrator has made you a member of a directory or a group which is mapped to the application.

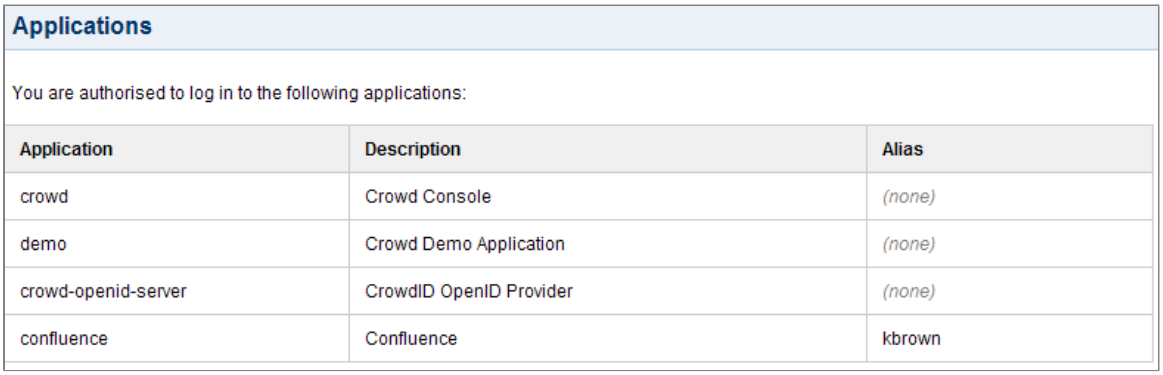
Crowd allows you to have different usernames in different applications. These different usernames are called 'aliases'. Your Crowd administrator can manage your aliases for the applications you are authorised to access.

- When you log in to Crowd itself, you must use your primary username i.e. the one registered in Crowd.
- If you choose to log in to another Crowd-connected application directly, such as Confluence or JIRA, instead of logging in via Crowd, then you must log in using the alias registered in that application (Confluence, JIRA, or whatever.)
- If SSO is enabled you will only need to log in or log out once, to Crowd or a Crowd-connected application. When you start another Crowd-connected application, you will be logged in automatically.

To see the applications which you can log in to,

1. Log in to Crowd.
2. If you are not a Crowd administrator, you can skip this step because you will go directly to the Crowd Self-Service Console.
    - If you are a Crowd administrator, the **Crowd Administration Console** will open. Click the '**My Profile**' link in the top navigation bar.
3. The **Crowd Self-Service Console** will open. Click '**Applications**' in the left-hand menu.
4. The '**Applications**' screen will appear, as shown in the screenshot below.

Screenshot: Applications



The 'crowd' application
One of the applications listed will be the '**crowd**' application. This is the Crowd Administration and Self-Service Console. If you can log in to Crowd, that means that you do have access to the 'crowd' application and you should see it in the list.

RELATED TOPICS

Viewing your Group Membership
Crowd User Guide

Crowd User's Glossary

Here is a list of all entries in the glossary, plus the first few lines of content. Click a link to see the full text for each entry.

- Alias (Glossary Entry) — Crowd allows you to have different usernames in different applications. These different usernames are called 'aliases'. Your Crowd administrator can manage your aliases for the applications you are authorised to access.
- Authorisation to Use Crowd (Glossary Entry) — If you are authorised to use Crowd, you can log in to Crowd's Self-Service Console to update your user profile and view other information about your username. The Crowd administrator can grant people access to the Self-Service Console, as described in the Crowd Administration Guide. Basically, the administrator should ensure that your username is in a user directory which is mapped to the Crowd application.
- Crowd Administrator (Glossary Entry) — A Crowd administrator is a user who has access to the **Crowd Administration Console**, which provides the functions described in the Crowd Administration Guide. The first administrator is defined during the installation of Crowd. A Crowd administrator can grant administration rights to other users, as described in the Crowd Administration Guide.
- Crowd-Connected Application (Glossary Entry) — A 'Crowd-connected application' is a software application which has been defined to and integrated with Crowd. These applications pass all login requests to Crowd for authentication. Depending on the integration level, the application may also make use of the groups and roles defined in Crowd for authorisation purposes, and allow single sign-on across the Crowd domain. The Crowd Administration Guide tells you how to connect an application to Crowd.
- Directory (Glossary Entry) — Crowd uses the term 'directory', or 'user directory', to refer to a store of information about a user. Typically, a directory will hold your username, name, password, email address, and so on. Your Crowd administrator can define one or more directories internally in Crowd or connect one or more external directories to Crowd. The external directory may be a corporate directory such as Microsoft's Active Directory. To learn more about Crowd's directory management, please refer to the Crowd Administration Guide.
- Self-Service Console (Glossary Entry) — Authorised Crowd users can access the Crowd Console, even if they are not Crowd administrators. Non-administrators will see a subset of the Crowd Console functionality, which we call the 'Self-Service Console'. The Crowd User Guide describes this functionality. The Crowd Administration Console presents the full range of Crowd administration functionality to authorised Crowd administrators.
- Single Sign-On (Glossary Entry) — Single sign-on (SSO) is a feature offered by Crowd. Your Crowd administrator can choose to enable this feature for the Crowd-connected applications. If SSO is enabled, you will only need to log in or log out once. Specifically:

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Alias (Glossary Entry)

Crowd allows you to have different usernames in different applications. These different usernames are called 'aliases'. Your Crowd administrator can manage your aliases for the applications you are authorised to access.

- When you log in to Crowd itself, you must use your primary username i.e. the one registered in Crowd.
- If you choose to log in to another Crowd-connected application directly, such as Confluence or JIRA, instead of logging in via Crowd, then you must log in using the alias registered in that application (Confluence, JIRA, or whatever.)
- If SSO is enabled you will only need to log in or log out once, to Crowd or a Crowd-connected application. When you start another Crowd-connected application, you will be logged in automatically.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide
Overview of SSO

Authorisation to Use Crowd (Glossary Entry)

If you are authorised to use Crowd, you can log in to Crowd's Self-Service Console to update your user profile and view other information about your username. The Crowd administrator can grant people access to the Self-Service Console, as described in the Crowd Administration Guide. Basically, the administrator should ensure that your username is in a user directory which is mapped to the Crowd application.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Crowd Administrator (Glossary Entry)

A Crowd administrator is a user who has access to the **Crowd Administration Console**, which provides the functions described in the Crowd Administration Guide. The first administrator is defined during the installation of Crowd. A Crowd administrator can grant administration rights to other users, as described in the Crowd Administration Guide.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Crowd-Connected Application (Glossary Entry)

A 'Crowd-connected application' is a software application which has been defined to and integrated with Crowd. These applications pass all login requests to Crowd for authentication. Depending on the integration level, the application may also make use of the groups and roles defined in Crowd for authorisation purposes, and allow single sign-on across the Crowd domain. The Crowd Administration Guide tells you how to connect an application to Crowd.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Directory (Glossary Entry)

Crowd uses the term 'directory', or 'user directory', to refer to a store of information about a user. Typically, a directory will hold your username, name, password, email address, and so on. Your Crowd administrator can define one or more directories internally in Crowd or connect one or more external directories to Crowd. The external directory may be a corporate directory such as Microsoft's Active Directory. To learn more about Crowd's directory management, please refer to the Crowd Administration Guide.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Self-Service Console (Glossary Entry)

Authorised Crowd users can access the Crowd Console, even if they are not Crowd administrators. Non-administrators will see a subset of the Crowd Console functionality, which we call the 'Self-Service Console'. The Crowd User Guide describes this functionality. The Crowd Administration Console presents the full range of Crowd administration functionality to authorised Crowd administrators.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Single Sign-On (Glossary Entry)

Single sign-on (SSO) is a feature offered by Crowd. Your Crowd administrator can choose to enable this feature for the Crowd-connected applications. If SSO is enabled, you will only need to log in or log out once. Specifically:

- You only need to log in once, to Crowd or a Crowd-connected application. When you start another Crowd-connected application, you will be logged in automatically.
- When you log out of Crowd or one of the Crowd-connected applications, you will be logged out of Crowd and the other application(s) at the same time.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide
Overview of SSO

Crowd Administration Guide

The *Crowd Administration Guide* is for people who have Crowd administration rights.

Table of Contents

- Getting Started
  - Concepts
  - Supported Applications and Directories
  - About the Crowd Administration Console
- Managing Directories
  - Using the Directory Browser
  - Adding a Directory
    - Configuring an Internal Directory
    - Configuring an LDAP Directory Connector
      - Apache Directory Server (ApacheDS)
      - Apple Open Directory
      - Fedora Directory Server
      - Generic LDAP Directories
      - Microsoft Active Directory
        - Configuring an SSL Certificate for Microsoft Active Directory
      - Novell eDirectory
      - OpenDS
      - OpenLDAP
      - OpenLDAP Using Posix Schema
      - Posix Schema for LDAP
      - Sun Directory Server Enterprise Edition (DSEE)
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
  - Configuring Caching for an LDAP Directory
  - Configuring Relaxed DN Standardisation
  - Specifying Directory Permissions
  - Importing Users and Groups into a Directory
    - Importing Users from Atlassian Confluence
    - Importing Users from Atlassian JIRA
    - Importing Users from Atlassian Bamboo
    - Importing Users from Jive Forums
    - Importing Users from CSV Files
      - Configuring the CSV Importer
      - Mapping CSV Fields to Crowd Fields
      - Confirming the CSV Importer Configuration
      - Viewing the Results of the Import
    - Importing Users from One Crowd Directory into Another
- Managing Applications
  - Using the Application Browser
  - Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
      - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
      - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
      - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
      - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
      - Jive SSO
    - Integrating Crowd with Spring Security
      - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
  - Configuring the Google Apps Connector
  - Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
      - Example of Directory Permissions
  - Specifying which Groups can access an Application
  - Viewing Users in Directories Mapped to an Application
  - Specifying an Application's Address or Hostname
  - Testing a User's Login to an Application
  - Enforcing Lower-Case Usernames, Groups and Roles for an Application
  - Managing an Application's Session
  - Deleting or Deactivating an Application
  - Configuring Caching for an Application
  - Overview of SSO
  - Configuring Options for an Application
- Managing Users, Groups and Roles
  - Using the User Browser
  - Adding a User
  - Deleting or Deactivating a User
  - Managing a User's Session
  - Editing a User's Details and Password

        - Specifying a User's Attributes
        - Specifying a User's Aliases
        - Granting Crowd Administration Rights to a User
        - Granting Crowd User Rights to a User
        - Managing Groups and Roles
            - Adding a Group or Role
            - Deleting or Deactivating a Group
        - Managing Group Members
            - Automatically Assigning New Users to Groups
            - Adding Users to a Group
            - Removing Users from a Group
            - Nested Groups in Crowd
            - Adding a Sub-Group
            - Removing a Sub-Group
        - Case Sensitivity of Usernames, Groups and Roles
        - Editing a User's Group and Role Membership
    - System Administration
        - Configuring Server Settings
            - Deployment Title
            - Domain
            - Token Seed
            - Session Configuration
            - Authorisation Caching
            - Compression of Server Output
            - Licensing
            - SSO Cookie
        - Configuring your Mail Server
        - Creating an Email Notification Template
        - Configuring Trusted Proxy Servers
        - Viewing Crowd's System Information
        - Backing Up and Restoring Data
        - Logging and Profiling
            - Performance Profiling
        - Overview of Caching
    - Crowd Security Advisories and Fixes
        - Crowd Security Advisory 2008-10-14 - Parameter Injection Vulnerability

Getting Started

- Concepts
- Supported Applications and Directories
- About the Crowd Administration Console

Concepts

Crowd is an application security framework that handles authentication and authorisation for your web-based applications. With Crowd you can quickly integrate multiple web applications into a single security architecture that supports single sign-on (SSO) and centralised identity management.

Crowd has the following components:

- The **Crowd Administration Console** is a clean and powerful web-interface for managing directories, users (known in Crowd as 'principals') and their security rights ('permissions'). Refer to the Crowd Administration Guide for details.
- The **Crowd Self-Service Console** allows authorised users to maintain their user profiles and passwords and to view their usernames, groups, roles and applications. Refer to the Crowd User Guide for details.
- The **Crowd integration API** provides a platform-neutral way to integrate web applications into a single security architecture. With the integration API, applications can quickly access user information and perform security checks.

Designed for ease of use, Crowd can be deployed with your existing infrastructure. Crowd supports:

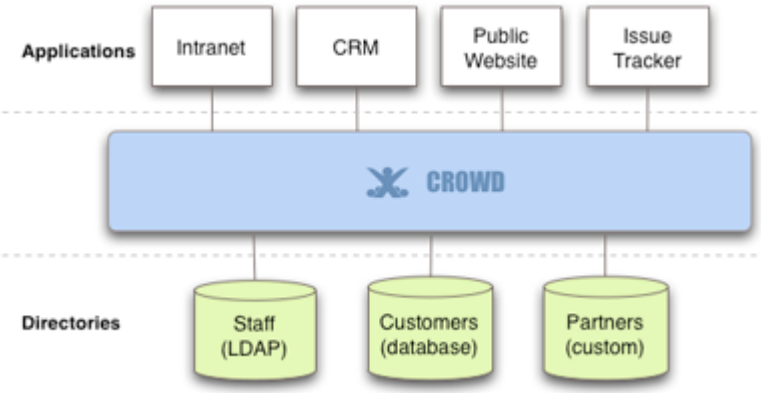
- Java, .NET and PHP applications.
- Popular directory servers such as Microsoft Active Directory, Sun ONE and OpenLDAP. Additionally, custom directory connectors may be developed using the Crowd integration API.

See the list of supported applications and directories.

Architectural Overview

Crowd is a middleware application that integrates web applications into a single security architecture, supporting single sign-on and centralised identity management. Crowd works by dispatching authentication and authorisation calls from configured applications to configured directories.

A typical deployment may be similar to the following:



When an application needs to validate a security or authentication request (e.g. when a user attempts to log in to the application) the application will make a simple API call to the Crowd framework, which will then forward the call to the appropriate directory.

About Applications

Crowd integrates and provisions applications. Once defined, an application is mapped to a directory(s), whose users are then granted access to the application. Note that an application can only communicate with Crowd when the application uses a known host address.

About Directories

Crowd supports an unlimited number of user directories. A directory can be one of the following types:

- Internal to Crowd.
- Connected to Crowd via an LDAP connector (e.g. for Active Directory), with all authentication and user/group/role management in LDAP.
- A Crowd internal directory for user/group/role management but with authentication delegated to LDAP (e.g. Active Directory).
- Connected via a custom directory connector (e.g. for a legacy database).

Once you have defined a directory in Crowd, you can map it to applications. Crowd will then pass authentication and authorisation requests to the directory, for all applications that are mapped to that directory. Modification of directory entities (users, groups and roles) can be done via the Crowd Administration Console or via the application, depending on the application's capabilities.

You can even map multiple directories to an application, providing the application with a single view of multiple directories in a specified order .

RELATED TOPICS

- Concepts
- Supported Applications and Directories
- About the Crowd Administration Console

Crowd Documentation

Supported Applications and Directories

Crowd integrates and provisions applications. Once defined, an application is mapped to one or more directories, whose users are then granted access to the application. This page lists the supported application and directory connectors.

Application Connectors

- Atlassian JIRA
- Atlassian Confluence
- Atlassian Bamboo
- Atlassian Fisheye
- Atlassian Crucible
- Google Apps
- Apache
- Subversion
- Jive Forums
- Atlassian CrowdID
- Acegi
- NTLM for JIRA — Third-party plugin not officially supported by Atlassian
- NTLM for Confluence — Third-party plugin not officially supported by Atlassian

You can also add your own custom applications.

Directory Connectors

Connecting to LDAP directories:

- Apache Directory Server (ApacheDS)
- Apple Open Directory
- Fedora Directory Server
- Generic LDAP Directory
- Microsoft Active Directory
- Novell eDirectory
- OpenLDAP
- OpenLDAP using Posix Schema
- OpenDS
- Posix Schema for LDAP
- Sun Java System Directory Server Enterprise Edition (DSEE, previously called SunONE)

Using Crowd's internal directories:

- Internal Crowd Directory
- Delegated Authentication Directory, combining the features of an internal Crowd directory with delegated LDAP authentication.

You can also add a connector to your own custom directory.

RELATED TOPICS

Concepts
Adding an Application
Adding a Directory
Crowd Documentation

About the Crowd Administration Console

The **Crowd Administration Console** presents the full range of Crowd administration functionality to authorised Crowd administrators.

Authorised Crowd users who are **not** administrators can also access the Crowd Console. They will see a subset of functionality, which we call the '**Self-Service Console**'. Refer to the Crowd User Guide for details.

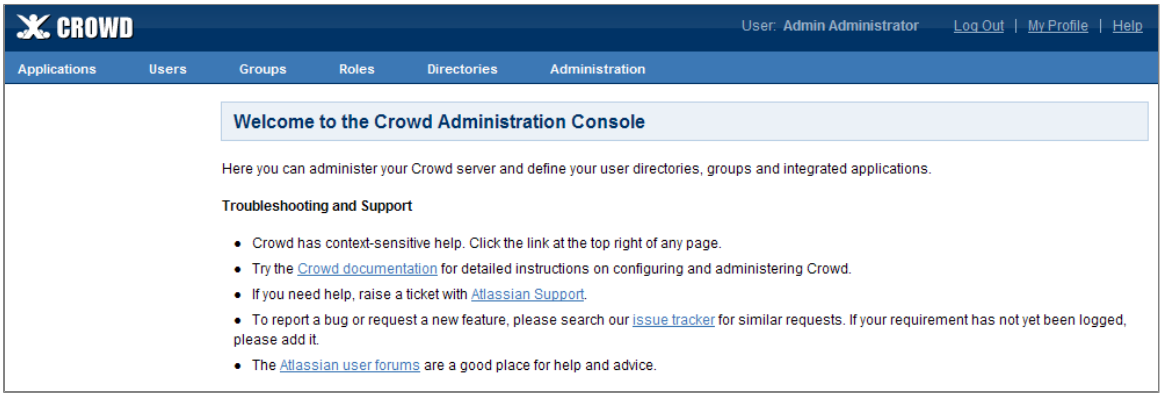
If you are a Crowd administrator, the Crowd Administration Console allows you to perform the following functions:

- Configure applications to access the Crowd framework.
- Create and manage users and adjust their group and role membership.
- Map directories to allow users to access integrated applications.
- Adjust server deployment properties, including those configured during the setup process.
- Back up and restore your Crowd data.
- View active sessions and manually expire sessions.
- View Crowd system information.
- Update your user profile and password and view the groups, roles and applications associated with your username. Refer to the Crowd User Guide for details.

To access the Crowd Administration Console,

1. Go to the URL http://localhost:8095/crowd or http://localhost:8095/crowd/console.

The welcome screen will appear, looking something like this:



The Crowd Administration Console is a web application provisioned by Crowd — you can see it in the list of applications shown in the Application Browser.

RELATED TOPICS

- Concepts
- Supported Applications and Directories
- About the Crowd Administration Console

Crowd User Guide
Crowd Documentation

Managing Directories

Crowd supports an unlimited number of user directories. A directory can be one of the following types:

- Internal to Crowd.
- Connected to Crowd via an LDAP connector (e.g. for Active Directory), with all authentication and user/group/role management in LDAP.
- A Crowd internal directory for user/group/role management but with authentication delegated to LDAP (e.g. Active Directory).
- Connected via a custom directory connector (e.g. for a legacy database).

Once you have defined a directory in Crowd, you can map it to applications. Crowd will then pass authentication and authorisation requests to the directory, for all applications that are mapped to that directory. Modification of directory entities (users, groups and roles) can be done via the Crowd Administration Console or via the application, depending on the application's capabilities.

You can even map multiple directories to an application, providing the application with a single view of multiple directories in a specified order .

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using the Directory Browser

About Directories

Crowd supports an unlimited number of user directories. A directory can be one of the following types:

- Internal to Crowd.
- Connected to Crowd via an LDAP connector (e.g. for Active Directory), with all authentication and user/group/role management in LDAP.
- A Crowd internal directory for user/group/role management but with authentication delegated to LDAP (e.g. Active Directory).
- Connected via a custom directory connector (e.g. for a legacy database).

Once you have defined a directory in Crowd, you can map it to applications. Crowd will then pass authentication and authorisation requests to the directory, for all applications that are mapped to that directory. Modification of directory entities (users, groups and roles) can be done via the Crowd Administration Console or via the application, depending on the application's capabilities.

You can even map multiple directories to an application, providing the application with a single view of multiple directories in a specified order .

About the Directory Browser

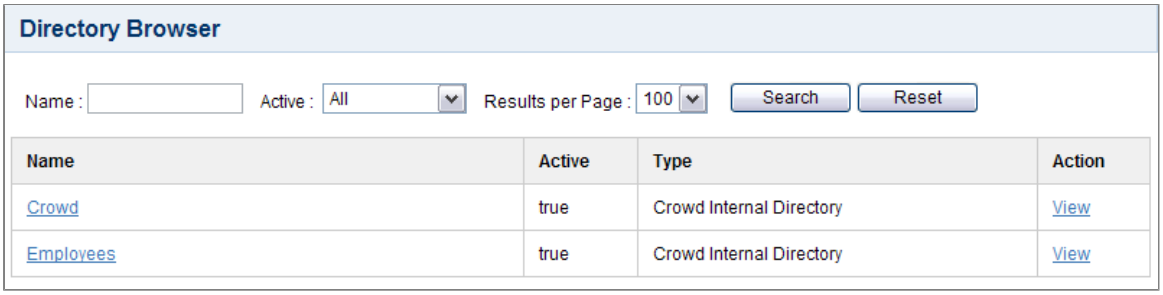
The Directory Browser allows you to view and search for configured directories.

To use the Directory Browser,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' tab in the top navigation bar.
3. This will display the Directory Browser, showing all the directories that exist in your Crowd system. You can refine your search by specifying a '**Name**' (note that this is case-sensitive), or '**Active**'/'**Inactive**' directories.
   An 'Inactive' directory cannot be used by any applications, regardless of whether or not they are mapped to it.
4. To view or edit a directory's details, click the '**View**' link.

You created one default directory when you set up Crowd. To add more directories, see Adding a Directory

Screenshot: 'Directory Browser'



RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Crowd Documentation

Adding a Directory

Directories contain authentication and authorisation information about users, groups and roles. Crowd supports an unlimited number of directories. Administrators can use different directories to create silos of users. For example, you might store your customers in one directory and your employees in another.

Crowd supports the following types of directory:

- Crowd Internal Directory
  Internal directories use the Crowd database to store user, group and role information. Internal directories are stored in Crowd's database server.
- Delegated Authentication Directory

  A Delegated Authentication directory combines the features of an internal Crowd directory with delegated LDAP authentication. This means that you can have your users authenticated via an external LDAP directory while managing the users, groups and roles in Crowd. You can use Crowd's flexible and simple group management when the LDAP groups do not suit your requirements.

  For example, you can set up a simple group configuration in Crowd for use with Confluence and other Atlassian products, while authenticating your users against the corporate LDAP directory. You can also avoid the performance issues which might result from downloading large numbers of groups from LDAP.

- LDAP Directory Connector
  Crowd provides built-in connectors for the most popular LDAP directory servers, including Microsoft Active Directory, Sun DSEE, OpenLDAP, Apache DS, and others.
- Custom Directory Connector
  Custom directory connectors allow developers to connect Crowd to custom user-stores, such as existing databases or legacy systems.

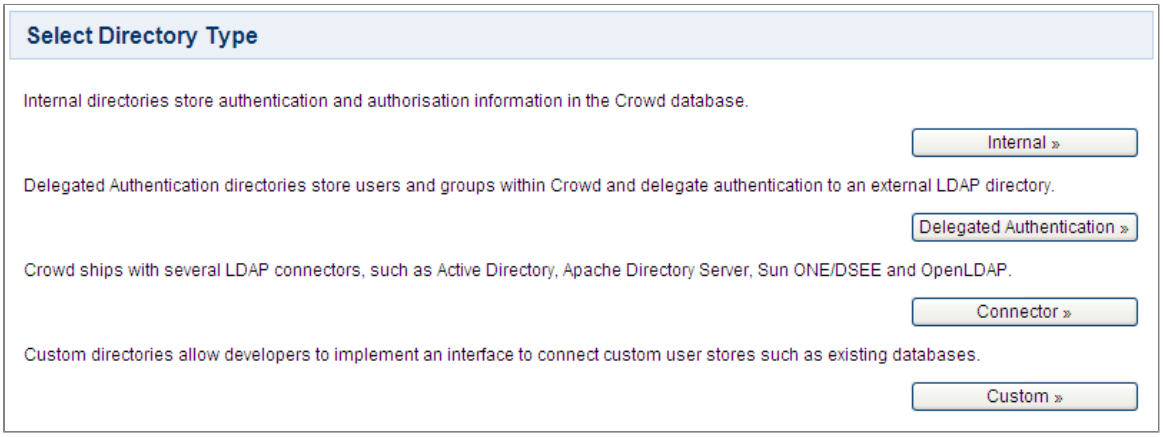
You can add as many directories of each type as you need.

To add a directory,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' link in the top navigation bar.
3. This will display the Directory Browser. Click the '**Add Directory**' link.
4. This will display the '**Select Directory Type**' screen (see below). Click the button corresponding to the type of directory you want to add:
   - '**Internal**' — see Configuring an Internal Directory
   - '**Delegated Authentication**' — see Configuring a Delegated Authentication Directory
   - '**Connector**' — see Configuring an LDAP Directory Connector (e.g. Microsoft Active Directory)
   - '**Custom**' — see Configuring a Custom Directory Connector

Once a directory has been configured, you will need to specify permissions for its users. You can then map the directory to appropriate applications.

Screenshot: 'Select Directory Type'



Related Topics

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP

        - Sun Directory Server Enterprise Edition (DSEE)
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
    - Importing Users from Atlassian Confluence
    - Importing Users from Atlassian JIRA
    - Importing Users from Atlassian Bamboo
    - Importing Users from Jive Forums
    - Importing Users from CSV Files
        - Configuring the CSV Importer
        - Mapping CSV Fields to Crowd Fields
        - Confirming the CSV Importer Configuration
        - Viewing the Results of the Import
    - Importing Users from One Crowd Directory into Another

Crowd Documentation

Configuring an Internal Directory

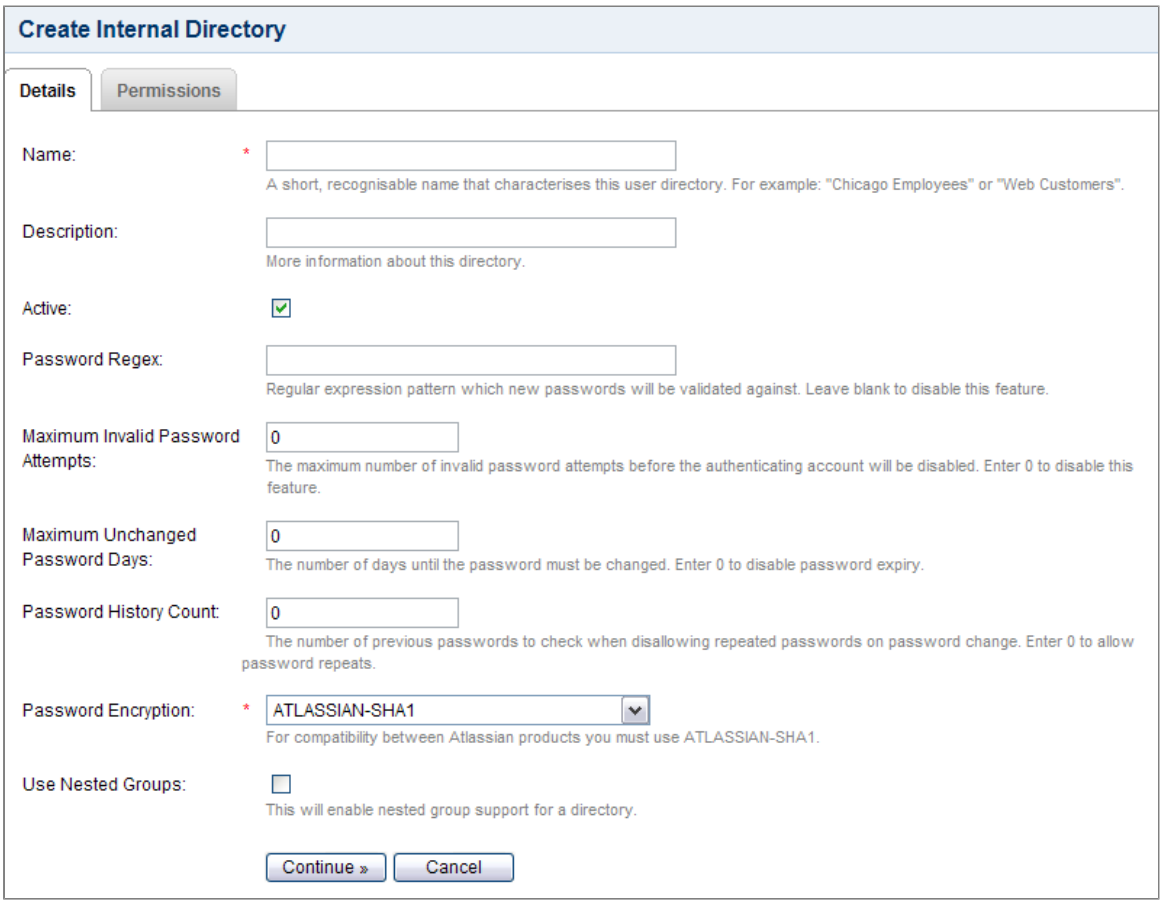
Internal directories use the Crowd database to store user, group and role information. Internal directories are stored in Crowd's database server.

To configure an internal directory,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' tab in the top navigation bar.
3. This will display the Directory Browser. Click '**Add Directory**' in the left-hand menu.
4. Click the '**Internal**' button.
5. Complete the fields as described in the table below.
6. Click the '**Continue**' button to configure the directory's permissions.

Once you have configured the directory's permissions, you will have finished configuring your new directory. You can then map the directory to appropriate applications.

Screenshot: Create internal directory



Internal Directory Attributes	Description
Name	The name used to identify the directory within Crowd. This is useful when there are multiple directories configured, e.g. Chicago Employees or Web Customers.
Description	Details about this specific directory.
Active	Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications.
Password Regex	Regex pattern which new passwords will be validated against. The regular expression format used is the java.util.regex.Pattern. For example, for an alphanumeric password of at least 8 characters, you could use the pattern: \[A-Za-z0-9\]{8,} Leave blank to disable this feature.
Maximum Invalid Password Attempts	The maximum number of invalid password attempts before the authenticating account will be disabled. Enter 0 to disable this feature.
Maximum Unchanged Password Days	The number of days until the password must be changed. This value is in days, enter 0 to disable this feature.
Password History Count	The number of previous passwords to prevent the user from using. Enter 0 to disable this feature.
Password Encryption	If you wish to import users into this directory from another Atlassian product, specify '**ATLASSIAN-SHA1**' in order to ensure password compatibility.
Use Nested Groups	Enable or disable support for nested groups on the internal user directory.

Next Step

See Specifying Directory Permissions.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Crowd Documentation

Configuring an LDAP Directory Connector

Crowd provides built-in connectors for the most popular LDAP directory servers, including Microsoft Active Directory, Sun DSEE, OpenLDAP, Apache DS, and others. The full list is below.

On this page:

- Summary of Configuration Steps
- Configuring Directory Details
- Configuring Connector Details
- Configuring LDAP Object and Attribute Settings
  - User Configuration
  - Group Configuration
  - Role Configuration
- LDAP Object Structures
- Hint: An LDAP Browser
- Next Step

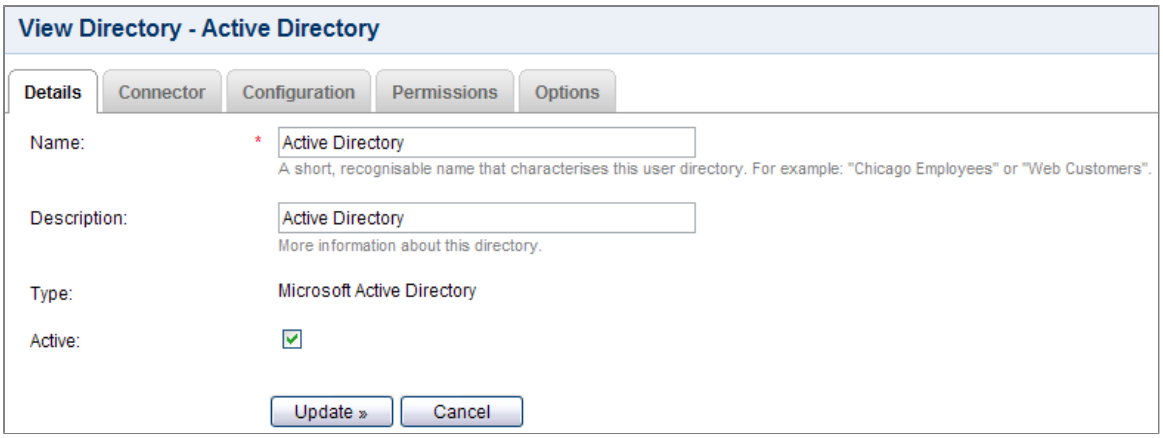
Summary of Configuration Steps

To configure an LDAP directory connector,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' link in the top navigation bar.
3. The Directory Browser will appear. Click the '**Add Directory**' link.
4. The 'Select Directory Type' screen will appear. Click the '**Connector**' button.
5. The '**Details**' tab will appear. See screenshot 1 below. Enter the '**Name**' and '**Description**' (see table of fields below), then click the '**Continue**' button.
6. The '**Connector**' tab will appear. See screenshot 2 below. Select the relevant connector type and fill in the basic connection information for your directory server. For details, please see:
   - Apache Directory Server (ApacheDS)
   - Apple Open Directory
   - Fedora Directory Server
   - Generic LDAP Directories
   - Microsoft Active Directory
   - Novell eDirectory
   - OpenDS
   - OpenLDAP
   - OpenLDAP Using Posix Schema
   - Posix Schema for LDAP
   - Sun Directory Server Enterprise Edition (DSEE)
7. Click the '**Test Connection**' button to verify that Crowd can successfully connect to the directory.
8. Click the '**Continue**' button.
9. The '**Configuration**' tab will appear. See the configuration screenshots below. Fill in the configuration details for your groups, roles and users, as described in the tables below the configuration screenshots. Also please see LDAP Object Structures (below).
10. Click the '**Test Search**' button to verify that Crowd can successfully locate groups/roles/users within the directory.
11. Click the '**Continue**' button to configure the directory's permissions.

Configuring Directory Details

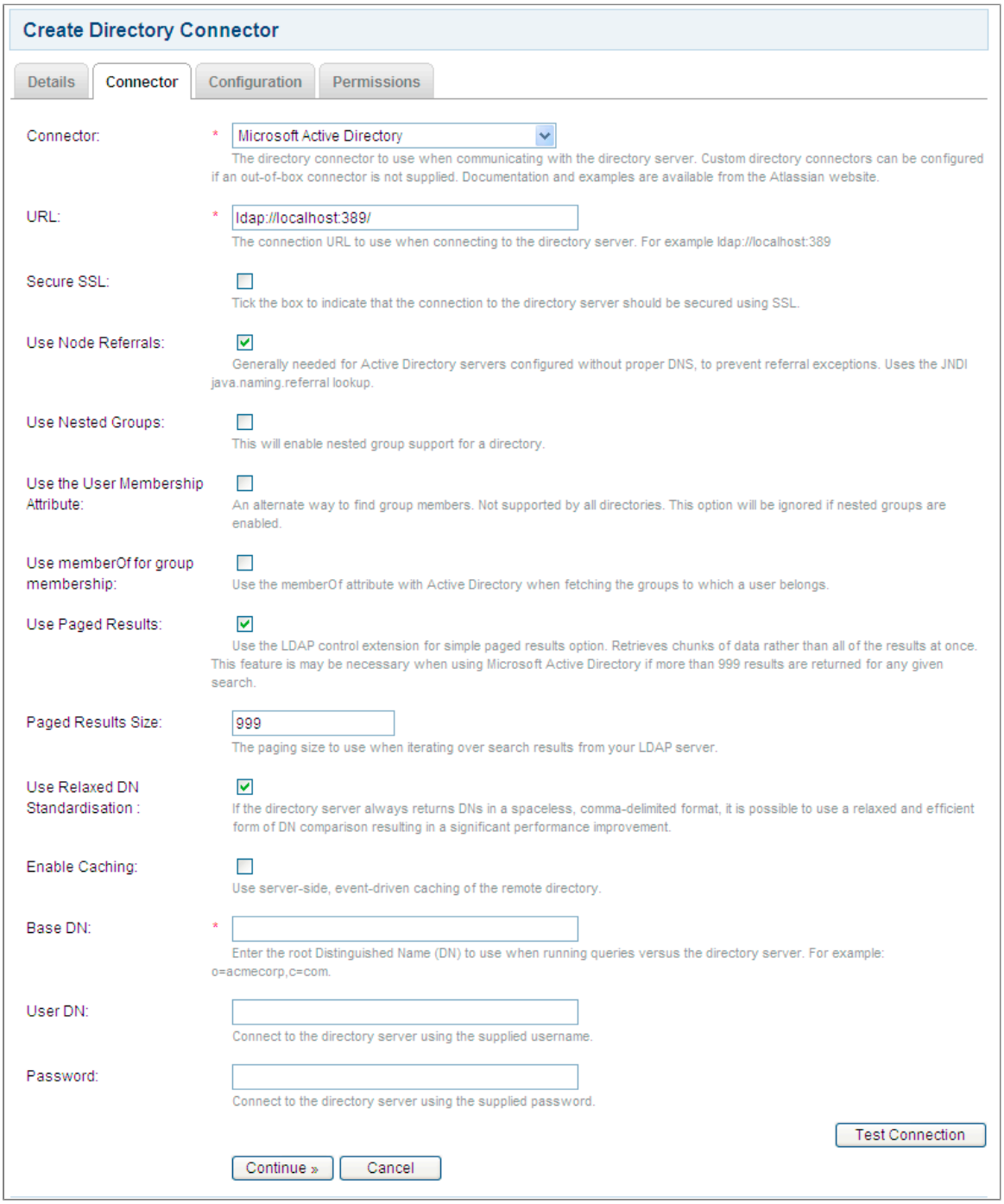
Screenshot 1: Directory details



Attribute	Description
Name	The name used to identify the directory within Crowd. This is useful when there are multiple directories configured, e.g. 'Chicago Employees' or 'Web Customers'.
Description	Details about this specific directory.
Active	Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications.

Configuring Connector Details

Screenshot 2: Connector details



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or port `636` for SSL.
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.

Use the User Membership Attribute	Put a tick in the checkbox if your Active Directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) • If this checkbox is ticked, Crowd will use the group membership attribute on the user when **retrieving the members of a given group**. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use 'memberOf' for Group Membership	Put a tick in the checkbox if your Active Directory supports the 'memberOf' attribute on the user. • If this checkbox is ticked, Crowd will use the 'memberOf' attribute when **retrieving the list of groups to which a given user belongs**. This will result in a more efficient search. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Use the LDAP control extension for simple paging of search results. Retrieves chunks of data rather than all of the search results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Paged Results Size	Enter the desired page size i.e. the maximum number of search results to be returned per page, when paged results are enabled. Defaults to 999 results.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for Active Directory, because Active Directory guarantees the format of DNs. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Enable Caching	Put a tick in the checkbox to enable directory caching. Directory caching can provide fast recurrent access to user, group and role data for a particular directory. This can provide significant performance improvements for applications such as JIRA, which require large amounts of user information. Please read the full instructions: Configuring Caching for an LDAP Directory.
Max Cache Elements in Memory	This checkbox appears if 'Enable Caching' is ticked. Specify the maximum number of cache elements to be held in memory before overflowing to disk. Please read the full instructions: Configuring Caching for an LDAP Directory.
Polling Interval	This checkbox appears if 'Enable Caching' is ticked. Crowd will send a request to Active Directory every x seconds, where 'x' is the number specified here. Please read the full instructions: Configuring Caching for an LDAP Directory.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

We have shown the settings for Active Directory. For details about the settings for your specific directory server, please see:

- Apache Directory Server (ApacheDS)
- Apple Open Directory
- Fedora Directory Server
- Generic LDAP Directories
- Microsoft Active Directory
- Novell eDirectory
- OpenDS
- OpenLDAP
- OpenLDAP Using Posix Schema
- Posix Schema for LDAP
- Sun Directory Server Enterprise Edition (DSEE)

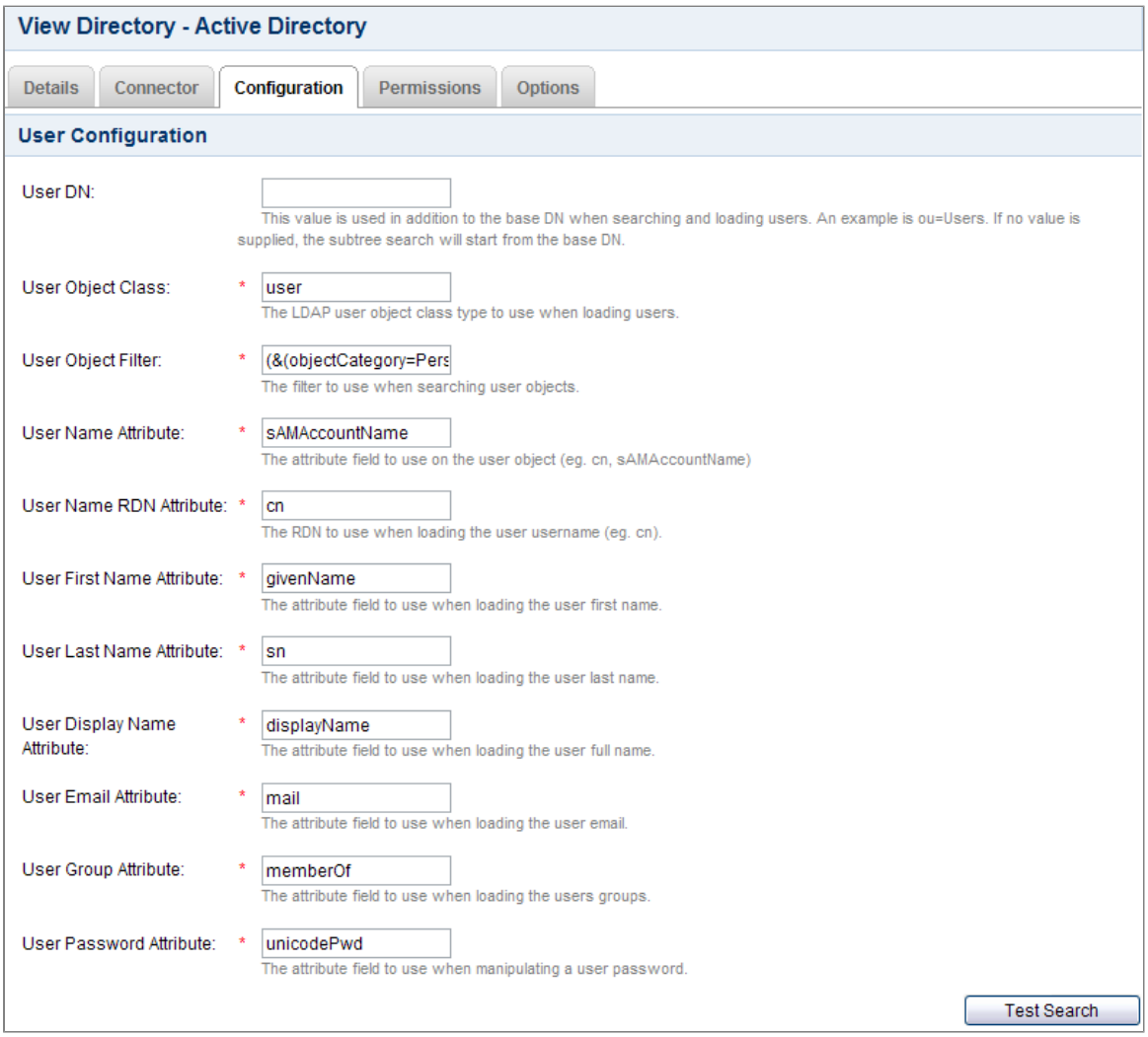
Configuring LDAP Object and Attribute Settings

Once you have selected a connector you can modify various LDAP object and attribute settings of the specific LDAP server for users, groups and roles as shown on the screenshots below. On first setup, Crowd will provide generic default settings based on the connector selected.

When configuring your LDAP connector, if you are using non-standard object types, you will need to adjust the default filter and object type configurations. If your connector is added successfully, but you are unable to see any data when browsing your LDAP directory, it is likely that your object and filters are configured incorrectly.

User Configuration

Screenshot 3: User configuration

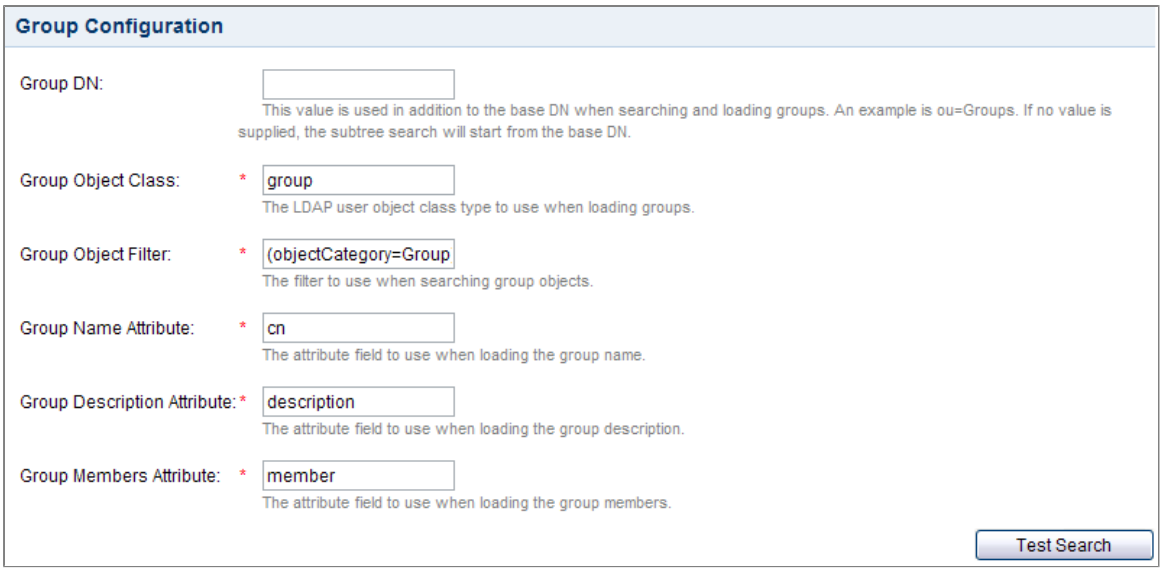


Attribute	Description
User DN	This value is used in addition to the base DN (distinguished name) when searching and loading users. An example is `ou=Users`. If no value is supplied, the subtree search will start from the base DN.
User Object Class	This is the name of the class used for the LDAP user object. An example is `user`.
User Object Filter	The filter to use when searching user objects.
User Name Attribute	The attribute field to use when loading the username. Examples are `cn` and `sAMAccountName`.
User Name RDN Attribute	The RDN (relative distinguished name) to use when loading the username. An example is `cn`. The DN for each LDAP entry is composed of two parts: the RDN and the location within the LDAP directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure.
User First Name Attribute	The attribute field to use when loading the user's first name. An example is `givenName`.
User Last Name Attribute	The attribute field to use when loading the user's last name. An example is `sn`.

User Display Name Attribute	The attribute field to use when loading the user's full name. An example is `displayName`.
User Email Attribute	The attribute field to use when loading the user's email address. An example is `mail`.
User Group Attribute	The attribute field to use when loading the user's groups. An example is `memberOf`. Please refer to the specific settings for group membership searches on the 'Connector' tab, as described above.
User Password Attribute	The attribute field to use when loading a user's password. An example is `unicodePwd`.

Group Configuration

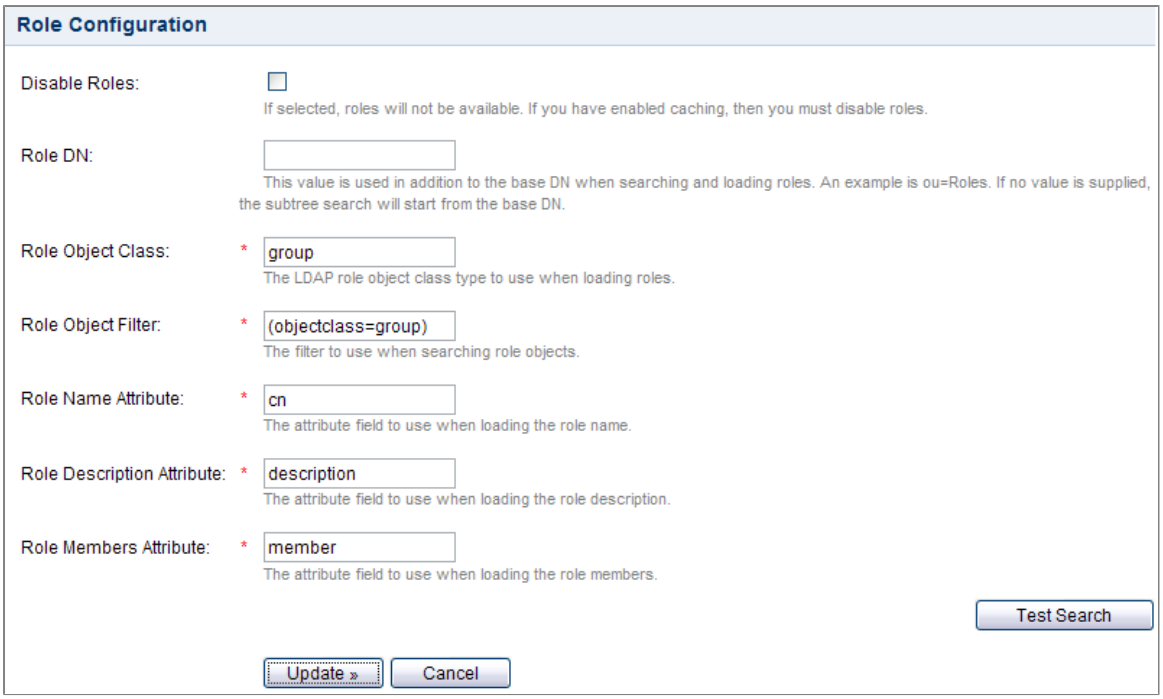
Screenshot 4: Group configuration



Attribute	Description
Group DN	This value is used in addition to the base DN when searching and loading groups, an example is ou=Groups. If no value is supplied, the subtree search will start from the base DN.
Group Object Class	This is the name of the class used for the LDAP group object. Examples are `groupOfUniqueNames` and `group`.
Group Object Filter	The filter to use when searching group objects. An example is `(objectCategory=Group)`.
Group Name Attribute	The attribute field to use when loading the group's name. An example is `cn`.
Group Description Attribute	The attribute field to use when loading the group's description. An example is `description`.
Group Members Attribute	The attribute field to use when loading the group's members. An example is `member`. Please refer to the specific settings for group membership searches on the 'Connector' tab, as described above.

Role Configuration

Screenshot 5: Role configuration



Attribute	Description
Disable Roles	When you create an LDAP directory connector, roles in Crowd will be disabled by default. To enable roles, remove the tick from the checkbox. You may need to click out of the checkbox (e.g. click the '**Update**' button) to see the role configuration fields. Then click the '**Update**' button again to apply the change. **Note:** If you have enabled caching for the directory, then you cannot enable roles for that directory. Caching does not work with roles in Crowd. **Crowd's role-based access control could be enhanced** At present, the implementation of roles in Crowd is identical to the implementation of groups. Additional development work would be needed to differentiate the functionality of roles from groups. If you would like to help us to design better role-based access control, please add a comment to the improvement request CWD-931, letting us know how you would like to see it work.
Role DN	This value is used in addition to the base DN when searching and loading roles. An example is `ou=Roles`. If no value is supplied, the subtree search will start from the base DN.
Role Object Class	This is the name of the class used for the LDAP role object. An example is `group`.
Role Object Filter	The filter to use when searching role objects. An example is `(objectclass=group)`.
Role Name Attribute	The attribute field to use when loading the role's name. An example is `cn`.
Role Description Attribute	The attribute field to use when loading the role's description. An example is `description`.
Role Members Attribute	The attribute field to use when loading the role's members. An example is `member`.

LDAP Object Structures

The Crowd LDAP connectors assume that all container objects (groups and roles) have the **full** DN to the associated member. Currently, the membership attributes on a User object are not used by Crowd; however, in the future these associations may be used to assist with performance when looking up memberships.

Supported Object Types

- groupOfUniqueNames
- inetorgperson

- posixGroup
- posixUser

Zimbra Mail Server
User objects have been tested and are known to work with the `zimbraAccount` LDAP object types.

Microsoft Active Directory
The Active Directory LDAP connector assumes that all LDAP object types are of the default structure. Any changes to the default object structure of the `User` and `Group` objects will require a custom connector to be coded.

Supported Attributes

Crowd's LDAP connectors support the adding and updating of the following user attributes when integrating with an LDAP server via an LDAP directory connector:

- surname
- given name
- email
- password

If you need support for additional LDAP attributes, the Crowd LDAP connector can be extended. With a license purchase, full source is available and the LDAP connectors can be modified to support any number of attributes.

Hint: An LDAP Browser

To help you identify your LDAP structure, you may find an LDAP browser useful. Take a look at our guide on using Apache Directory Studio.

Next Step

Specify the directory permissions, which allow you to restrict the way in which applications can use the directories. See Specifying Directory Permissions.

Once you have configured the directory's permissions, you have finished configuring your new directory. You can then map the directory to appropriate applications.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
    - Configuring an Internal Directory
    - Configuring an LDAP Directory Connector
        - Apache Directory Server (ApacheDS)
        - Apple Open Directory
        - Fedora Directory Server
        - Generic LDAP Directories
        - Microsoft Active Directory
            - Configuring an SSL Certificate for Microsoft Active Directory
        - Novell eDirectory
        - OpenDS
        - OpenLDAP
        - OpenLDAP Using Posix Schema
        - Posix Schema for LDAP
        - Sun Directory Server Enterprise Edition (DSEE)
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
    - Importing Users from Atlassian Confluence
    - Importing Users from Atlassian JIRA
    - Importing Users from Atlassian Bamboo
    - Importing Users from Jive Forums
    - Importing Users from CSV Files
        - Configuring the CSV Importer
        - Mapping CSV Fields to Crowd Fields
        - Confirming the CSV Importer Configuration
        - Viewing the Results of the Import
    - Importing Users from One Crowd Directory into Another

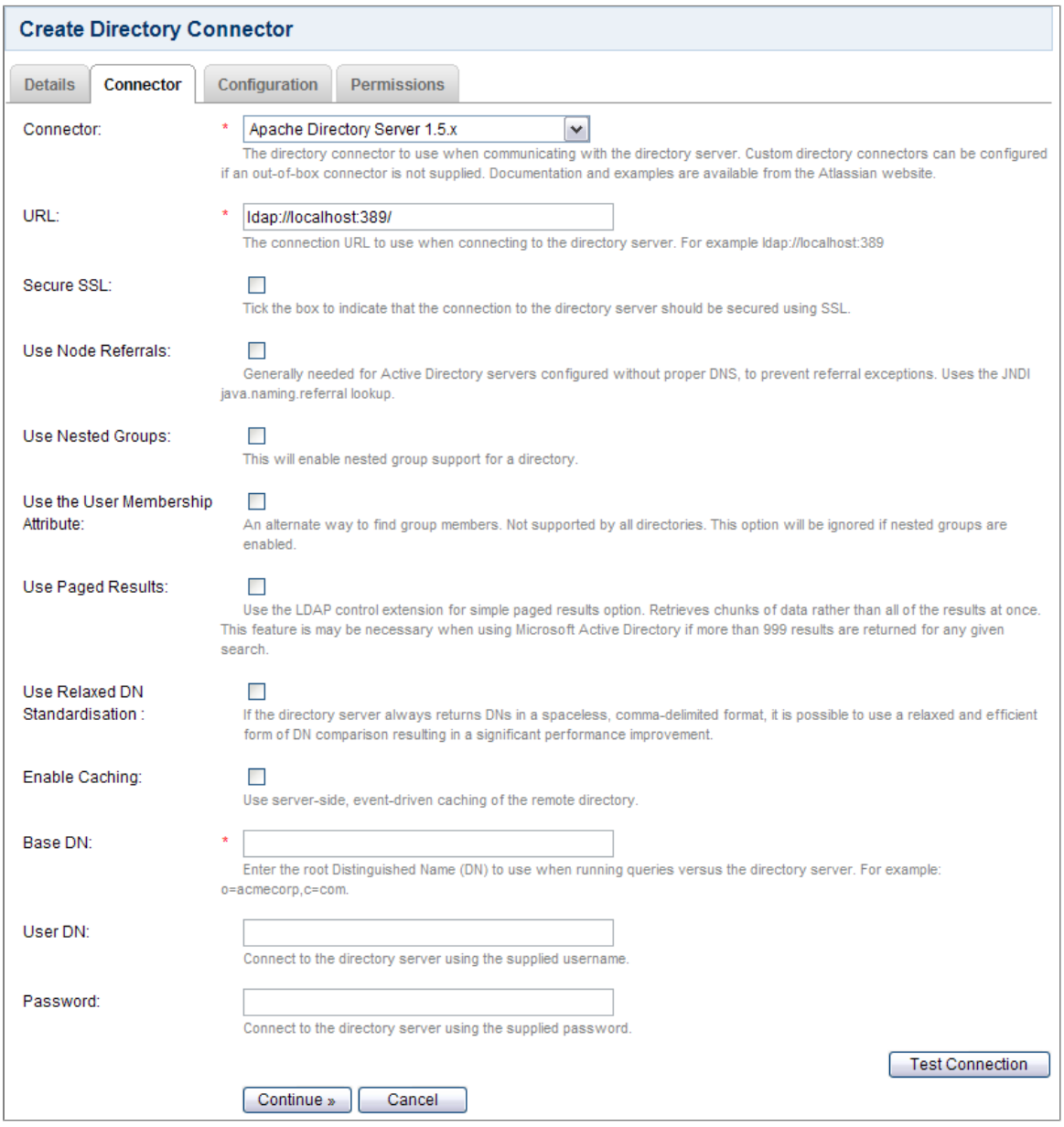
Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

Apache Directory Server (ApacheDS)

This page provides configuration notes for Apache Directory Server. This page is related to Configuring an LDAP Directory Connector.

Known issues with ApacheDS and Crowd:
1. ApacheDS 1.0.2 does not support password resets without a restart. This is an ApacheDS limitation.
2. ApacheDS does not support paged results. CWD-1109: Cannot browse users or groups if Use Paged Results is enabled . Again, this is an ApacheDS limitation.

Screenshot: Connector — ApacheDS



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or `port 639` for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.

Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use Paged Results	Use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version. This is the **default and recommended setting** for ApacheDS. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison, which will result in a significant performance improvement. This is only possible if the directory guarantees the format of DNs.
Enable Caching	This option is available for **ApacheDS 1.5.0 and later**. Put a tick in the checkbox to enable directory caching. Directory caching can provide fast recurrent access to user, group and role data for a particular directory. This can provide significant performance improvements for applications such as JIRA, which require large amounts of user information. Please read the instructions: Configuring Caching for an LDAP Directory.
Max Cache Elements in Memory	This checkbox appears if 'Enable Caching' is ticked. Specify the maximum number of cache elements to be held in memory before overflowing to disk. Please read the full instructions: Configuring Caching for an LDAP Directory.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	The username that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Configuration details for ApacheDS

OpenLDAP Directory Example	Value
Base DN	dc=example,dc=com

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files

- Configuring the CSV Importer
- Mapping CSV Fields to Crowd Fields
- Confirming the CSV Importer Configuration
- Viewing the Results of the Import
- Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

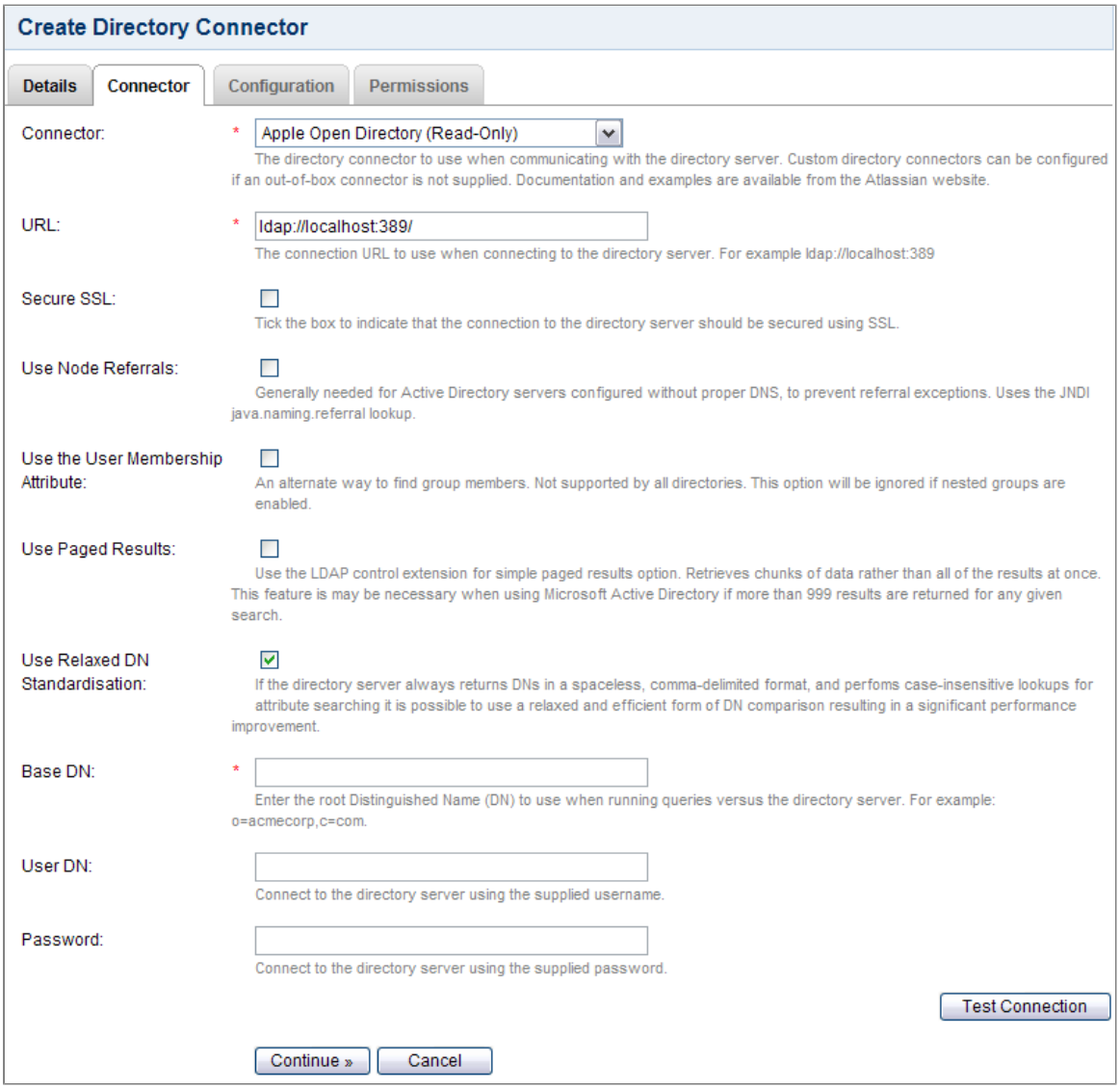
Apple Open Directory

This page provides configuration notes for Apple OS X Open Directory. This page is related to Configuring an LDAP Directory Connector.

Crowd supports read-only connections to Apple OS X Open Directory services.

> **Crowd's Apple Open Directory support is read-only**
> Currently, you cannot add or update user details or group details in a Crowd-connected Apple OS X Open Directory server. Users will not be able to change their passwords from Crowd or from Crowd-connected applications.

Screenshot: Connector — Apple OS X Open Directory



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g. `ldap://localhost:389`, or port `639` for SSL.
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.

Use Node Referrals	Specifies whether to use the JNDI lookup `java.naming.referral` option.
Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Specifies whether to use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for Apple Open Directory. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Base DN	The root distinguished name to use when running queries against the directory server, e.g. `o=acmecorp,c=com`.
User DN	The distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Group Relationships

Crowd will check both the `gidNumber` and the `memberUid` attributes to determine if a user is a member of a group. The name of the `gidNumber` attribute is not configurable — Crowd will always use this attribute to determine membership.

The RFC 2307 schema does not support nesting of groups, so Crowd does not support nested groups in Apple Open Directory.

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

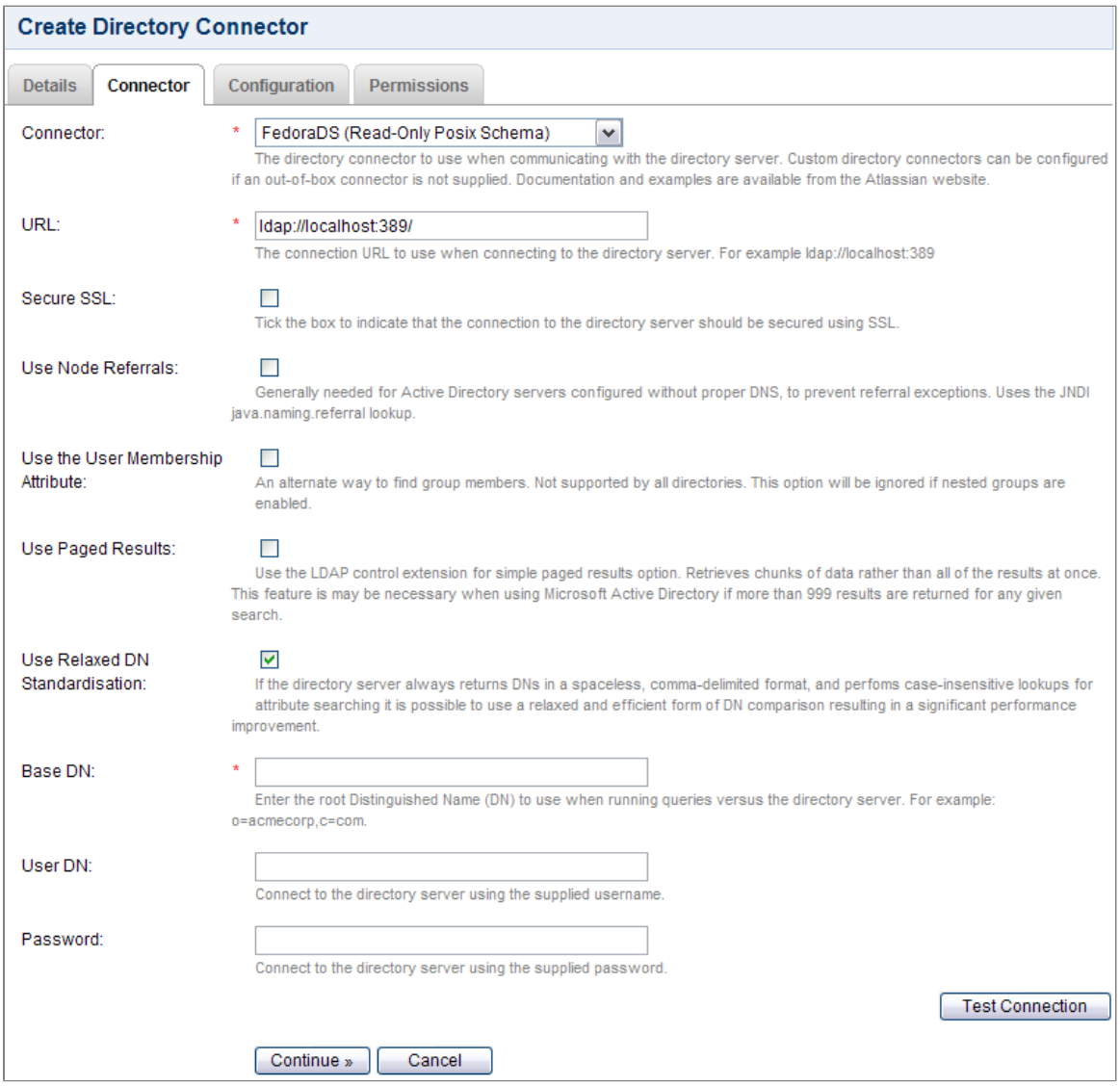
Fedora Directory Server

This page provides configuration notes for Fedora Directory Server (Fedora DS). This page is related to Configuring an LDAP Directory Connector.

Crowd supports read-only connections to Fedora DS using the Posix/NIS schema RFC 2307.

> **Crowd's Fedora DS support is read-only**
> Currently, you cannot add or update user details or group details in a Crowd-connected Fedora Directory server. Users will not be able to change their passwords from Crowd or from Crowd-connected applications.

Screenshot: Connector — Fedora DS



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or port 639 for SSL.
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.
Use Node Referrals	Specifies whether to use the JNDI lookup `java.naming.referral` option.

Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Specifies whether to use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for Fedora DS. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Base DN	The root distinguished name to use when running queries against the directory server, e.g.: `o=acmecorp,c=com`.
User DN	The distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Group Relationships

Crowd will check both the `gidNumber` and the `memberUid` attributes to determine if a user is a member of a group. The name of the `gidNumber` attribute is not configurable — Crowd will always use this attribute to determine membership.

The RFC 2307 schema does not support nesting of groups, so Crowd does not support nested groups in Fedora DS.

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

Generic LDAP Directories

This page provides configuration notes for generic LDAP directories. This page is related to Configuring an LDAP Directory Connector.

Screenshot: Connector — Generic Directory Server

Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or `port 639` for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.

Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use Paged Results	Use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version. This is the default setting for generic LDAP directories. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison, which will result in a significant performance improvement. This is only possible if the directory guarantees the format of DNs.
Password Encryption	Select the type of encryption that the directory uses.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	The username that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

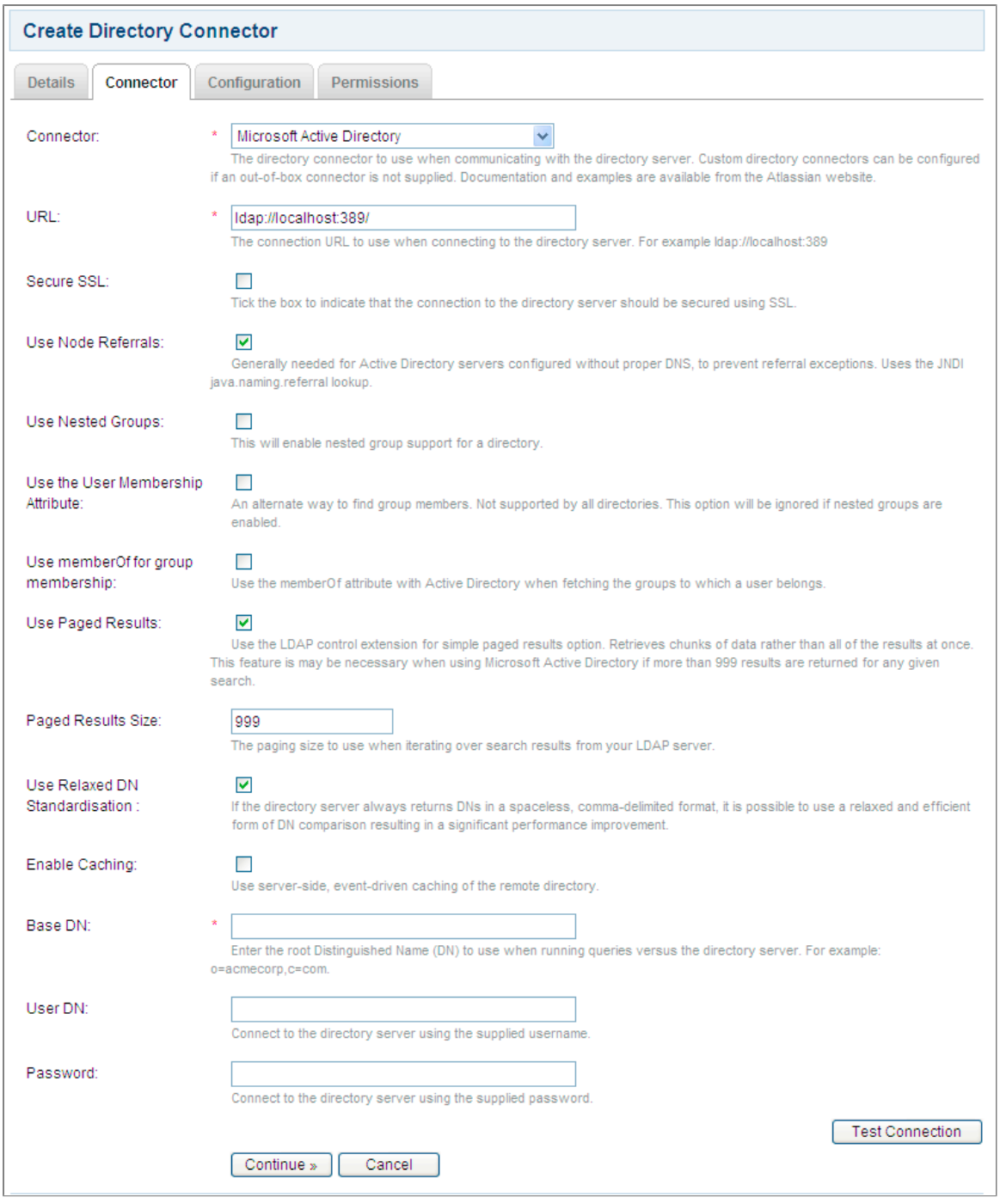
- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

Microsoft Active Directory

This page provides configuration notes for Microsoft Active Directory. This page is related to Configuring an LDAP Directory Connector.

Screenshot: Connector — Microsoft Active Directory



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or port 636 for SSL.
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.

Use the User Membership Attribute	Put a tick in the checkbox if your Active Directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) • If this checkbox is ticked, Crowd will use the group membership attribute on the user when **retrieving the members of a given group**. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use 'memberOf' for Group Membership	Put a tick in the checkbox if your Active Directory supports the 'memberOf' attribute on the user. • If this checkbox is ticked, Crowd will use the 'memberOf' attribute when **retrieving the list of groups to which a given user belongs**. This will result in a more efficient search. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Use the LDAP control extension for simple paging of search results. Retrieves chunks of data rather than all of the search results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Paged Results Size	Enter the desired page size i.e. the maximum number of search results to be returned per page, when paged results are enabled. Defaults to 999 results.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for Active Directory, because Active Directory guarantees the format of DNs. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Enable Caching	Put a tick in the checkbox to enable directory caching. Directory caching can provide fast recurrent access to user, group and role data for a particular directory. This can provide significant performance improvements for applications such as JIRA, which require large amounts of user information. Please read the full instructions: Configuring Caching for an LDAP Directory.
Max Cache Elements in Memory	This checkbox appears if 'Enable Caching' is ticked. Specify the maximum number of cache elements to be held in memory before overflowing to disk. Please read the full instructions: Configuring Caching for an LDAP Directory.
Polling Interval	This checkbox appears if 'Enable Caching' is ticked. Crowd will send a request to Active Directory every x seconds, where 'x' is the number specified here. Please read the full instructions: Configuring Caching for an LDAP Directory.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Configuration notes for Microsoft Active Directory

Active Directory Attribute	Value
URL	The URL for Microsoft Active Directory should be in the following format: `ldap://domainname`.
Base DN	`cn=users,dc=ad,dc=acmecorp,dc=com` For Microsoft Active Directory, specify the Base DN in the following format: `dc=domain1,dc=local`. You will need to replace the `domain1` and `local` for your specific configuration. Microsoft Server provides a tool called `ldp.exe` which is useful for finding out and configuring the the LDAP structure of your server.
User DN	`administrator@ad.acmecorp.com`

Configuring an SSL Certificate for Microsoft Active Directory
If you wish to use Crowd to add users or change passwords in Microsoft Active Directory, you will need to install an SSL certificated generated by your Active Directory server and then install the certificate into your JVM keystore. Please read the instructions: Configuring an SSL Certificate for Microsoft Active Directory.

Integrating Crowd with ADAM
We have not tested Crowd integration with Active Directory Application Mode (ADAM). However, ADAM and Active Directory share the same code base, LDAP interface and API. So ADAM should work with Crowd, following the same integration instructions as above. If you try it, we'd be interested to hear of your experiences.

Next Step

Go back to Configuring an LDAP Directory Connector

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

Configuring an SSL Certificate for Microsoft Active Directory

You can configure Crowd to work with Microsoft Active Directory by setting up an LDAP connector in Crowd. If you wish to use Crowd to add users or change passwords in Active Directory, you will need to install an SSL certificate generated by your Active Directory server and then install the certificate into your JVM keystore.

On this page:

- Prerequisites
- Step 1. Install the Microsoft Certificate Services
- Step 2. Obtain the Server Certificate
- Step 3. Import the Server Certificate
  - Windows
  - Unix
  - Mac OS X

Prerequisites

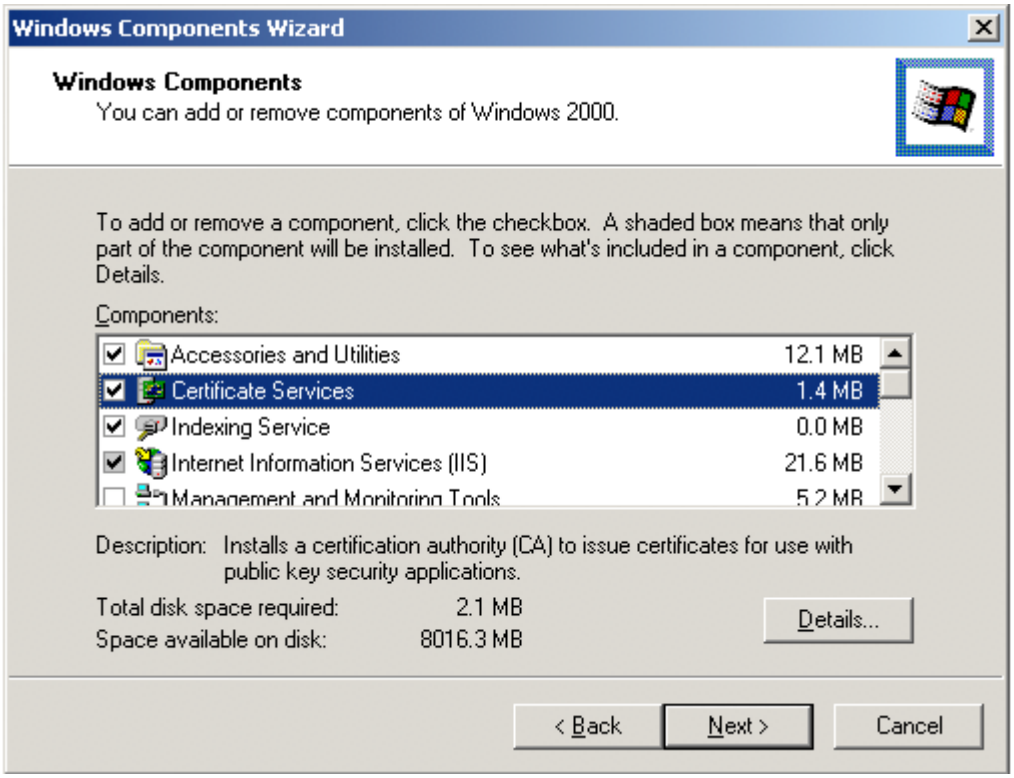
Make sure that you have the following installed on your Windows server (domain controller):

Required Component	Description
Windows 2000 Service Pack 2	Required if you are using Windows 2000
Internet Information Services (IIS)	This is required before you can install Windows Certificate Services.

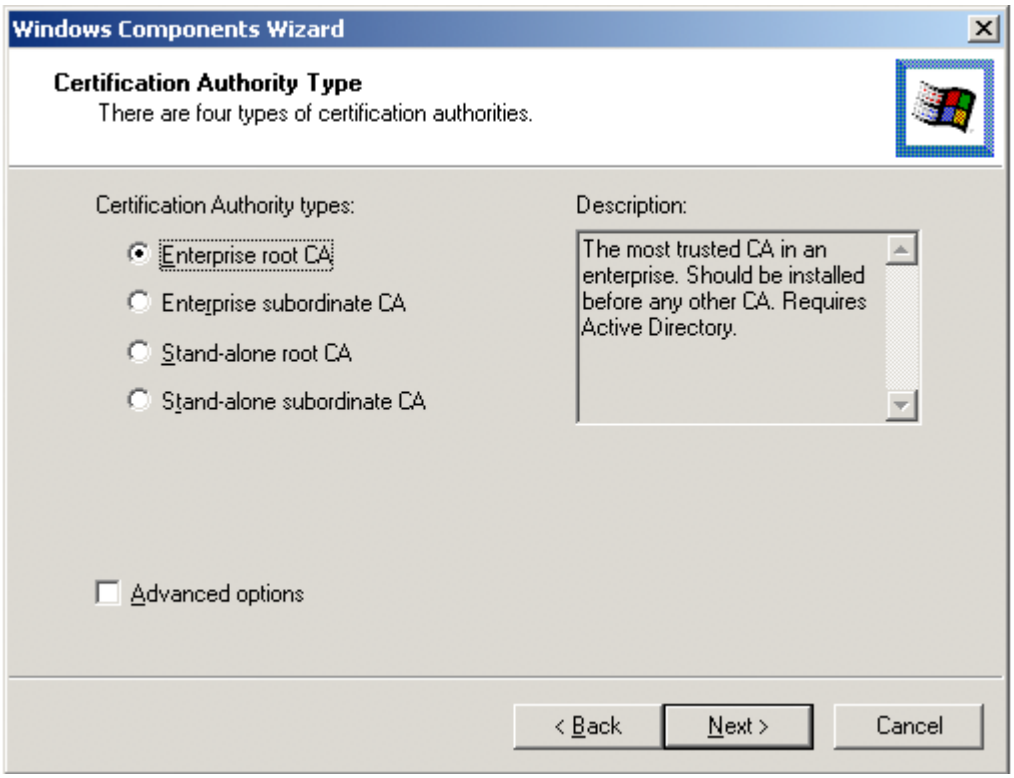
Windows Certificate Services	This installs a certification authority (CA) which is used to issue certificates.
Windows 2000 High Encryption Pack (128-bit)	Required if you are using Windows 2000. Provides the highest available encryption level (128-bit).

Step 1. Install the Microsoft Certificate Services

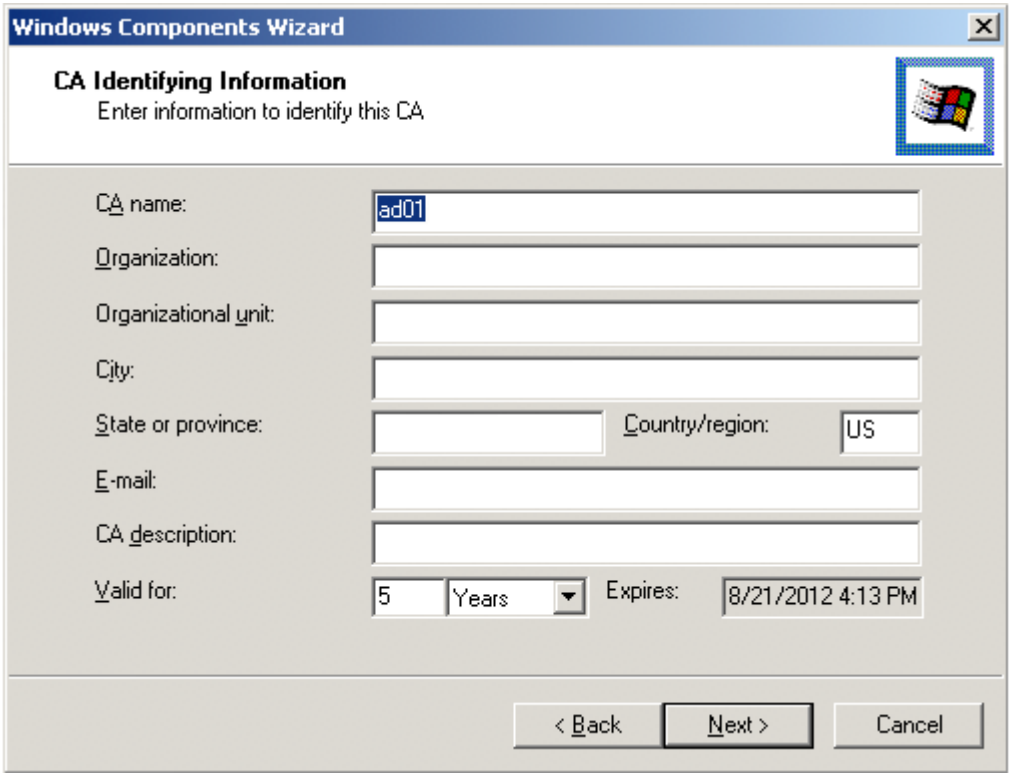
1. Using the Active Directory **Control Panel – Add/Remove Programs** administration tool:
   - Select '**Add/Remove Windows Components**' to start the **Windows Components Wizard**.
   - Place check marks next to '**Certificate Services**' and '**Internet Information Services (IIS)**'.
   - Click '**Next>**'.



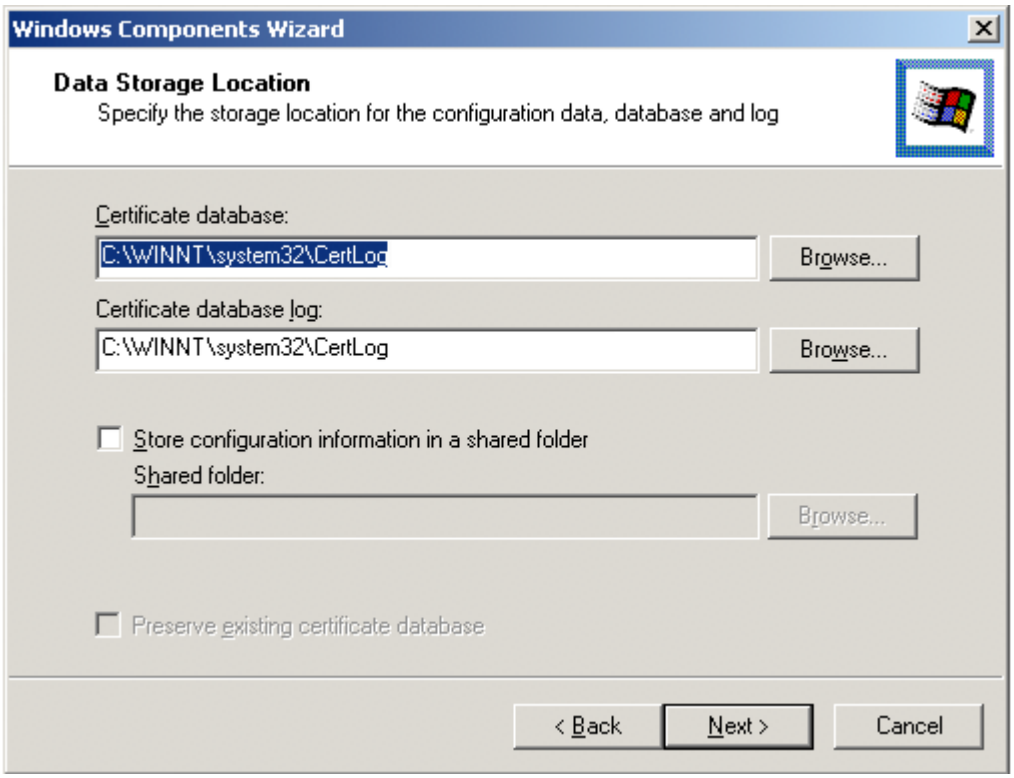
2. Select '**Enterprise root CA**' Certificate Authority Type and click '**Next>**'.



3. Enter a '**CA name**' (server name) and click '**Next>**'. On Windows Server 2003, this is the '**Common name for this CA**'.

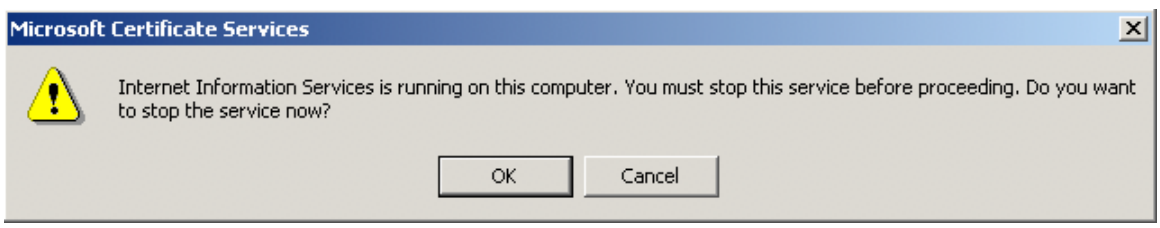


4. Leave the '**Data Storage Locations**' as default and click '**Next>**'.



5. The software installation process is complete. Click '**Finish**'.

6. Click '**OK**' to restart IIS.



7. You will now need to restart your Microsoft Active Directory Server.

Step 2. Obtain the Server Certificate

The steps above describe how to install the certification authority (CA) on your Microsoft Active Directory server. Next, you will need to add the Microsoft Active Directory server's SSL certificate to the list of accepted certificates used by the JDK that runs your Crowd server.

The Active Directory certificate is automatically generated and placed in root of the C:\ drive, matching a file format similar to the tree structure of your Active Directory server, e.g. `c:\crowd-ad2000.ad01.crowd.atlassian.com_ad01.crt`.

You can also export the certificate by executing this command on the Active Directory server:

```
1.certutil -ca.cert crowd-client.crt
```

Step 3. Import the Server Certificate

For a Crowd server to trust your directory's certificate, the certificate must be imported into your Java runtime environment. The JDK stores trusted certificates in a file called a keystore. The default keystore file is called `cacerts` and it lives in the `lib\security` sub-directory of your Java installation.

In the following examples, we use `server-certificate.crt` to represent the certificate file exported by your Directory Server. You will need to alter the instructions below to match the name actually generated.

Windows

1. Navigate to the directory in which Java is installed. It's probably called something like `C:\Program Files\Java\jdk1.5.0_12`.
2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

```
keytool -import -keystore .\lib\security\cacerts -file server-certificate.crt
```

3. `keytool` will prompt you for a password. The default keystore password is `changeit`.
4. When prompted `Trust this certificate? [no]:` enter `yes` to confirm the key import:

```
Enter keystore password:  changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 15563d6677a4e9e4582d8a84be683f9
Valid from: Tue Aug 21 01:10:46 ACT 2007 until: Tue Aug 21 01:13:59 ACT 2012
Certificate fingerprints:
         MD5:  D6:56:F0:23:16:E3:62:2C:6F:8A:0A:37:30:A1:84:BE
         SHA1: 73:73:4E:A6:A0:D1:4E:F4:F3:CD:CE:BE:96:80:35:D2:B4:7C:79:C1
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

You may now use the `Secure` SSL option when using Crowd to connect to your directory.

Unix

1. Navigate to the directory in which Java is installed. `cd $JAVA_HOME` will usually get you there.
2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

```
sudo keytool -import -keystore ./lib/security/cacerts -file server-certificate.crt
```

3. `keytool` will prompt you for a password. The default keystore password is `changeit`.
4. When prompted `Trust this certificate? [no]:` enter `yes` to confirm the key import:

```
Password:
Enter keystore password:  changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 15563d6677a4e9e4582d8a84be683f9
Valid from: Tue Aug 21 01:10:46 ACT 2007 until: Tue Aug 21 01:13:59 ACT 2012
Certificate fingerprints:
         MD5:  D6:56:F0:23:16:E3:62:2C:6F:8A:0A:37:30:A1:84:BE
         SHA1: 73:73:4E:A6:A0:D1:4E:F4:F3:CD:CE:BE:96:80:35:D2:B4:7C:79:C1
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

You may now use the `Secure` SSL option when using Crowd to connect to your directory.

Mac OS X

1. Navigate to the directory in which Java is installed. This is usually `/Library/Java/Home`.
2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

```
sudo keytool -import -keystore ./lib/security/cacerts -file server-certificate.crt
```

3. `keytool` will prompt you for a password. The default keystore password is `changeit`.
4. When prompted `Trust this certificate? [no]:` enter `yes` to confirm the key import:

```
Password:
Enter keystore password:  changeit
Owner: CN=ad01, C=US
Issuer: CN=ad01, C=US
Serial number: 15563d6677a4e9e4582d8a84be683f9
Valid from: Tue Aug 21 01:10:46 ACT 2007 until: Tue Aug 21 01:13:59 ACT 2012
Certificate fingerprints:
         MD5:  D6:56:F0:23:16:E3:62:2C:6F:8A:0A:37:30:A1:84:BE
         SHA1: 73:73:4E:A6:A0:D1:4E:F4:F3:CD:CE:BE:96:80:35:D2:B4:7C:79:C1
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

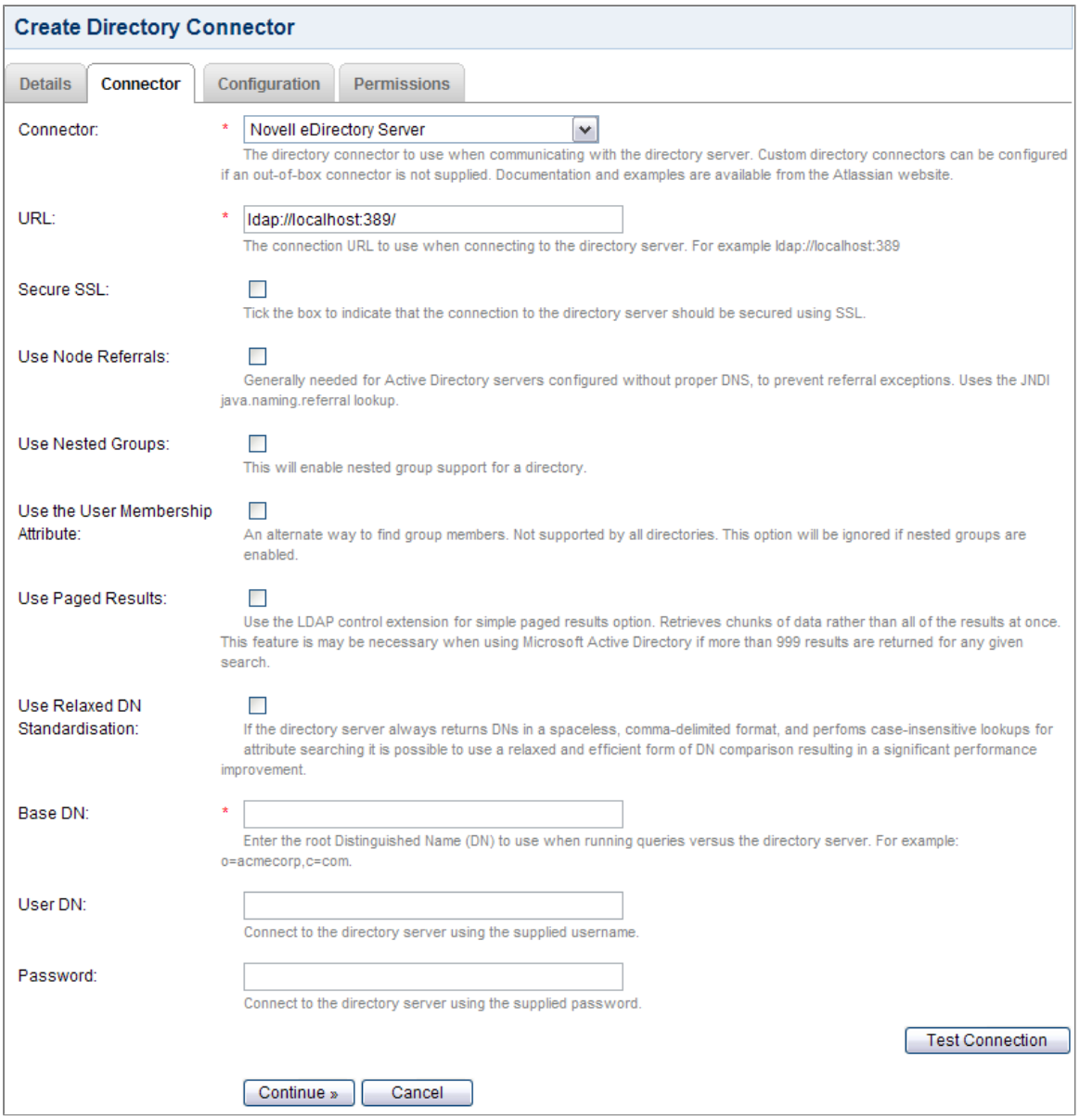
You may now use the `Secure` SSL option when using Crowd to connect to your directory.
RELATED TOPICS

Microsoft Active Directory
Configuring Crowd to Work with SSL

Novell eDirectory

This page provides configuration notes for Novell eDirectory. This page is related to Configuring an LDAP Directory Connector.

Screenshot: Connector — Novell eDirectory Server



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or port 636 for SSL.
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.
Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.

Use Paged Results	Use the LDAP control extension for simple paging of search results. Retrieves chunks of data rather than all of the search results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version. This is the default setting for Novell eDirectory. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison, which will result in a significant performance improvement. This is only possible if the directory guarantees the format of DNs.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Next Step

Go back to Configuring an LDAP Directory Connector

RELATED TOPICS

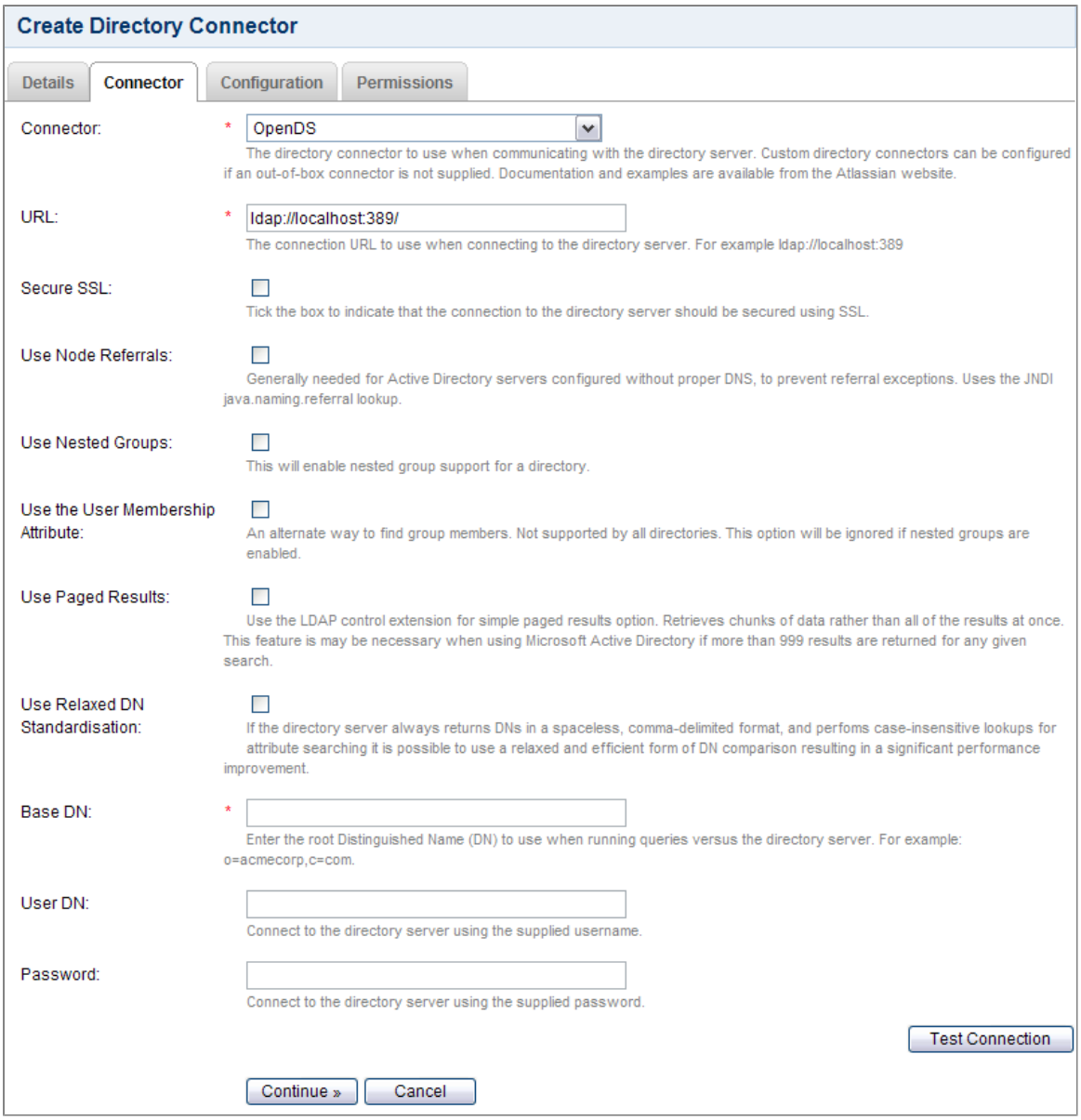
- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

OpenDS

This page provides configuration notes for OpenDS. This page is related to Configuring an LDAP Directory Connector.

Screenshot: Connector — OpenDS



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or `port 639` for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.
Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.

Use Paged Results	Use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version. This is the default setting for OpenDS. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison, which will result in a significant performance improvement. This is only possible if the directory guarantees the format of DNs.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Configuration Details for OpenDS

OpenDS Directory Attribute	Value
Base DN	dc=example,dc=com
User DN	cn=Manager,dc=example,dc=com

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

OpenLDAP

This page provides configuration notes for OpenLDAP. This page is related to Configuring an LDAP Directory Connector.

Screenshot: Connector — OpenLDAP

Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or `port 639` for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.
Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.

Use Paged Results	Use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for OpenLDAP. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Password Encryption	Select the type of encryption that the directory uses.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Configuration Details for OpenLDAP

OpenLDAP Directory Attribute	Value
Base DN	dc=example,dc=com
User DN	cn=Manager,dc=example,dc=com

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

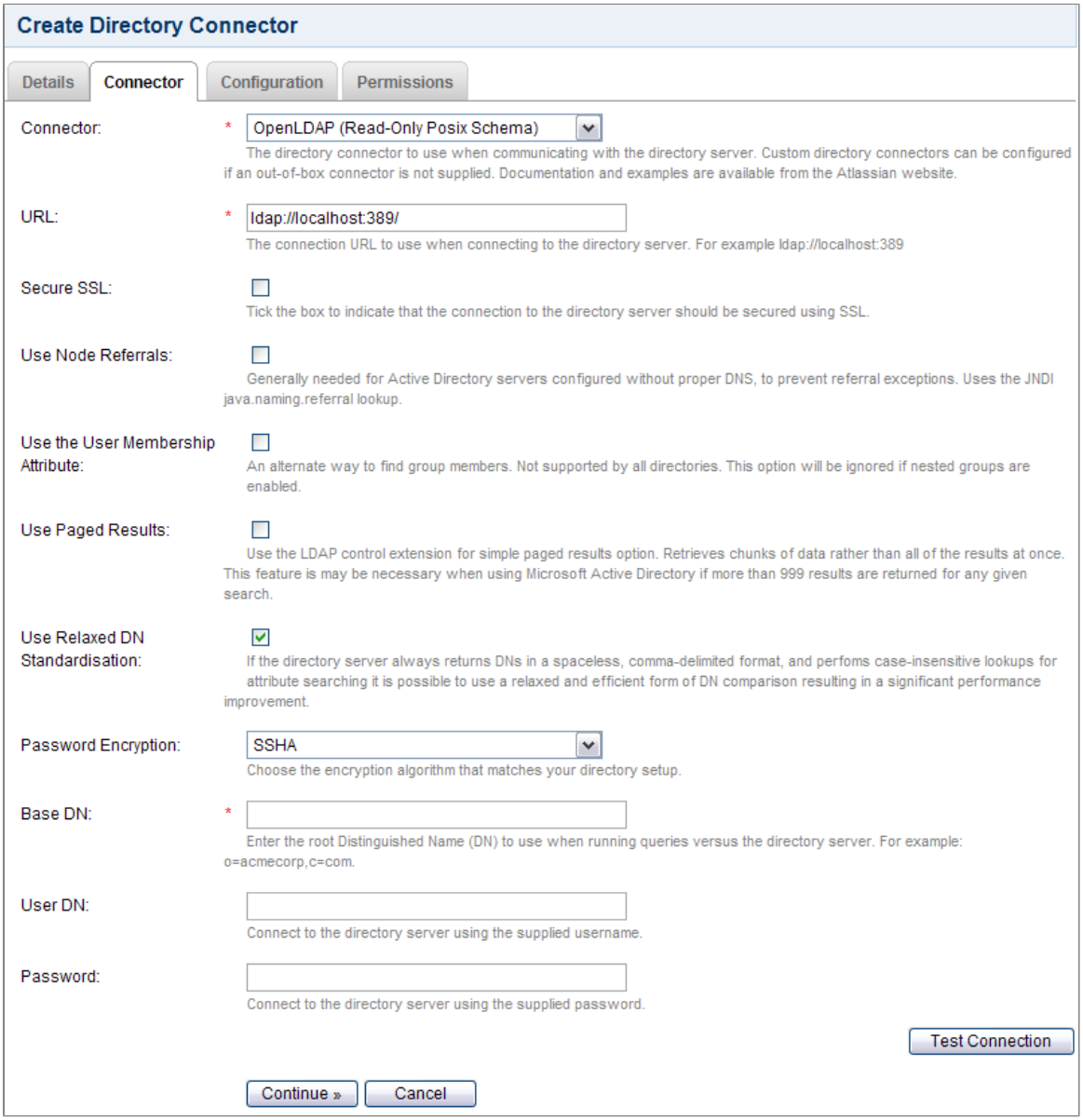
Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

OpenLDAP Using Posix Schema

This page provides configuration notes for an OpenLDAP directory using the Posix/NIS schema RFC 2307. This page is related to Configuring an LDAP Directory Connector.

Posix support is read-only
Currently, you cannot add or update user details or group details in a Crowd-connected OpenLDAP directory based on the Posix/NIS schema. Users will not be able to change their passwords from Crowd or from Crowd-connected applications.

Screenshot: Connector — OpenLDAP on a Posix schema



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or port `639` for SSL.
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.
Use Node Referrals	Specifies whether to use the JNDI lookup `java.naming.referral` option.

Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Specifies whether to use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for OpenLDAP Posix. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Password Encryption	Select the type of encryption that the directory uses.
Base DN	The root distinguished name to use when running queries against the directory server, e.g.: `o=acmecorp,c=com`.
User DN	The distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Group Relationships

Crowd will check both the `gidNumber` and the `memberUid` attributes to determine if a user is a member of a group. The name of the `gidNumber` attribute is not configurable — Crowd will always use this attribute to determine membership.

The RFC 2307 schema does not support nesting of groups, so Crowd does not support nested groups in OpenLDAP based on the Posix/NIS schema.

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration

Crowd Documentation

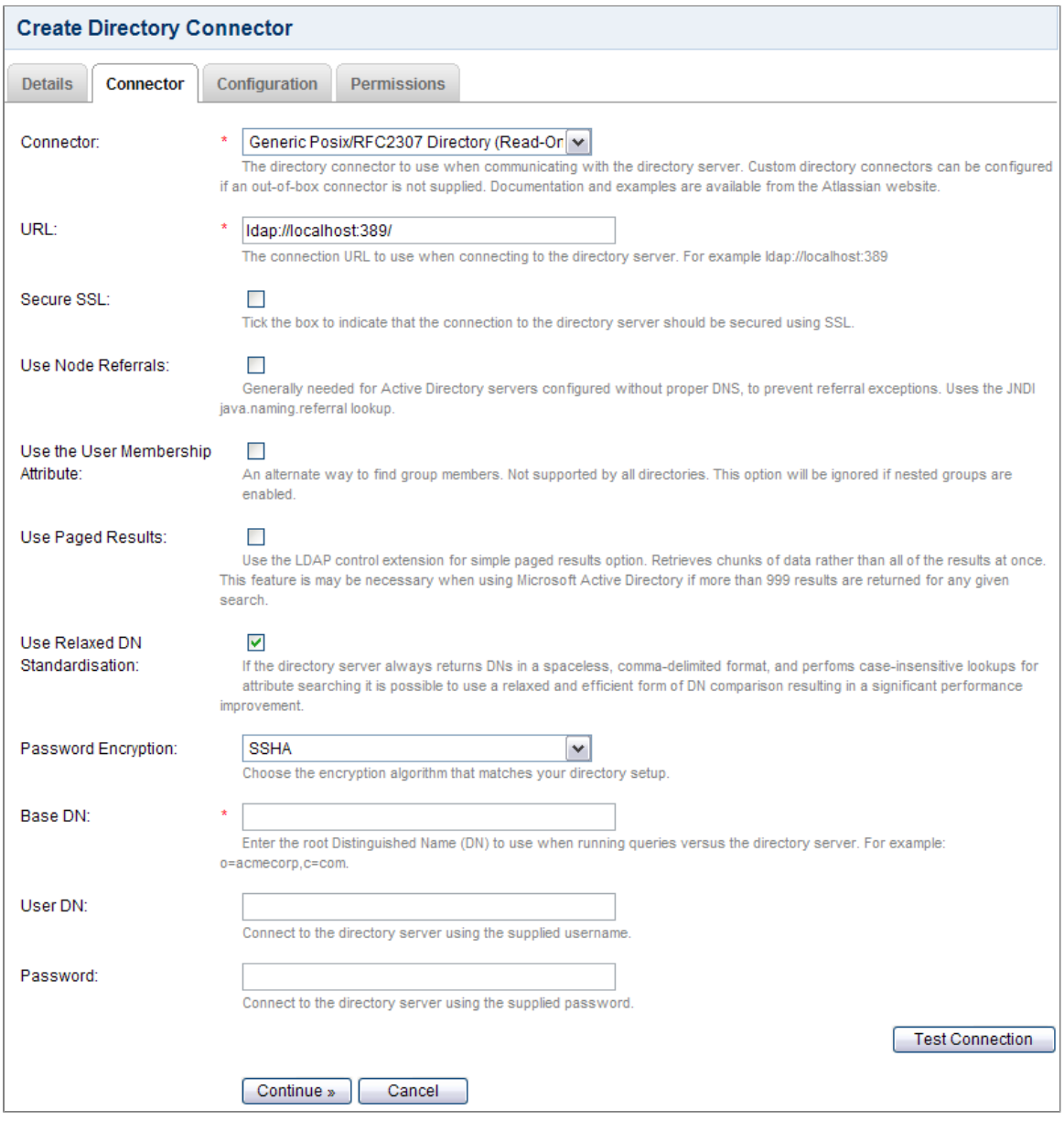
Posix Schema for LDAP

This page provides notes for configuring an LDAP directory using the Posix/NIS schema RFC 2307. This page is related to Configuring an LDAP Directory Connector.

Crowd supports read-only connections to an LDAP directory using the Posix/NIS schema. This is useful if you have a Unix installation and want to integrate with an LDAP directory. The Posix/NIS schema allows integration between an LDAP directory and the Unix NIS (Network Information Service).

> **Crowd's Posix support is read-only**
> Currently, Crowd supports read-only access to the directory based on the Posix schema. You cannot add or update user details.

Screenshot: 'Connector — LDAP using Posix schema'



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or `port 636` for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.

Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for Posix schemas. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Password Encryption	Select the type of encryption that the directory uses.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Group Relationships

Crowd will check both the `gidNumber` and the `memberUid` attributes to determine if a user is a member of a group. The name of the `gidNumber` attribute is not configurable — Crowd will always use this attribute to determine membership.

The RFC 2307 schema does not support nesting of groups, so Crowd does not support nested groups in the Posix schema.

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import

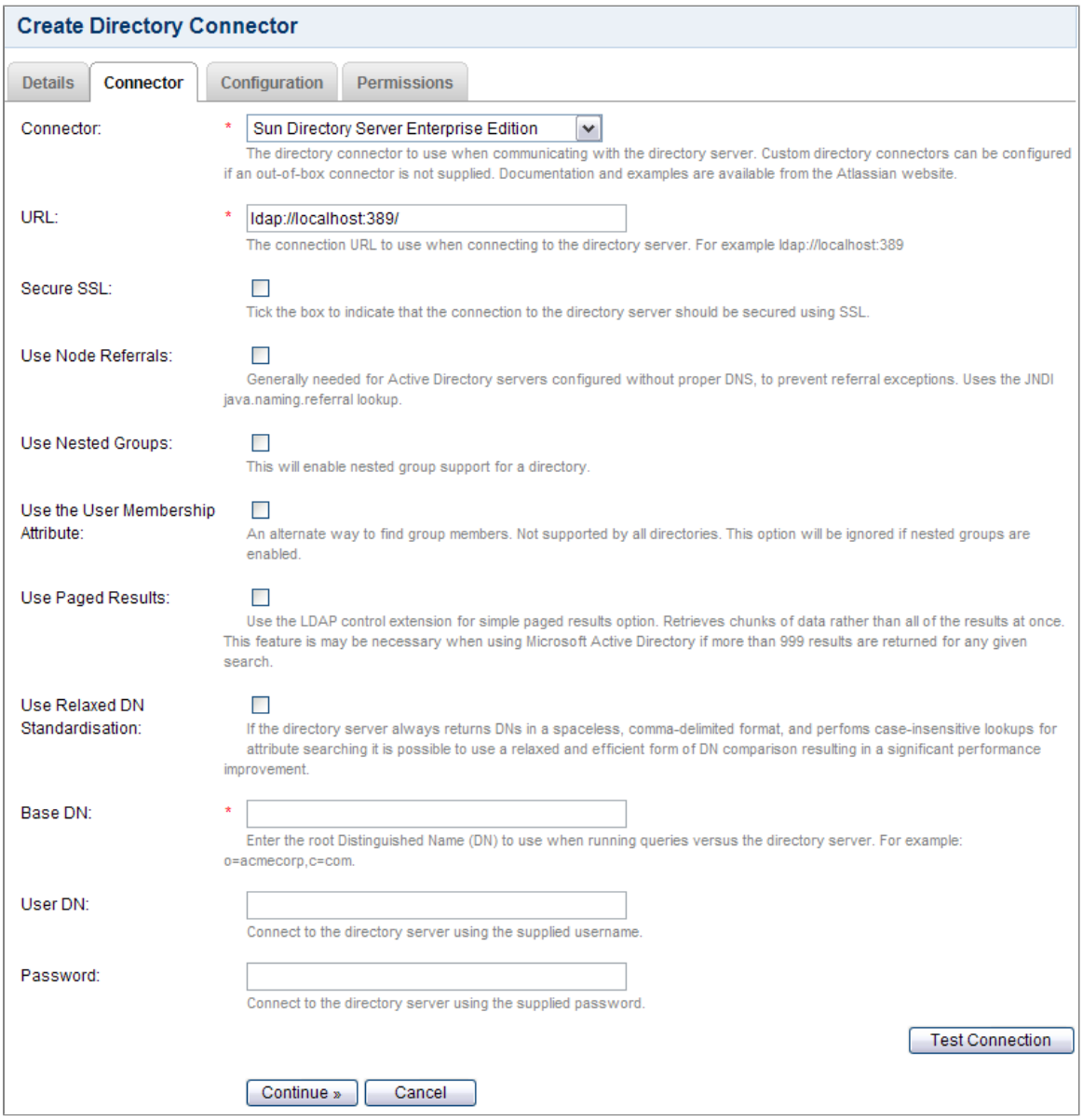
- Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration
Crowd Documentation

Sun Directory Server Enterprise Edition (DSEE)

This page provides configuration notes for Sun's Java System Directory Server Enterprise Edition (DSEE, previously called 'SunONE Directory Server'). This page is related to Configuring an LDAP Directory Connector.

Screenshot: Connector — Sun Directory Server Enterprise Edition (DSEE)



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or `port 639` for SSL.
Secure SSL	Specifies if the connection to the directory server is a SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.

Use the User Membership Attribute	Put a tick in the checkbox if your directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) For instructions on enabling this feature in your directory, please refer to the OpenLDAP documentation. • If this checkbox is ticked, Crowd will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use Paged Results	Use the LDAP control extension for simple paged results option. Retrieves chunks of data rather than all of the results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version. This is the **default and recommended setting** for Sun DSEE. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison, which will result in a significant performance improvement. This is only possible if the directory guarantees the format of DNs.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	The username that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

Configuration details for Sun DSEE

Sun DSEE Example	Value
Base DN	dc=acmecorp,dc=com
User DN	cn=Directory Manager

Next Step

Go back to Configuring an LDAP Directory Connector.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Using Apache Directory Studio for Crowd LDAP Configuration

Crowd Documentation

Configuring a Custom Directory Connector

Custom directory connectors allow developers to connect Crowd to custom user-stores, such as existing databases or legacy systems.

First you need to create a custom directory connector. The simplest way to accomplish this is to add a JAR file with the necessary classes to the Crowd `WEB-INF/lib` folder. For details, please see Creating a Custom Directory Connector.

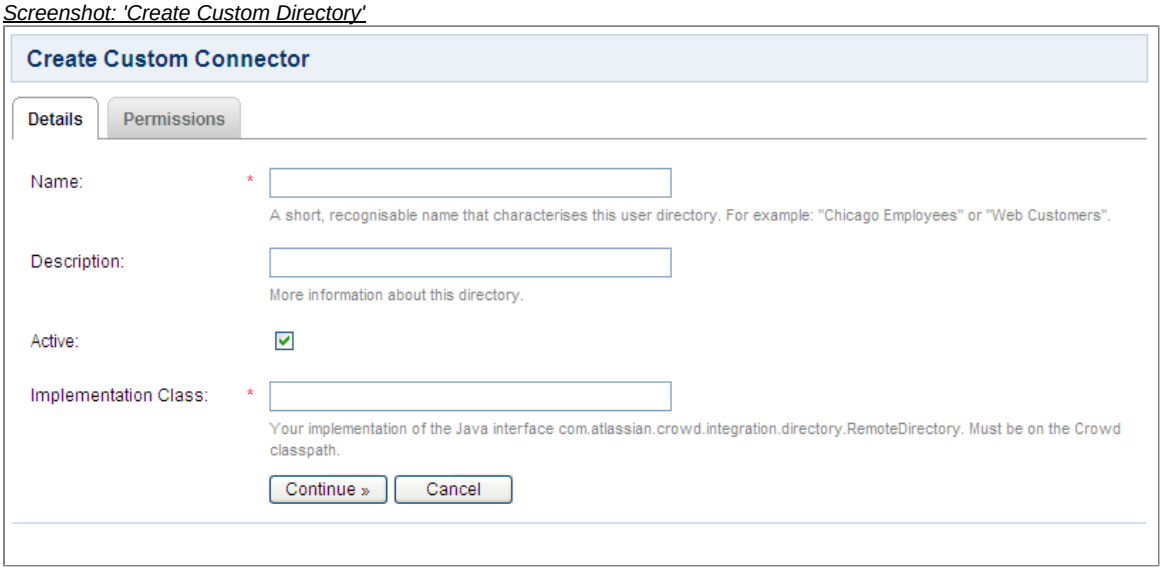
Once you have added your JAR file to the Crowd `WEB-INF/lib` folder, you are ready to configure a Custom Directory Connector, as described below.

To configure a Custom Directory Connector,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' link in the top navigation bar.
3. This will display the Directory Browser. Click the '**Add Directory**' link.
4. Click the '**Custom**' button.
5. Complete the fields as described in the table below.
6. Click the '**Continue**' button to configure the directory's permissions.

Once you have configured the directory's permissions, you will have finished configuring your new directory. You can then map the directory to appropriate applications.

Screenshot: 'Create Custom Directory'



Custom Directory Store Attributes	Description
Name	The name used to identify the directory within Crowd. This is useful when there are multiple directories configured, e.g. Chicago Employees or Web Customers.
Description	Details about this specific directory.
Active	Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications.
Implementation Class	Implementation of `com.atlassian.crowd.integration.directory.RemoteDirectory` Java interface. Must be in the Crowd CLASSPATH.

Next Step:

See Specifying Directory Permissions

Related Topics

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector

- Apache Directory Server (ApacheDS)
- Apple Open Directory
- Fedora Directory Server
- Generic LDAP Directories
- Microsoft Active Directory
  - Configuring an SSL Certificate for Microsoft Active Directory
- Novell eDirectory
- OpenDS
- OpenLDAP
- OpenLDAP Using Posix Schema
- Posix Schema for LDAP
- Sun Directory Server Enterprise Edition (DSEE)

- Configuring a Custom Directory Connector
- Configuring a Delegated Authentication Directory

- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

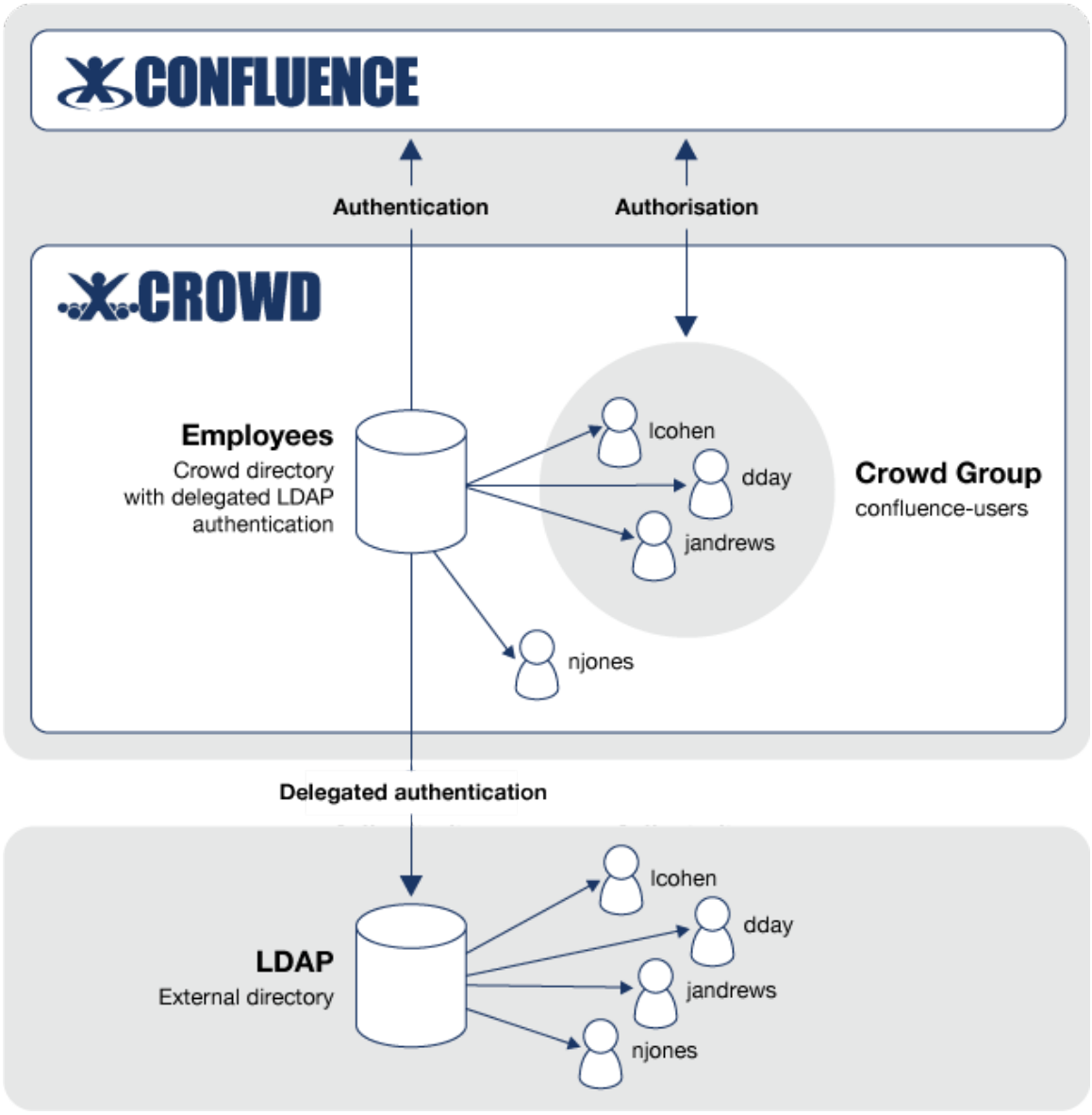
Crowd Documentation

Configuring a Delegated Authentication Directory

A Delegated Authentication directory combines the features of an internal Crowd directory with delegated LDAP authentication. This means that you can have your users authenticated via an external LDAP directory while managing the users, groups and roles in Crowd. You can use Crowd's flexible and simple group management when the LDAP groups do not suit your requirements.

For example, you can set up a simple group configuration in Crowd for use with Confluence and other Atlassian products, while authenticating your users against the corporate LDAP directory. You can also avoid the performance issues which might result from downloading large numbers of groups from LDAP.
The diagram below gives a conceptual overview of delegated LDAP authentication. This example assumes that you have:

- The Confluence application integrated with Crowd.
- A Crowd Delegated Authentication directory called 'Employees' which contains the group 'confluence-users'.
- An LDAP directory containing all your employees and their authentication details (e.g. username and password).



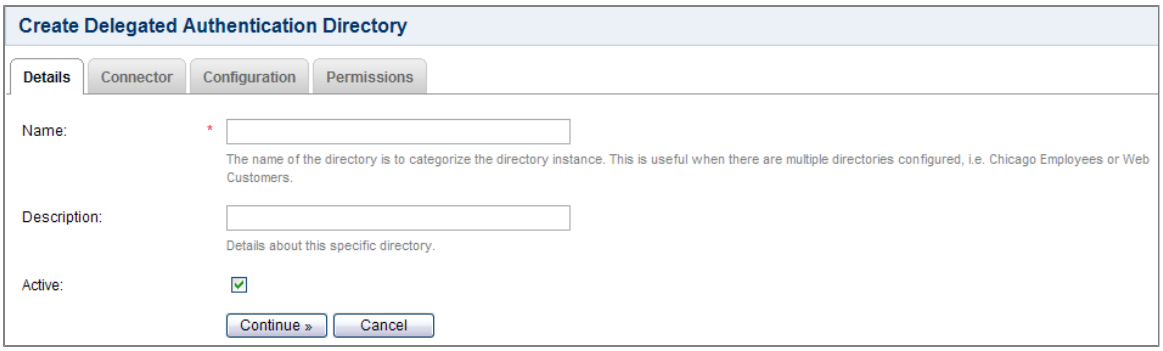
Summary of Configuration Steps

To configure a Delegated Authentication directory,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' link in the top navigation bar.
3. This will display the Directory Browser. Click the '**Add Directory**' link.
4. This will display the 'Select Directory Type' screen. Click the '**Delegated Authentication**' button.
5. This will display the '**Details**' tab (see Screenshot 1 below). Enter the '**Name**' and '**Description**' fields, then click the '**Continue**' button.
6. This will display the '**Connector**' tab (see Screenshot 2 below). Select the relevant connector type, and fill in the basic connection information for your directory server. For details, please see:
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
7. Click the '**Test Connection**' button to verify that Crowd can successfully connect to the directory.
8. Click the '**Continue**' button.
9. This will display the '**Configuration**' tab (see Screenshot 3 below). Fill in the configuration details for your users.
10. Click the '**Test Search**' button to verify that Crowd can successfully locate groups/roles/users within the directory.
11. Click the '**Continue**' button to configure the directory's permissions.

Configuring Directory Details

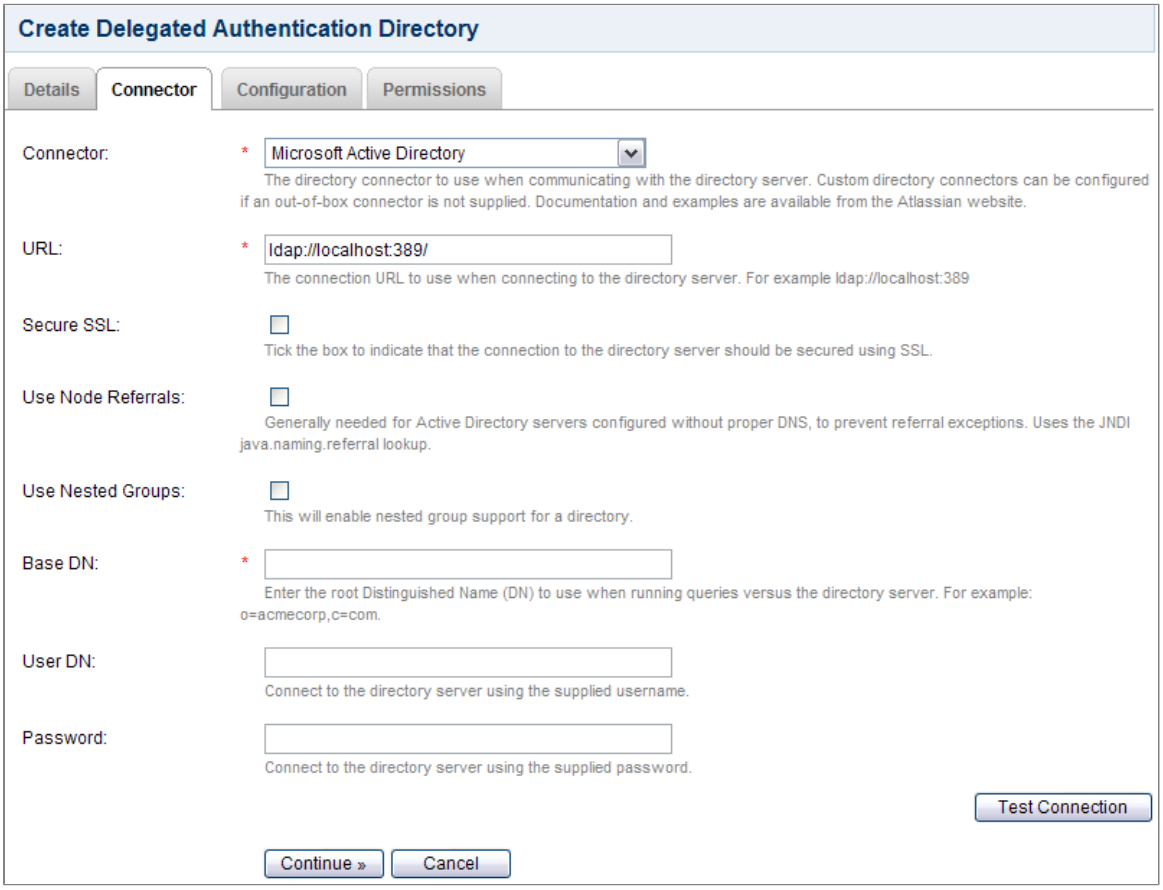
Screenshot 1: Directory details



Attribute	Description
Name	The name used to identify the directory within Crowd. For example: 'Chicago Employees' or 'Web Customers'.
Description	More information about this directory.
Active	Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications.

Configuring Connector Details

Screenshot 2: Connector



Attribute	Description
Connector	The directory connector to use when communicating with the directory server.
URL	The connection URL to use when connecting to the directory server, e.g.: `ldap://localhost:389`, or port 636 for SSL.

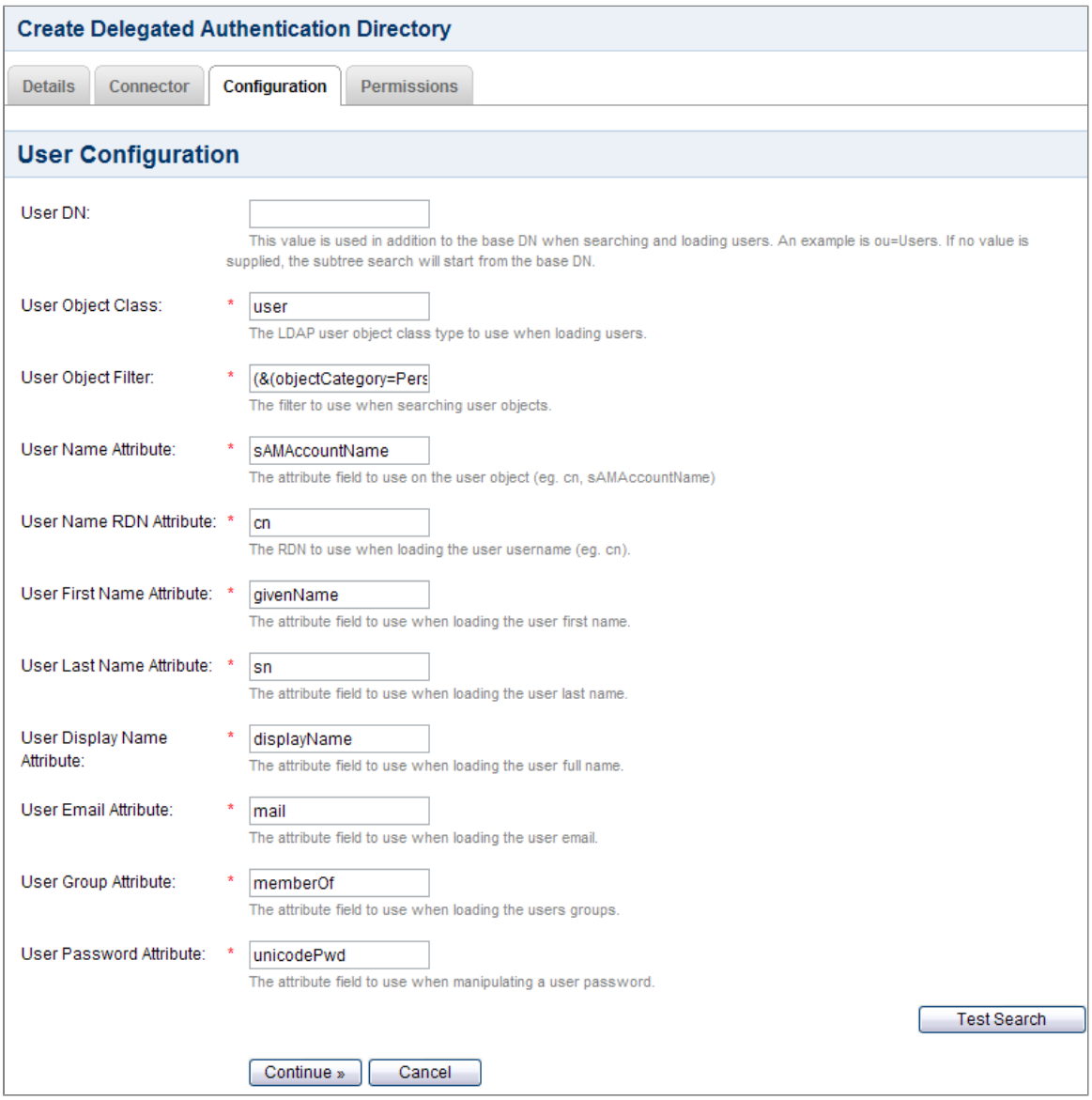
Secure SSL	Specifies whether the connection to the directory server is an SSL connection.
Use Node Referrals	Use the JNDI lookup java.naming.referral option. Generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.
Use Nested Groups	Enable or disable support for nested groups on the LDAP user directory.
Use the User Membership Attribute	Put a tick in the checkbox if your Active Directory supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.) • If this checkbox is ticked, Crowd will use the group membership attribute on the user when **retrieving the members of a given group**. This will result in a more efficient retrieval. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search. • If the 'Use Nested Groups' checkbox is ticked, Crowd will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use 'memberOf' for Group Membership	Put a tick in the checkbox if your Active Directory supports the 'memberOf' attribute on the user. • If this checkbox is ticked, Crowd will use the 'memberOf' attribute when **retrieving the list of groups to which a given user belongs**. This will result in a more efficient search. • If this checkbox is not ticked, Crowd will use the members attribute on the group ('member' by default) for the search.
Use Paged Results	Use the LDAP control extension for simple paging of search results. Retrieves chunks of data rather than all of the search results at once. This feature may be necessary when using Microsoft Active Directory if more than 999 results are returned for any given search.
Paged Results Size	Enter the desired page size i.e. the maximum number of search results to be returned per page, when paged results are enabled. Defaults to 999 results.
Use Relaxed DN Standardisation	This setting determines how Crowd will compare DNs to determine if they are equal. See Configuring Relaxed DN Standardisation. • If this checkbox is ticked, Crowd will do a direct, case-insensitive, string comparison. This is the **default and recommended setting** for Active Directory, because Active Directory guarantees the format of DNs. Using relaxed DN standardisation will result in a significant performance improvement. • If this checkbox is not ticked, Crowd will parse the DN and then check the parsed version.
Enable Caching	Put a tick in the checkbox to enable directory caching. Directory caching can provide fast recurrent access to user, group and role data for a particular directory. This can provide significant performance improvements for applications such as JIRA, which require large amounts of user information. Please read the full instructions: Configuring Caching for an LDAP Directory.
Max Cache Elements in Memory	This checkbox appears if 'Enable Caching' is ticked. Specify the maximum number of cache elements to be held in memory before overflowing to disk. Please read the full instructions: Configuring Caching for an LDAP Directory.
Polling Interval	This checkbox appears if 'Enable Caching' is ticked. Crowd will send a request to Active Directory every x seconds, where 'x' is the number specified here. Please read the full instructions: Configuring Caching for an LDAP Directory.
Base DN	Enter the root distinguished name to use when running queries versus the directory server, e.g.: `o=acmecorp,c=com`.
User DN	Distinguished name of the user that Crowd will use when connecting to the directory server.
Password	The password that Crowd will use when connecting to the directory server.

We have shown the settings for Active Directory. For details about the settings for your specific directory server, please see:

- Apache Directory Server (ApacheDS)
- Apple Open Directory
- Fedora Directory Server
- Generic LDAP Directories
- Microsoft Active Directory
- Novell eDirectory
- OpenDS
- OpenLDAP
- OpenLDAP Using Posix Schema
- Posix Schema for LDAP
- Sun Directory Server Enterprise Edition (DSEE)

Configuring LDAP Object and Attribute Settings

Screenshot 3: Configuration



Attribute	Description
User DN	This value is used in addition to the base DN (distinguished name) when searching and loading users. An example is `ou=Users`. If no value is supplied, the subtree search will start from the base DN.
User Object Class	This is the name of the class used for the LDAP user object. An example is `user`.
User Object Filter	The filter to use when searching user objects.
User Name Attribute	The attribute field to use when loading the username. Examples are `cn` and `sAMAccountName`.
User Name RDN Attribute	The RDN (relative distinguished name) to use when loading the username. An example is `cn`. The DN for each LDAP entry is composed of two parts: the RDN and the location within the LDAP directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure.
User First Name Attribute	The attribute field to use when loading the user's first name. An example is `givenName`.
User Last Name Attribute	The attribute field to use when loading the user's last name. An example is `sn`.

User Display Name Attribute	The attribute field to use when loading the user's full name. An example is `displayName`.
User Email Attribute	The attribute field to use when loading the user's email address. An example is `mail`.
User Group Attribute	The attribute field to use when loading the user's groups. An example is `memberOf`.
User Password Attribute	The attribute field to use when loading a user's password. An example is `unicodePwd`.

Please refer to the notes on LDAP object structures in the page about LDAP connectors.

Next Steps

Once you have configured the directory's permissions, you have finished configuring your new directory.

Next steps will be:

1. Map the directory to the appropriate applications.
2. Consider how you would like to add your users to Crowd's Delegated Authentication directory. There are a few options:
    - Manually add the users to the Crowd directory.
    - Use Crowd's Directory importer to copy your LDAP users into your Delegated Authentication directory.
    - Let Crowd do it for you, at login time. If a user logs in successfully via LDAP authentication but does not yet exist in Crowd, Crowd will automatically add them to the Delegated Authentication directory. You will then need to add the user to any necessary groups, to allow them to access applications where group membership is required.

Same username required in Crowd and LDAP
The username must be the same in the Crowd Delegated Authentication directory and in the LDAP directory. Changing the username in LDAP will break the link to the Crowd Delegated Authentication directory.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
    - Configuring an Internal Directory
    - Configuring an LDAP Directory Connector
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
    - Importing Users from Atlassian Confluence
    - Importing Users from Atlassian JIRA
    - Importing Users from Atlassian Bamboo
    - Importing Users from Jive Forums
    - Importing Users from CSV Files
    - Importing Users from One Crowd Directory into Another

Crowd Documentation

Configuring Caching for an LDAP Directory

Directory caching can be used to enable fast recurrent access to user, group and role data for a particular directory.

This page describes caching that can be configured on the Crowd server, to store user and group information from a Crowd-connected LDAP directory. For an overview of the other types of caching offered by Crowd, please refer to Overview of Caching.

Passwords are not cached
The Crowd cache does not store user passwords. All authentication is performed by calls to the LDAP directory itself.

On this page:

- Features of LDAP Caching in Crowd
- Supported LDAP Directories
- Configuring the Cache

    - Setting the Maximum Number of Elements in Memory
    - Setting the Polling Interval for Cache Updates
- Inspecting and Flushing the Cache
- Limitations
    - Limitations for All Directories
    - Additional Limitations for Microsoft Active Directory
- Error Handling

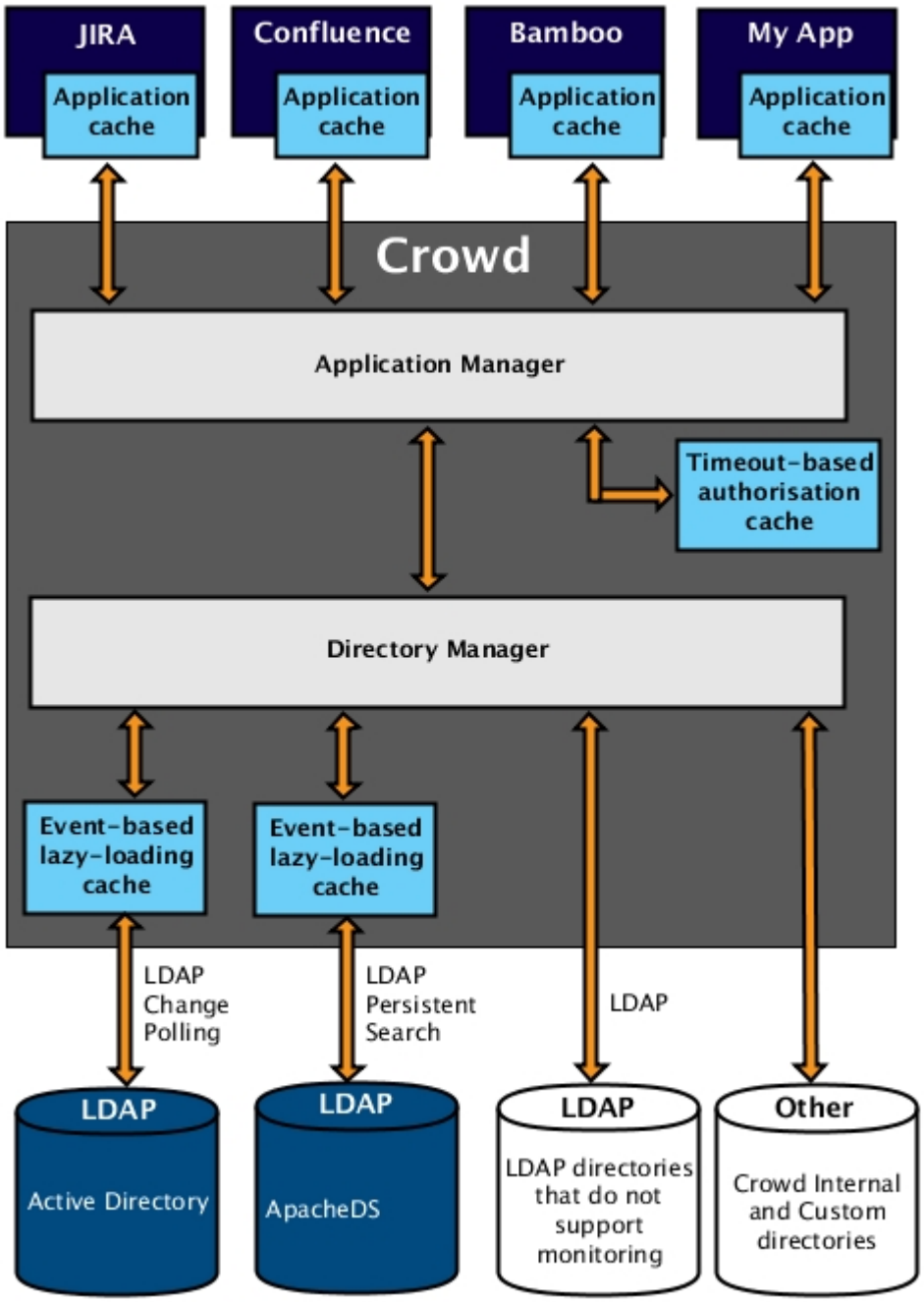
Features of LDAP Caching in Crowd

Where the LDAP directory supports it, Crowd will keep an up-to-date cache of user, group and role information retrieved from the LDAP directory. Use of the cache should improve performance particularly in directories which are large, slow or off site.

Summary of the caching features:

- The cache uses lazy loading where possible, storing only the information that is required rather than loading the entire directory into the cache.
- Crowd ensures that the cache remains up to date by monitoring the LDAP directory for updates. When a change occurs, Crowd updates the server-side cache incrementally. The monitoring mechanism depends on the type of LDAP directory, as described in the list of supported directories below.
- The caches are held in memory on the Crowd server machine. They can become considerably large. When necessary, the cache will overflow to disk.

The diagram below gives a conceptual overview of the caches supported by Crowd, including the LDAP caching discussed on this page. For an overview of the other types of caching offered by Crowd, please refer to Overview of Caching.



Supported LDAP Directories

Directory	Monitoring Mechanism
ApacheDS version 1.5.0 and later	Listening, via 'persistent search'. See details below.
Microsoft Active Directory	Polling, via 'uSNChanged'. See details below.

LDAP Persistent Search

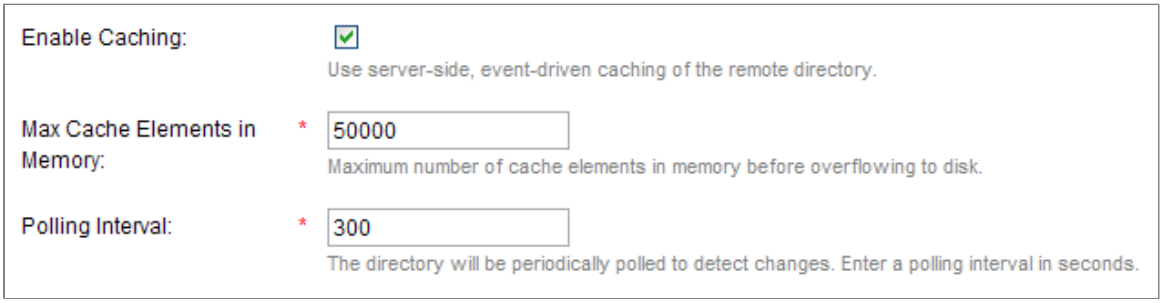
For supported LDAP directories, Crowd monitors changes via the LDAP change notification feature known as 'persistent search'. The word 'persistent' means that the search remains active forever, once initiated. Crowd performs the initial search on the LDAP directory and receives the results. From that point on, whenever an entry in the result set is updated, the LDAP directory sends Crowd a new copy of that entry.

Microsoft Active Directory Change Notification

Crowd sends a request to AD at regular intervals, asking to be notified of changes made since the last request, via the `uSNChanged` attribute. You can configure the time interval on Crowd's Directory Connector screen for your Active Directory. Details are in MSDN.

Configuring the Cache

Screen snippet: Cache Configuration for Microsoft Active Directory



You can **enable or disable the cache** for each directory via the Crowd Directory Connector screen, provided that the directory supports LDAP caching. Below are descriptions of and guidelines for the configuration settings.

Setting the Maximum Number of Elements in Memory

When configuring the cache, you can set the '**Max Cache Elements in Memory**'. This number is proportional to the number of users/groups that can be stored in memory before overflowing to disk. If you have limited JVM memory constraints, you can set the number to a lower value. Note that loading from disk can be significantly slower than loading from memory.

Crowd uses Ehcache for its cache implementation. The maximum number of elements in memory is one of the configuration settings allowed by Ehcache. Each cached directory consists of about 20 internal caches. The maximum number of elements in memory corresponds to the maximum elements per internal cache.

The largest internal cache maps DNs to principals/groups/roles. Therefore the maximum number of elements in memory should approximate the sum of principals + groups + roles in the scope of your configured LDAP subtrees.

We recommend leaving this value at the default 50,000 unless you experience memory problems. This setting means that if the number of principals + groups + roles exceeds 50,000, then some of the entities will overflow to a disk-based cache.

We have tested with up to 8,000 users and 8,000 groups. We used 512 MB of JVM memory (-Xmx) and set the maximum number of elements in memory to 50,000.

Setting the Polling Interval for Cache Updates

When configuring the cache for Microsoft Active Directory, you can set the '**Polling Interval**'. This is the time interval (number of seconds) that Crowd will wait between its requests for updates from AD.

The length of your polling interval depends on the length of time you can tolerate stale data. If you poll more frequently, then your data will be more up to date. The downside of polling more frequently is that you may overload your AD server with requests.

A good value for the polling interval would take into account the performance of your AD server and the size of the pipe between the AD server and the Crowd server.

If in doubt, we recommend that you start with an interval of 5 or 10 minutes and reduce the value incrementally. You will need to experiment with your setup. You can use Crowd's performance profiling feature to see the performance of your setup.

Very short and very long intervals:

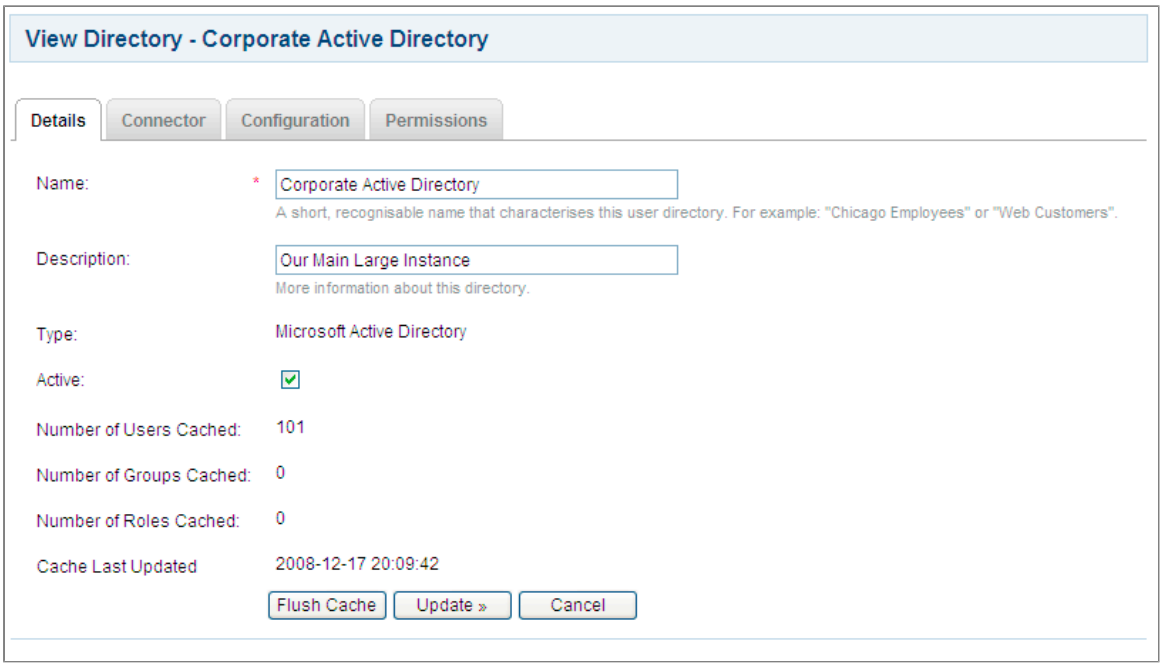
- We have tested polling intervals as short as 15 seconds, when the AD server is on the local network. Any shorter would mean that the polling interval exceeds the time taken to perform the poll operation. Each poll consists of 6 operations: searching for created or updated principals/groups/roles (3), and searching for deleted principals/groups/roles (3).
- Alternatively, if you are confident that most of the AD updates will be done via Crowd and that there will be very few changes on the AD server that do not originate from Crowd, then you can set the polling interval to be something much larger, e.g. 2 hours. Note that in this case, any change made to the AD server outside of Crowd may take up to two hours to appear in Crowd.

Inspecting and Flushing the Cache

You can view directory information via the Directory Browser. The 'View Directory' screen allows you to:

- See basic cache information — View the number of users, groups, and roles cached and the date on which the cache was last updated.
- Flush the cache — Click the '**Flush Cache**' button to remove all cached elements. Crowd will lazily reload data when the data is next requested by a client application.

Screenshot: Inspecting and flushing the cache



Limitations

Limitations for All Directories

The following comments apply to all directory types, including Microsoft Active Directory.

1. **Only specifed directories are supported.** — Crowd can only support caching for directories which provide a suitable mechanism. See the list of supported directories above.
2. **Memory usage is higher.** — Because of the memory requirements imposed by the caching, we recommend increasing the amount of heap allocated to Crowd to at least 512MB (`-Xmx512m`).
3. **Delegated Authentication directories are not supported.** — Delegated Authentication directories are not cached, because only the authentication is delegated to the directory, and authentication itself is not cached.
4. **Posix/NIS schema is not supported** — LDAP directories using the Posix/NIS schema RFC 2307 will not be cached, because the group memberships fetching scheme does not support caching.
5. **Externally moving objects out of scope causes problems.** — Do not use the external LDAP directory interface to move objects out of the scope of the sub-tree, as defined on Crowd's Directory Connector screen. This will result in an inconsistent cache. If you do need to make structural changes to your LDAP directory, flush the directory cache after you have made the changes to ensure cache consistency.
6. **Nested groups will cause a single large performance hit when finding a user's memberships.** — If you are using nested groups, you will notice a single (possibly huge) performance hit on the first call to find the memberships of user. This one-off hit will occur every time the cache is flushed, such as when Crowd is restarted or when you manually flush the cache.
7. **Unique entities are required, with respect to entity type and name** — There can be only one entity with a specific entity name and type. For example, you cannot have two groups with the same name in the visible group tree.
8. **DN mapping must be unique.** — Directory entities must be mapped in such a way that their DN mapping is unique. For example you cannot have an entity with DN that could correspond to a group and a role.
9. **Renaming objects is not supported.** — If the DN of an object is changed externally, the cache will be out of date until flushed.

Additional Limitations for Microsoft Active Directory

In addition to the general limitations listed above, please take note of these comments which apply specifically to Microsoft Active Directory (AD).

1. **Syncing between AD servers is not supported.** — Microsoft Active Directory does not replicate the `uSNChanged` attribute across instances. For that reason, Crowd does not support connecting to different AD servers for syncing. (You can of course define multiple different directories in Crowd, each pointing to its own respective AD server.)
2. **You must restart Crowd after restoring AD from backup.** — On restoring from backup of an AD server, the `uSNChanged` timestamps are reverted to the backup time. To avoid the resulting confusion, you will need to flush the directory cache after a Active Directory restore operation.
3. **Obtaining object deletions requires Administrator access.** — Active Directory stores deleted objects in a special container called `cn=Deleted Objects`. By default, to access this container you need to connect as an Administrator and so, for Crowd to be aware of deletions, you must use Administrator credentials. Alternatively, it's possible to change the permissions on the `cn=Deleted Objects` container. If you wish to do so, please see this Microsoft KB Article.

Error Handling

If Crowd detects a connection timeout or an error, Crowd will automatically flush the caches and re-start the directory monitors. To manually flush the cache and re-start the monitors, use the "Flush Cache" button.

RELATED TOPICS

- Overview of Caching
- Authorisation Caching
- Configuring Caching for an Application
- Configuring Relaxed DN Standardisation
- Configuring an LDAP Directory Connector
- Managing Directories
- A blog post by a Crowd developer on Caching in Crowd 1.6

Crowd Documentation

Configuring Relaxed DN Standardisation

When configuring an LDAP directory connector in Crowd, you can set relaxed DN standardisation on or off. This page gives some background to the setting of this option.

Crowd needs to compare DNs (distinguished names) to check a number of things, such as whether a user is a member of a group. Some directories guarantee that DNs will always be in a standard format, and some return slight variants with changes such as extra whitespace. If we know that, in a specific directory, DNs are case insensitive and are always returned in a compact format (i.e. the separators are commas without spaces) then we can convert both the attribute names and values to lower case and just do a direct string comparison.

Using relaxed DN standardisation provides significant performance benefits, so we recommend enabling it where possible.

Effect of Turning Relaxed DN Standardisation On or Off

Relaxed DN Standardisation in Crowd	Processing in Crowd	Comments
Off	Crowd will perform the full DN parsing and compare the parsed version.	See below for default settings for each directory type.
On	Crowd will perform a `toLower` operation and then do a direct comparison of the two DN strings.	If this setting is 'off' by default for your directory type (see below) then you may be able to turn it on. Both of the following two statements need to be true: 1. The directory server always returns memberDNs in a compact format i.e. the separators are commas without spaces. For example: • Compact format: '`cn=bob,dc=example,dc=com`' • Not compact: '`cn=bob, dc=example, dc=com`' 2. The attribute names in the RDN are always lower case, or all searches for DNs and memberDN attributes are case insensitive.

Default Settings in Crowd

Crowd ships with the following default settings, as determined by the characteristics of each directory type.

Directory Type	Relaxed DN Standardisation
ApacheDS 1.0.x	Off
ApacheDS 1.5.x	Off
Apple Open Directory	On
FedoraDS	On
Generic LDAP	Off
Microsoft Active Directory	On
Novell eDirectory	Off
OpenDS	Off
OpenLDAP	On
OpenLDAP Posix	On
Generic Posix	On
Sun Directory Server DSEE	Off

RELATED TOPICS

- Overview of Caching
- Managing Directories

Crowd Documentation

Specifying Directory Permissions

Directory permissions allow you to restrict the way in which directories can be used by mapped applications. Often, administrators need to limit applications to only being able to read — not modify — directory entity data, i.e. the users, groups and roles contained within the directory. You can achieve this by disabling the relevant directory permissions.

Directory permissions are defined at two levels:

1. **Directory-level permissions** are defined on the 'Permissions' tab of the 'View Directory' screen. These permissions apply to each application mapped to the directory, unless the application has its own application-level permissions.
2. **Application-level directory permissions** are defined on the 'Permissions' tab of the 'View Application' screen. If a permission is enabled at directory level, you can enable it for a specific application. For example, you could enable the 'Add User' permission on the 'Customers' directory in JIRA but disable the permission for Confluence.

Take a look at an example.

Disabling a directory-level permission will override any permissions enabled at application level. If a permission is enabled at application level and then subsequently disabled at directory level, the directory-level permission will apply. (The application-level permissions will be 'remembered' and will apply again if re-enabled at directory level.)

How do directory permissions affect the Crowd application (Crowd Administration Console)?

- If a particular permission is turned off at directory level, then **no** application can perform the related function - not even the Crowd application. So, for example, if you disable the 'Remove User' permission for a directory, then the Crowd Administration Console will not allow you to delete a user from that directory.
- The Crowd application is not bound by application-level permissions.

Below, we tell you about directory-level permissions. You can also read more about application-level directory permissions.

Directory-Level Permissions

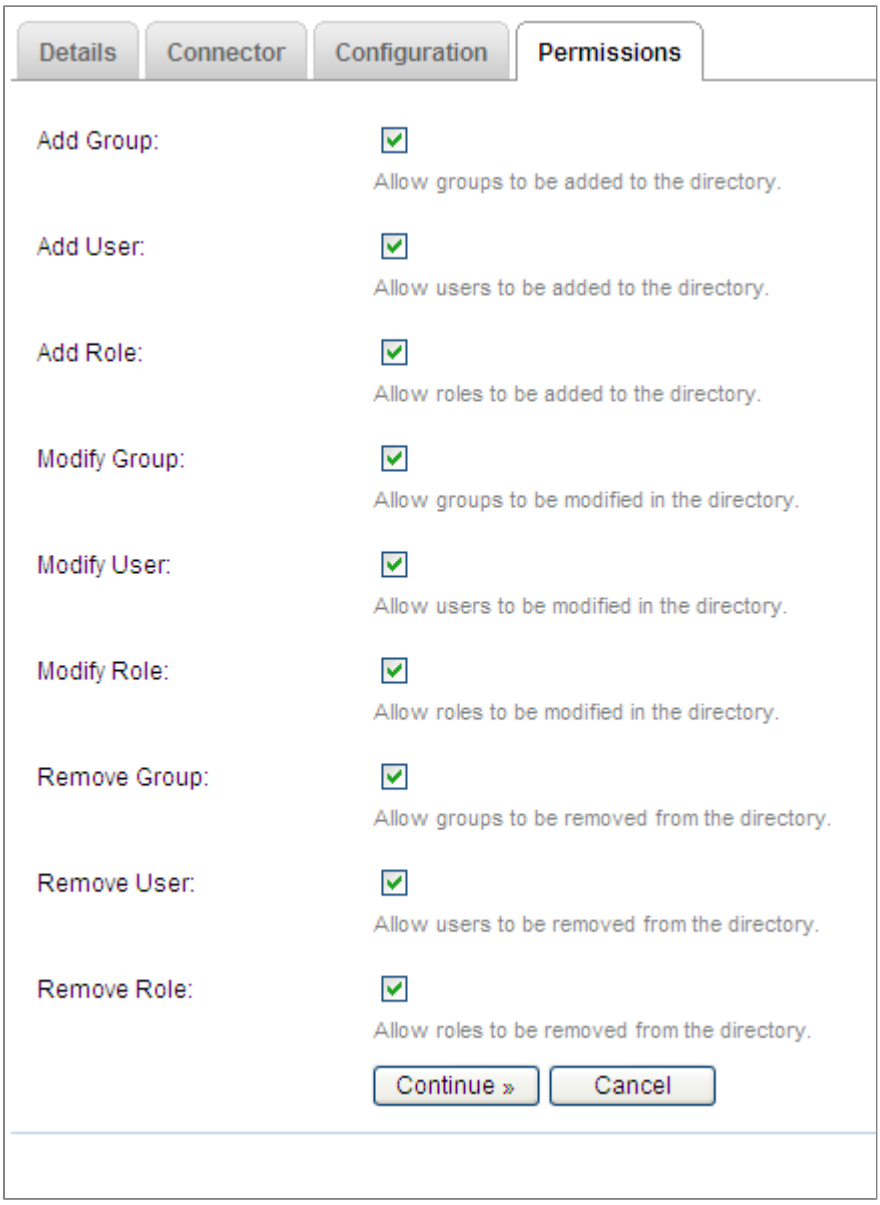
Permission	Description
Add Group	Allows applications to add groups to the directory.
Add User	Allows applications to add users to the directory.
Add Role	Allows applications to add roles to the directory.
Modify Group	Allows applications to modify groups in the directory.
Modify User	Allows applications to modify users in the directory.
Modify Role	Allows applications to modify roles in the directory.
Remove Group	Allows applications to delete groups from the directory.
Remove User	Allows applications to delete users from the directory. Consider carefully whether you allow the deletion of users, as some applications contain historical data, e.g. documents that the user has created. Read more.
Remove Role	Allows applications to delete roles from the directory.

When you add a new directory, all of its permissions are enabled by default.

To specify directory permissions,

1. Configure a new directory as described in Adding a Directory ***or*** select an existing directory from the Directory Browser .
2. Click the '**Permissions**' tab. This will display a list of permissions as shown in the screenshot below.
   - **To enable a directory permission**, select the corresponding checkbox.
   - **To disable a directory permission**, deselect the corresponding checkbox.

Screenshot: Directory permissions



Need to grant users permission to access an application?
To control which users within a directory may access a mapped application, see Specifying which Groups can access an Application.

RELATED TOPICS

Specifying an Application's Directory Permissions

- Using the Directory Browser
- Adding a Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory

Crowd Documentation

Importing Users and Groups into a Directory

Once you have added a directory, you can import groups and users into it from external user-stores or from another directory defined in Crowd. This can reduce the number of user-stores within your organisation, and give you a consolidated, centralised point of user management. Once you have imported users into a Crowd directory, you can manage them via the Crowd Administration Console (assuming the directory's permissions allow this).

For example, your organisation might currently have user IDs for Atlassian JIRA users stored within JIRA's database, and user IDs for Jive

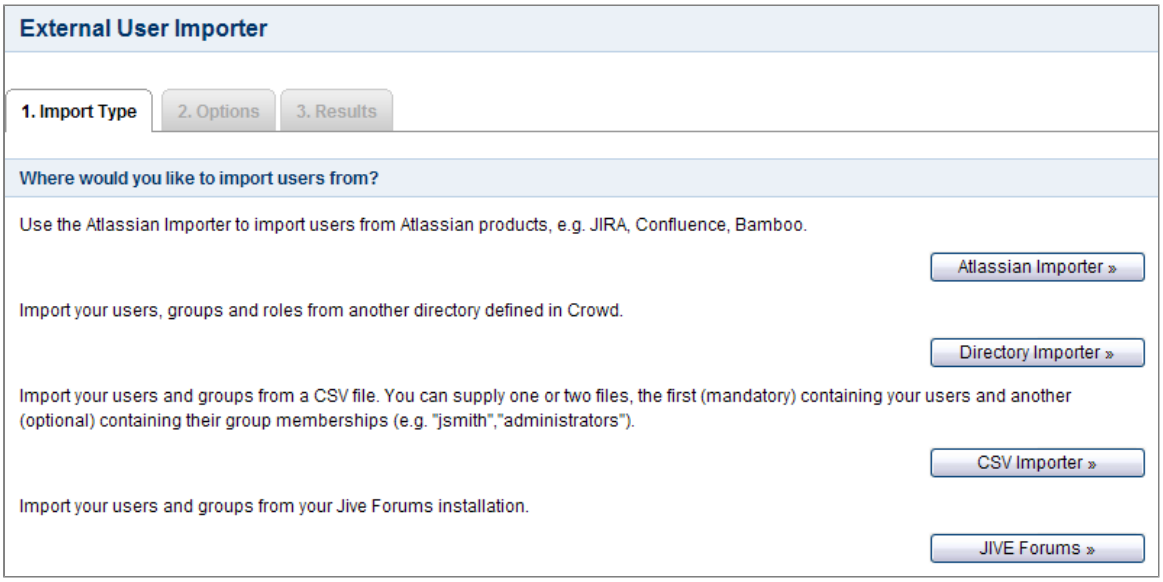
Forums users stored within Jive's database. You could use Crowd to import all the user IDs from both places into Microsoft Active Directory.

You can import from different user-stores into a single Crowd directory, or into different Crowd directories, depending on your needs.

To import users into a directory,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Click the '**Import Users**' link.
4. This will display the '**Import Type**' screen (see below). Click the button corresponding to the type of user-store or file from which you want to import external users into Crowd:
    - '**Atlassian Importer**' — see Importing Users from Atlassian Confluence, Importing Users from Atlassian JIRA and Importing Users from Atlassian Bamboo
    - '**Directory Importer**' — see Importing Users from One Crowd Directory into Another
    - '**CSV Importer**' — see Importing Users from CSV Files
    - '**JIVE**' — see Importing Users from Jive Forums

Screenshot: 'Select Import Type'



Related Topics

- Using the Directory Browser
- Adding a Directory
    - Configuring an Internal Directory
    - Configuring an LDAP Directory Connector
        - Apache Directory Server (ApacheDS)
        - Apple Open Directory
        - Fedora Directory Server
        - Generic LDAP Directories
        - Microsoft Active Directory
            - Configuring an SSL Certificate for Microsoft Active Directory
        - Novell eDirectory
        - OpenDS
        - OpenLDAP
        - OpenLDAP Using Posix Schema
        - Posix Schema for LDAP
        - Sun Directory Server Enterprise Edition (DSEE)
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
    - Importing Users from Atlassian Confluence
    - Importing Users from Atlassian JIRA
    - Importing Users from Atlassian Bamboo
    - Importing Users from Jive Forums
    - Importing Users from CSV Files
        - Configuring the CSV Importer
        - Mapping CSV Fields to Crowd Fields
        - Confirming the CSV Importer Configuration
        - Viewing the Results of the Import

- Importing Users from One Crowd Directory into Another

Crowd Documentation

Importing Users from Atlassian Confluence

If you have already been using Atlassian Confluence, and are now configuring Confluence as a Crowd application, you will probably want to import your existing Confluence users and groups into a Crowd directory.

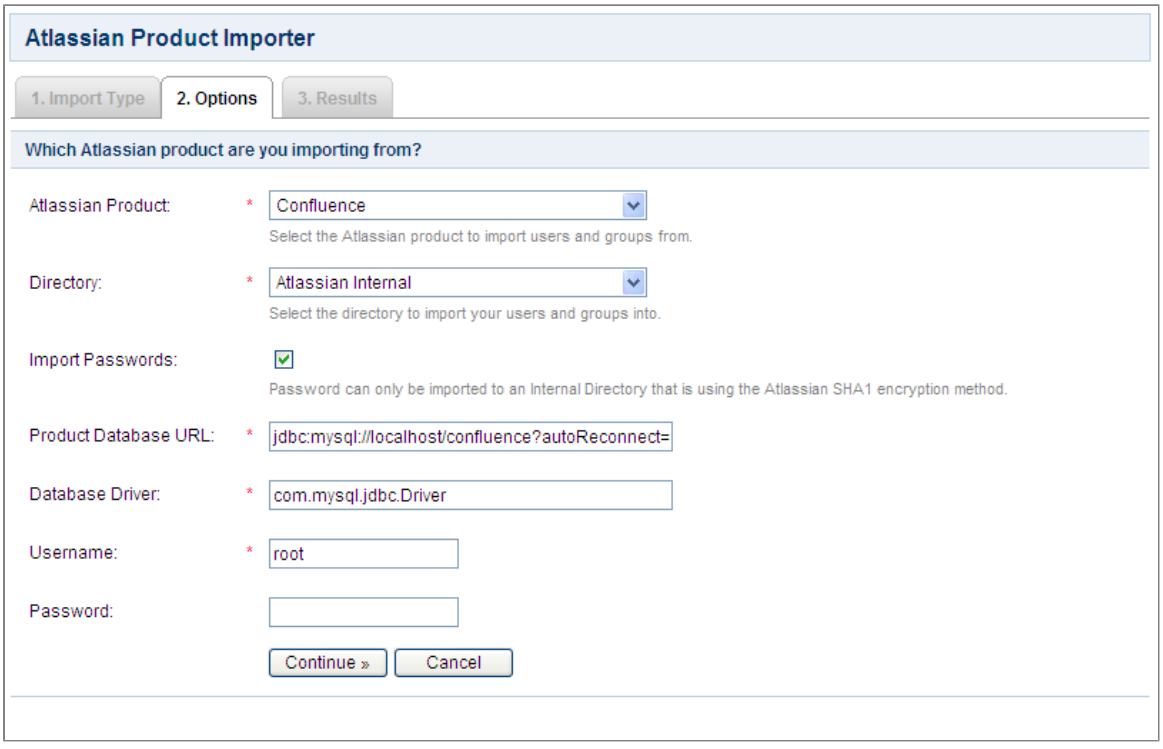
It is recommended that you import your Confluence users into an Internal Directory that has its '**Password Encryption**' set to '**ATLASSIAN-SHA1**'. Otherwise, users' passwords will not be copied across to Crowd.

To import users and groups from Atlassian Confluence into a Crowd directory,

1. Ensure that the database driver for the Confluence database is on Crowd's classpath. To do this, simply copy the JDBC driver jar for your particular Confluence database across to `apache-tomcat/common/lib` in your Crowd installation directory. Then restart Crowd.
2. Log in to the Crowd Administration Console.
3. Click the '**Users**' link in the top navigation bar.
4. This will display the User Browser. Click the '**Import Users**' link.
5. This will display the '**Import Type**' screen. Click the '**Atlassian Importer**' button.
6. This will display the '**Options**' screen. Complete the fields as follows:
    - '**Atlassian Product**' — Select 'Confluence'.
    - '**Directory**' — Select the directory that you have created for your Confluence users.
    - '**Import Passwords**' — Select this checkbox if you wish to import the users' passwords from Confluence. You can only import passwords if the Crowd directory is using the 'Atlassian SHA1' encryption method.
    - '**Product Database URL**' — Type the URL of your Confluence instance's database. The exact syntax will depend on which database you are using; see Database Configuration in the *Confluence Configuration Guide*.
    - '**Database Driver**' — type the name of your Confluence instance's database JDBC driver (e.g. for MYSQL, type `com.mysql.jdbc.Driver`).
    - '**Username**' — Type the username of the database user that Crowd will use to login to your Confluence instance's database.
    - '**Password**' — Type the password of the database user Crowd will use to login to your Confluence instance's database.

      The import process will log in to the database, not into Confluence.
7. Click the '**Continue**' button to import the users from your Confluence instance into your Crowd directory.
8. The '**Results**' screen will be displayed, showing how many users and groups have been imported into your Crowd directory.
9. Click the '**Users**' button to view and manage the imported users and groups via the Crowd Administration Console (assuming the directory's permissions allow this).

Screenshot: 'Import Confluence Users'



Next Step

To give the imported groups access to the Confluence application, see Specifying which Groups can access an Application.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Crowd Documentation

Importing Users from Atlassian JIRA

If you have already been using Atlassian JIRA, and are now configuring JIRA as a Crowd application, you will probably want to import your existing JIRA users and groups into a Crowd directory.

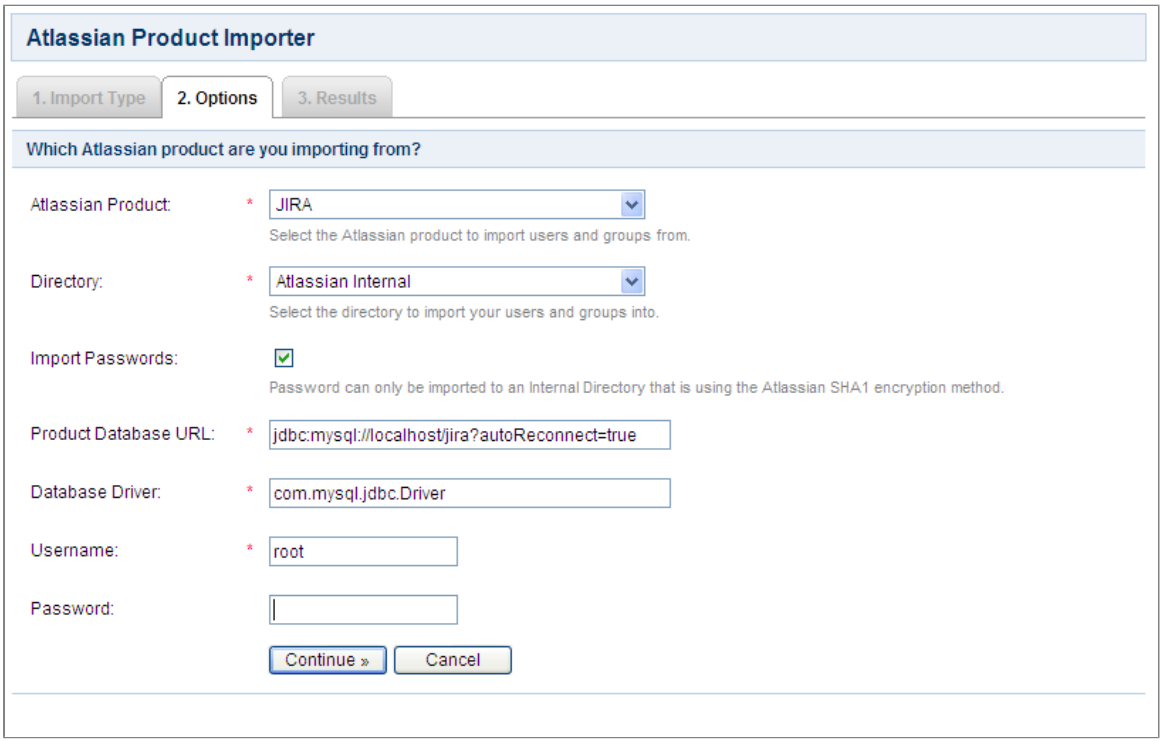
It is recommended that you import your JIRA users into an Internal Directory that has its '**Password Encryption**' set to '**ATLASSIAN-SHA1**'. Otherwise, users' passwords will not be copied across to Crowd.

To import users and groups from Atlassian JIRA into a Crowd directory,

1. Ensure that the database drivers for the JIRA database are on Crowd's classpath. To do this, simply copy the JDBC driver jar for your particular JIRA database across to `apache-tomcat/common/lib` in your Crowd installation directory. Then restart Crowd.
2. Log in to the Crowd Administration Console.
3. Click the '**Users**' link in the top navigation bar.
4. This will display the User Browser. Click the '**Import Users**' link.
5. This will display the '**Import Type**' screen. Click the '**Atlassian Importer**' button.
6. This will display the '**Options**' screen. Complete the fields as follows:
   - '**Atlassian Product**' — Select 'JIRA'.
   - '**Directory**' — Select the directory that you have created for your JIRA users.
   - '**Import Passwords**' — Select this checkbox if you wish to import the users' passwords from JIRA. You can only import passwords if the Crowd directory is using the 'Atlassian SHA1' encryption method.
   - '**Product Database URL**' — Type the URL of your JIRA instance's database. The exact syntax will depend on which database you are using; see Connecting JIRA to a Database in the *JIRA Installation Guide*.
   - '**Database Driver**' — Type the name of your JIRA instance's database JDBC driver (e.g. for MYSQL, type `com.mysql.jdbc.Driver`).
   - '**Username**' — Type the username of the database user that Crowd will use to log in to your JIRA instance's database.
   - '**Password**' — Type the password of the database user Crowd will use to log in to your JIRA instance's database.

     The import process will log in to the database, not into JIRA.
7. Click the '**Continue**' button to import the users from your JIRA instance into your Crowd directory.
8. The '**Results**' screen will be displayed, showing how many users and groups have been imported into your Crowd directory.
9. Click the '**Users**' button to view and manage the imported users and groups via the Crowd Administration Console (assuming the directory's permissions allow this).

Screenshot: 'Import JIRA Users'



Next Step

To give the imported groups access to the JIRA application, see Specifying which Groups can access an Application.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Crowd Documentation

Importing Users from Atlassian Bamboo

If you have already been using Atlassian Bamboo, and are now configuring Bamboo as a Crowd application, you will probably want to import your existing Bamboo users and groups into a Crowd directory.

We recommend that you import your Bamboo users into an internal Crowd directory that has its '**Password Encryption**' set to '

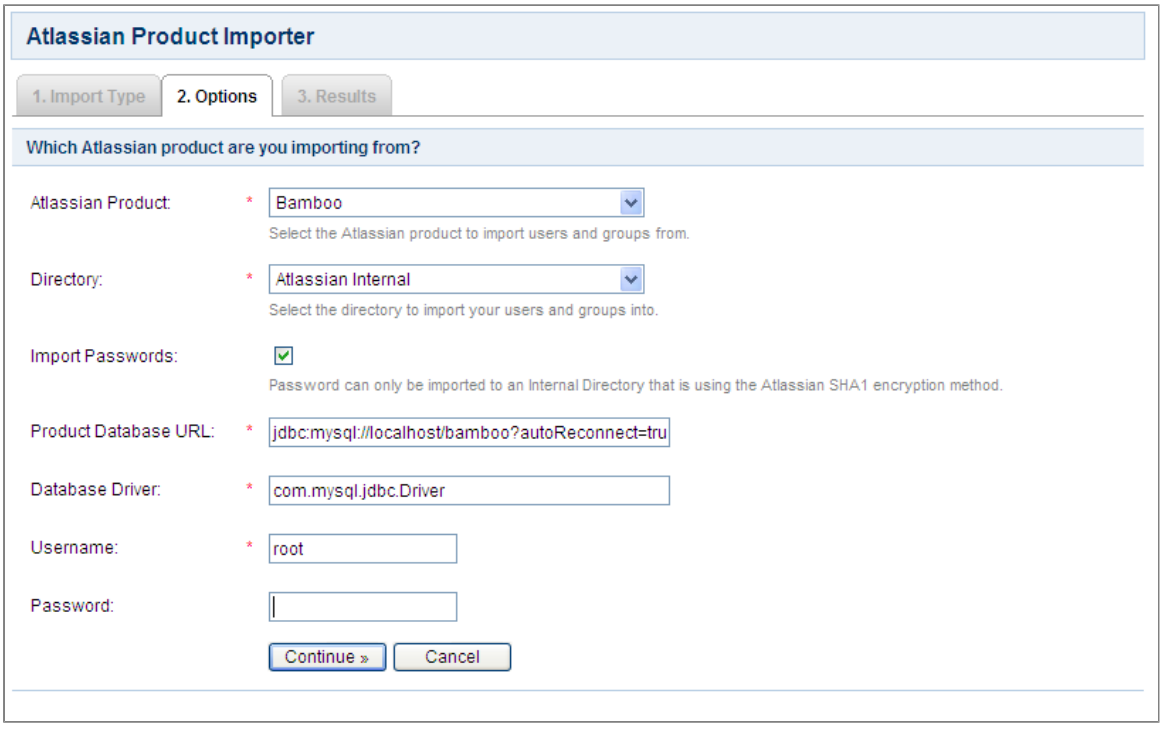
ATLASSIAN-SHA1'. Otherwise, users' passwords will not be copied across to Crowd.

To import users and groups from Atlassian Bamboo into a Crowd directory,

1. Ensure that the database drivers for the Bamboo database are on Crowd's classpath. To do this, simply copy the JDBC driver jar for your particular Bamboo database across to `apache-tomcat/common/lib` in your Crowd installation directory. Then restart Crowd.
2. Log in to the Crowd Administration Console.
3. Click the '**Users**' link in the top navigation bar.
4. This will display the User Browser. Click the '**Import Users**' link.
5. This will display the '**Import Type**' screen. Click the '**Atlassian Importer**' button.
6. This will display the '**Options**' screen. Complete the fields as follows:
   - '**Atlassian Product**' — Select 'Bamboo'.
   - '**Directory**' — Select the directory that you have created for your Bamboo users.
   - '**Import Passwords**' — Select this checkbox if you wish to import the users' passwords from Bamboo. You can only import passwords if the Crowd directory is using the 'Atlassian SHA1' encryption method.
   - '**Product Database URL**' — Type the URL of your Bamboo instance's database. The exact syntax will depend on which database you are using. See Database Configuration in the *Bamboo Installation Guide*.
   - '**Database Driver**' — Type the name of your Bamboo instance's database JDBC driver (e.g. for MYSQL, type `com.mysql.jdbc.Driver`).
   - '**Username**' — Type the username of the database user that Crowd will use to log in to your Bamboo instance's database.
   - '**Password**' — Type the password of the database user Crowd will use to log in to your Bamboo instance's database.

     The import process will log in to the database, not into Bamboo.
7. Click the '**Continue**' button to import the users from your Bamboo instance into your Crowd directory.
8. The '**Results**' screen will be displayed, showing how many users and groups have been imported into your Crowd directory.
9. Click the '**Users**' button to view and manage the imported users and groups via the Crowd Administration Console (assuming the directory's permissions allow this).

Screenshot: 'Import Bamboo Users'



Next Step

To give the imported groups access to the Bamboo application, see Specifying which Groups can access an Application.

Related Topics

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory

- Fedora Directory Server
- Generic LDAP Directories
- Microsoft Active Directory
  - Configuring an SSL Certificate for Microsoft Active Directory
- Novell eDirectory
- OpenDS
- OpenLDAP
- OpenLDAP Using Posix Schema
- Posix Schema for LDAP
- Sun Directory Server Enterprise Edition (DSEE)

- Configuring a Custom Directory Connector
- Configuring a Delegated Authentication Directory

- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Crowd Documentation

Importing Users from Jive Forums

If you have already been using Jive Forums, and are now configuring Jive Forms as a Crowd application, you will probably want to import your existing Jive users and groups into a Crowd directory.

Before you begin:
The database drivers for the Jive Forums database will need to be on Crowd's classpath. To do this, simply copy the database driver JAR for your particular Jive database across to `CROWD/apache-tomcat/common/lib` and restart Crowd.

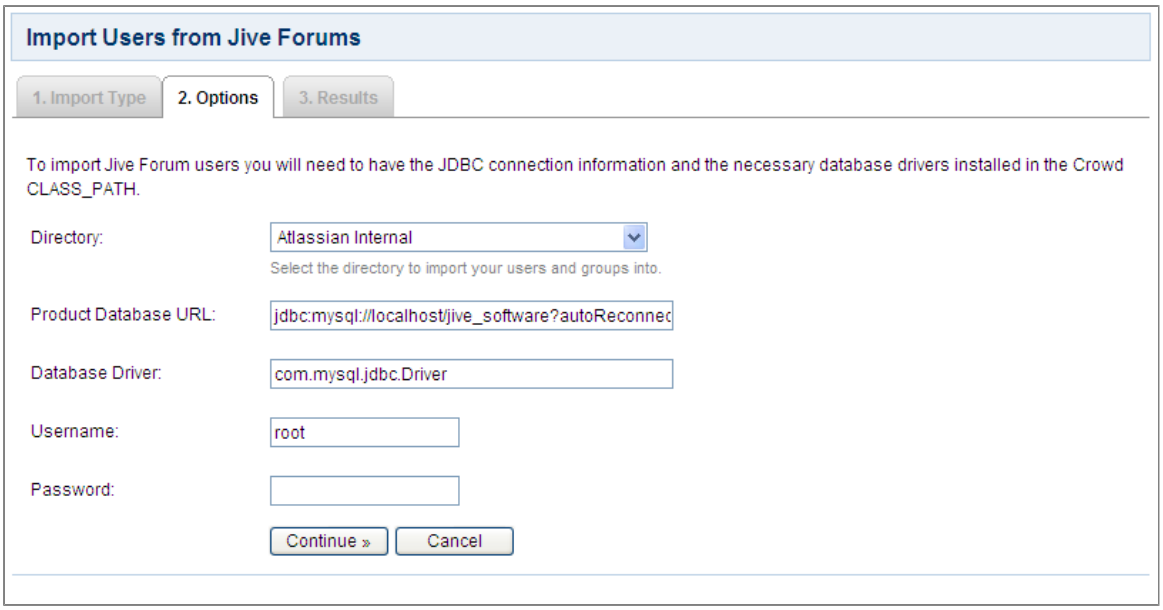
Note: the passwords for users in Jive will not be copied across to Crowd as they are stored as hashes in Jive's internal database.

To import users and groups from Jive Forums into a Crowd directory,

1. Login to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Click the '**Import Users**' link.
4. This will display the '**Import Type**' screen. Click the '**JIVE**' button.
5. This will display the '**Options**' screen. Complete the fields as follows:
   - '**Directory**' — select the directory that is mapped to the Jive Forums application.
   - '**DB URL**' — type the URL of Jive's database.
   - '**DB Driver**' — type the name of Jive's database JDBC driver.
   - '**Username**' — type the username of the database user that Crowd will use to login to Jive's database.
   - '**Password**' — type the password of the database user Crowd will use to login to Jive's database.

   The import process will log in to the database, not to Jive Forums.
6. Click the '**Continue**' button to import the users from Jive Forums into your Crowd directory.
7. The '**Status**' screen will be displayed, showing how many users and groups have been imported into your Crowd directory.
8. Click the '**Users**' button to view and manage the imported users and groups via the Crowd Administration Console (assuming the directory's permissions allow this).

Screenshot: 'Import Jive Users'



Next Step

To give the imported groups access to the Jive Forums application, see Specifying which Groups can access an Application.

Related Topics

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - Apache Directory Server (ApacheDS)
    - Apple Open Directory
    - Fedora Directory Server
    - Generic LDAP Directories
    - Microsoft Active Directory
      - Configuring an SSL Certificate for Microsoft Active Directory
    - Novell eDirectory
    - OpenDS
    - OpenLDAP
    - OpenLDAP Using Posix Schema
    - Posix Schema for LDAP
    - Sun Directory Server Enterprise Edition (DSEE)
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
  - Importing Users from Atlassian Confluence
  - Importing Users from Atlassian JIRA
  - Importing Users from Atlassian Bamboo
  - Importing Users from Jive Forums
  - Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
  - Importing Users from One Crowd Directory into Another

Crowd Documentation

Importing Users from CSV Files

You can copy users from an external directory or user base into Crowd via a CSV (comma-separated values) file. There are two phases involved:

1. Export your existing users and their group memberships from your external directory into a CSV file or files.
2. Import the users, groups and group memberships into a Crowd directory from the CSV files.

The CSV importer is available with Crowd 1.1.1 and later.

Preparing your CSV Files

You will need:

- a CSV file containing user information, and
- optionally, another CSV file containing group memberships.

Attached are simple examples of the CSV files:

- Example user CSV file
- Example group membership CSV file

The CSV Importer's 'File Mappings' screen allows you to match the CSV fields to Crowd's User and Group fields.

Formatting and location of the CSV files:

Requirement	Description
Location	The CSV files must be on the local drive (e.g. C:) of the Crowd server.
Supported attributes	The CSV Importer does not support custom attributes. The supported attributes are shown in the drop-down lists on the 'File Mappings' screen.
Header row	The first row in each CSV file must be a header row. The CSV Importer will not import the information in the first row. The information in the first row is displayed in the column labelled 'CSV Header Row' on the 'File Mappings' screen
Delimiter	The fields in the CSV file must be separated by a single-character delimiter. The CSV Importer's 'Configuration' screen lets you tell Crowd which delimiter you have used.
Passwords	You will need to decide whether to import your passwords into Crowd. And if you do import the passwords, you must choose to import them as either encrypted or clear text. Check the password encryption in the directory you are exporting users from, and compare it with the encryption method of the Crowd directory you want to import the users into. You can use Crowd's Directory Browser to view the directory's configuration details, including the encryption method. The CSV Importer's 'Configuration' screen lets you tell the CSV Importer whether to encrypt the passwords.

To export information from your user directory into a CSV file,

1. Export the users from your external user directory or database into a CSV file. Your directory or user base should have an option to allow you to do this.
2. If you want to copy your existing group memberships into Crowd, export the groups and group memberships into another CSV file.

Importing the CSV Files into Crowd

Once you have prepared your CSV file(s), you can import the users and groups into a Crowd directory.

To import users and groups from CSV files,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Click the '**Import Users**' link.
4. This will display the '**Import Type**' screen. Click the '**CSV Importer**' button.
5. This will display the '**Configuration**' tab of the '**CSV Importer**'.
6. Enter the details of the CSV files as described in 'Configuring the CSV Importer'.

RELATED TOPICS

- Configuring the CSV Importer
- Mapping CSV Fields to Crowd Fields
- Confirming the CSV Importer Configuration
- Viewing the Results of the Import

Crowd Documentation

Configuring the CSV Importer

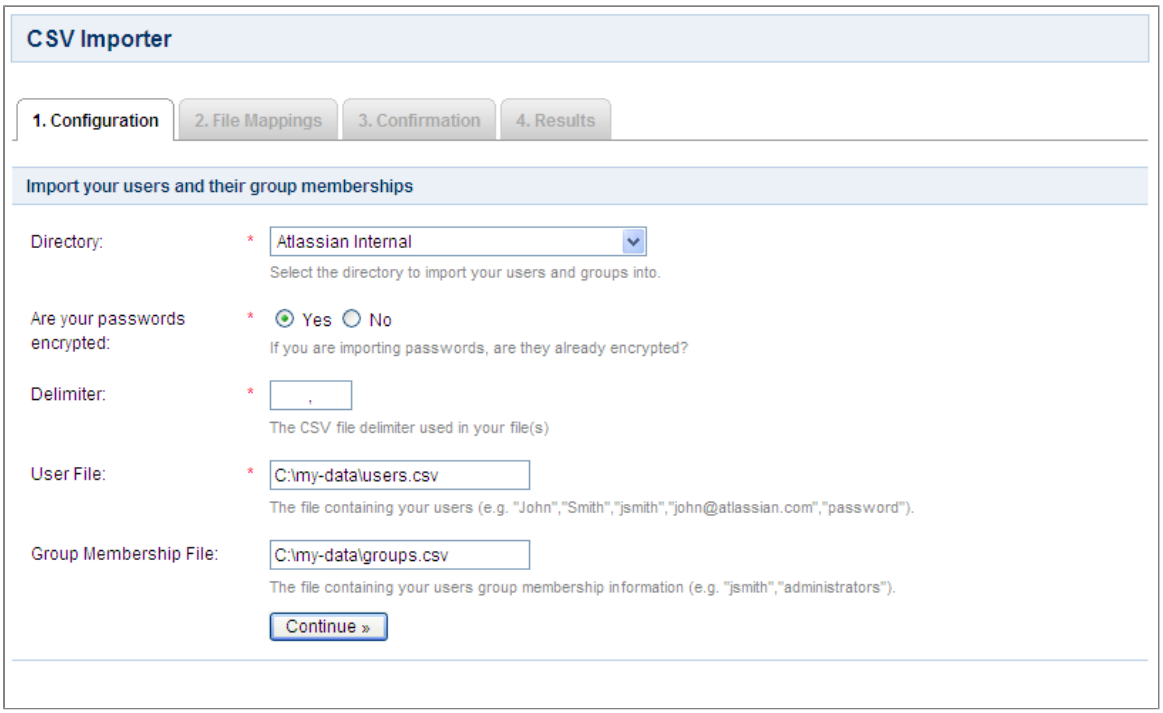
Once you have started the CSV Importer, the '**Configuration**' screen allows you to specify information about the Crowd directory and CSV file(s) involved in the import.

Refer to information on preparing your CSV files.

To configure the CSV importer,

1. Start the CSV Importer.
2. This will display the '**Configuration**' screen. Complete the fields as follows:
    - '**Directory**' — Select the Crowd user directory into which you want to import the users.
    - '**Are your passwords encrypted?**' — Select 'Yes' if the passwords in your CSV file are already encrypted. Crowd will not re-encrypt the passwords during the import. Select 'No' if the passwords in your CSV file are not encrypted. Crowd will encrypt the passwords during the import, using the encryption method of the Crowd directory you are importing into.
    - '**Delimiter**' — Type the single-character delimiter used to separate the fields in your CSV file(s).
    - '**User File**' — Type the location of the CSV file containing the users you wish to import.
    - '**Group Membership File**' — If you want to import groups and group memberships of your users, type the location of the CSV file containing the group membership information.
3. Click the '**Continue**' button to map the CSV fields to the Crowd directory fields.

Screenshot: 'CSV Importer - Configuration'



RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
    - Configuring an Internal Directory
    - Configuring an LDAP Directory Connector
        - Apache Directory Server (ApacheDS)
        - Apple Open Directory
        - Fedora Directory Server
        - Generic LDAP Directories
        - Microsoft Active Directory
            - Configuring an SSL Certificate for Microsoft Active Directory
        - Novell eDirectory
        - OpenDS
        - OpenLDAP
        - OpenLDAP Using Posix Schema
        - Posix Schema for LDAP
        - Sun Directory Server Enterprise Edition (DSEE)
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory
    - Importing Users from Atlassian Confluence
    - Importing Users from Atlassian JIRA
    - Importing Users from Atlassian Bamboo
    - Importing Users from Jive Forums
    - Importing Users from CSV Files
        - Configuring the CSV Importer
        - Mapping CSV Fields to Crowd Fields

- Confirming the CSV Importer Configuration
- Viewing the Results of the Import
- Importing Users from One Crowd Directory into Another

Crowd Documentation

Mapping CSV Fields to Crowd Fields

Once you have entered details on the Configuration screen of the CSV Importer, the '**File Mappings**' screen allows you to match the CSV fields to the User and Group fields in Crowd. Crowd will use these mappings to import the information from the CSV file(s) into your Crowd directory.

Refer to information on preparing your CSV files.

The '**File Mappings**' screen has two main sections:

- '**User Mappings**' — Use this section to map the fields in your 'User' CSV file.
- '**Group Mappings**' — Use this section to map the fields in your 'Group Membership' CSV file, if you have one. This section will only appear if you have specified a 'Group Membership File' on the Configuration screen.

Each section has the following columns:

Column	Description
CSV Header Row	This column shows the text from each field in the first row of your CSV file. The CSV Importer assumes that the first row is a header row.
Sample Row	This column shows the text from each field in the second row of your CSV file. This is done to help you with the mapping process.
Mapping	Each row in this column contains a drop-down list of the Crowd field names available for mapping. To map a Crowd field to a CSV field, select the appropriate Crowd field name from the drop-down list to match the CSV field shown in the 'CSV Header Row' column.

In the '**User Mappings**' section, the 'Mapping' drop-down lists contain the following Crowd field names:

Crowd field	Description
First Name	Required. One of the rows on the screen must map this value to the CSV field containing the users' first names.
Last Name	Required. One of the rows on the screen must map this value to the CSV field containing the users' last names.
Email Address	Required. One of the rows on the screen must map this value to the CSV field containing the users' email addresses.
Username	Required. One of the rows on the screen must map this value to the CSV field containing the usernames.
Password	If your CSV file contains passwords, map this value to the CSV field containing the passwords.
None	Select 'None' if the CSV field displayed under 'CSV Header Row' is not to be mapped to any Crowd fields. These CSV fields will not be imported into Crowd.

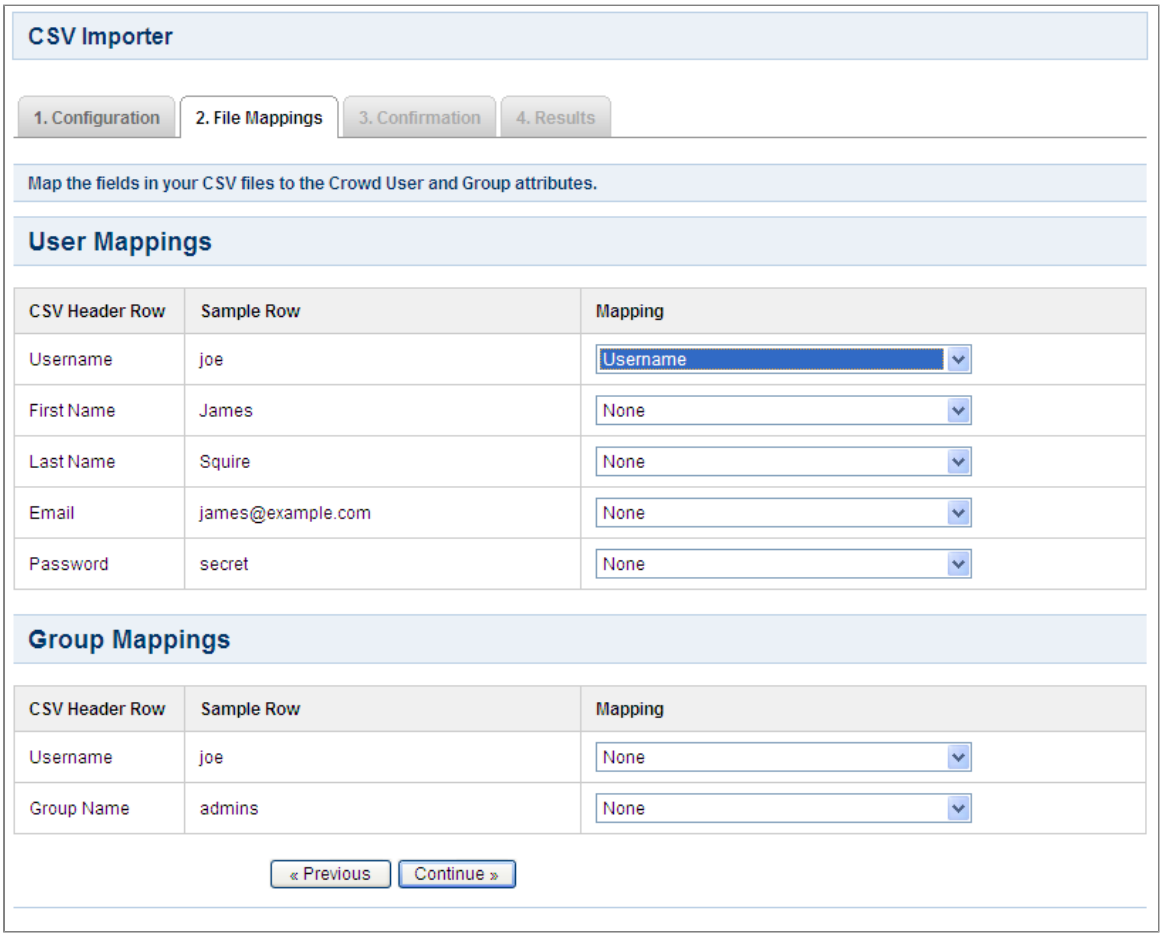
In the '**Group Mappings**' section (if present), the 'Mapping' drop-down lists contain the following Crowd field names:

Crowd field	Description
Group Name	Required. One of the rows on the screen must map this value to the CSV field containing the names of the groups.
Username	Required. One of the rows on the screen must map this value to the CSV field containing the usernames.
None	Select 'None' if the CSV field displayed under 'CSV Header Row' is not to be mapped to any Crowd fields. These CSV fields will not be imported into Crowd.

To map the CSV fields to Crowd fields,

1. Start the CSV Importer.
2. Complete the details on the 'Configuration screen' and click the 'Continue' button.
3. This will display the '**File Mappings**' screen. Complete the mappings in the '**User Mappings**' section as follows:
    - In the 'CSV Header Row' column, find the field which contains your users' first names — select '**First Name**' from the drop-down list in the 'Mapping' column.
    - In the 'CSV Header Row' column, find the field which contains your users' last names — select '**Last Name**' from the drop-down list in the 'Mapping' column.
    - In the 'CSV Header Row' column, find the field which contains your users' email addresses — select '**Email Address**' from the drop-down list in the 'Mapping' column.
    - In the 'CSV Header Row' column, find the field which contains the usernames — select '**Username**' from the drop-down list in the 'Mapping' column.
    - In the 'CSV Header Row' column, find the field which contains your users' passwords — select '**Password**' from the drop-down list in the 'Mapping' column.
    - Select '**None**' from the drop-down lists for all unmatched rows.
4. Complete the mappings in the '**Group Mappings**' section (if present) as follows:
    - In the 'CSV Header Row' column, find the field which contains the group names — select '**Group Name**' from the drop-down list in the 'Mapping' column.
    - In the 'CSV Header Row' column, find the field which contains the usernames — select '**Username**' from the drop-down list in the 'Mapping' column.
    - Select '**None**' from the drop-down lists for all unmatched rows.
5. Click the '**Continue**' button to confirm the CSV configuration.

Screenshot: 'CSV Importer - File Mappings'



RELATED TOPICS

- Configuring the CSV Importer
- Mapping CSV Fields to Crowd Fields
- Confirming the CSV Importer Configuration
- Viewing the Results of the Import

Crowd Documentation

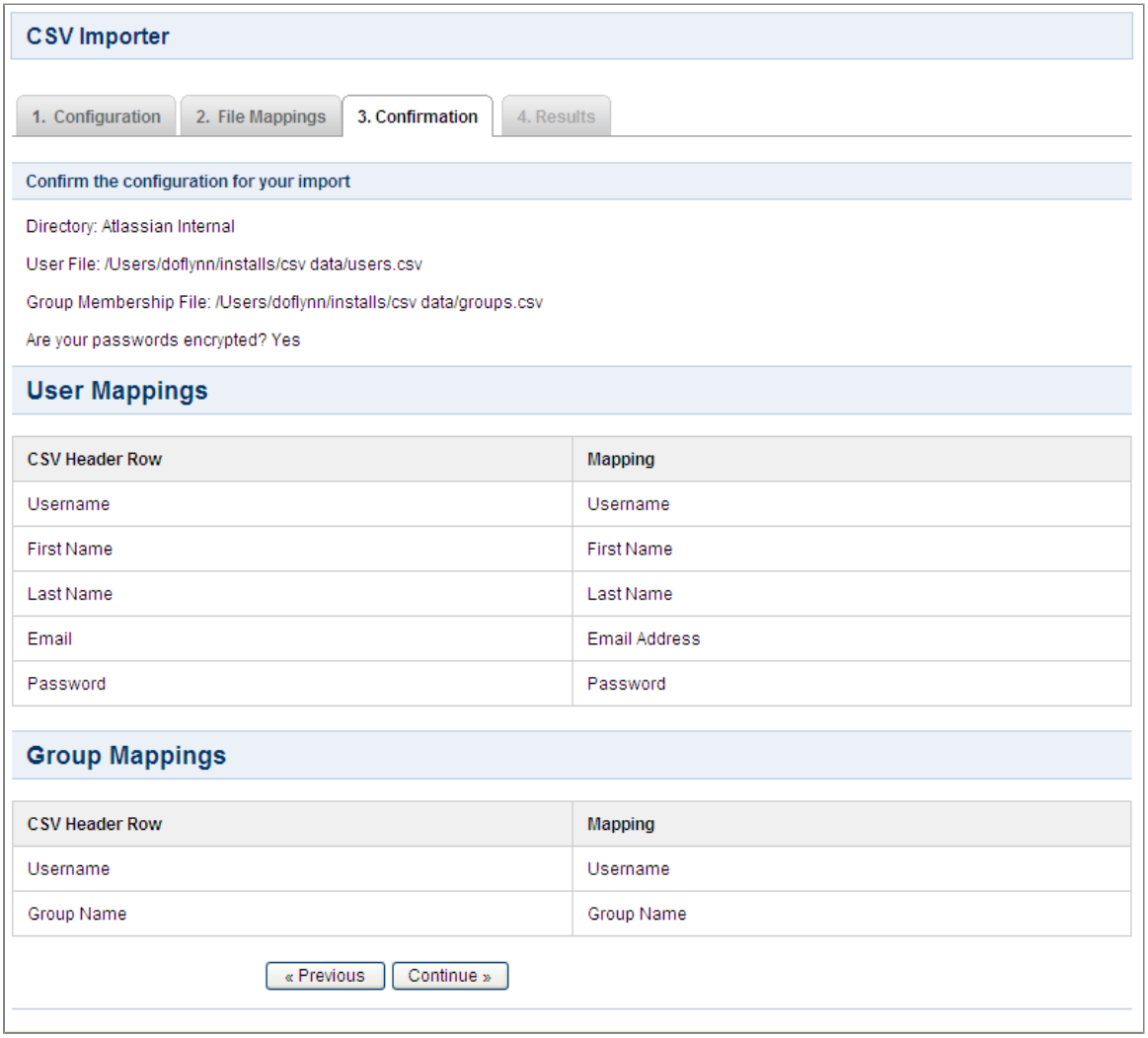
Confirming the CSV Importer Configuration

The '**Confirmation**' screen allows you to review your configuration and mapping before performing the CSV import.

To confirm the CSV configuration and mapping,

1. Review the information shown on the 'Confirmation' screen.
2. Click the '**Continue**' button to import the users from your CSV file into your Crowd directory.
3. Once the import is complete, Crowd will display the 'Results' screen.

Screenshot: 'CSV Importer - Confirmation'



RELATED TOPICS

- Configuring the CSV Importer
- Mapping CSV Fields to Crowd Fields
- Confirming the CSV Importer Configuration
- Viewing the Results of the Import

Crowd Documentation

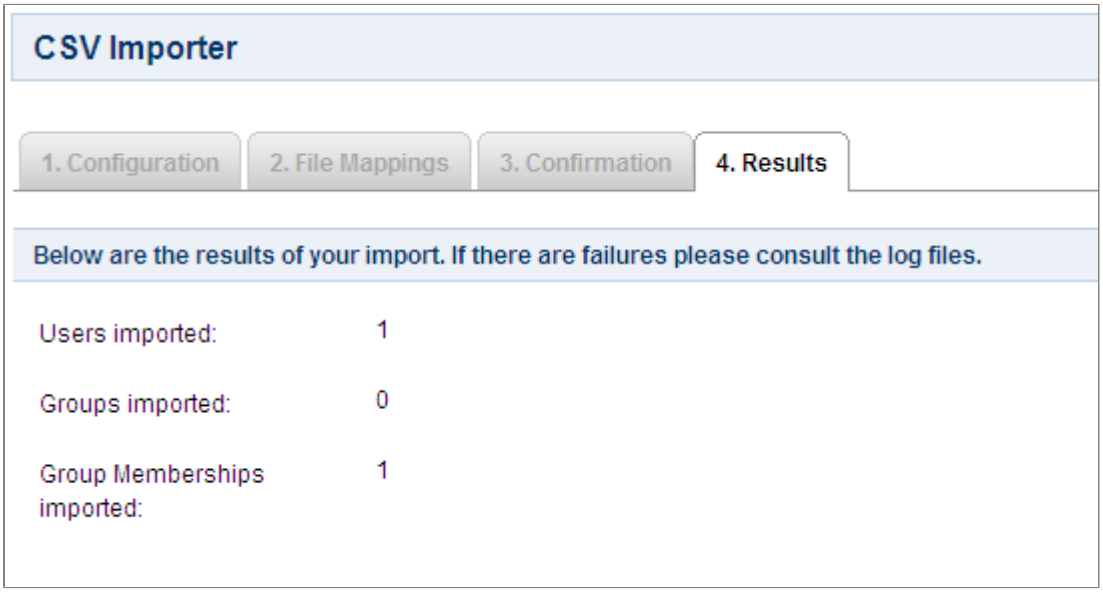
Viewing the Results of the Import

The '**Results**' screen shows the outcome of the CSV import.

The CSV Importer **adds** to the Crowd directory, but does not update or delete existing information:

- If the Username already exists in Crowd, the CSV Importer does not overwrite the information for that user even if the Username exists in the CSV file with different user information.
- The CSV Importer does not remove users from Crowd.
- If your 'Group Membership' CSV file contains additional group(s) for a user, the additional group(s) and group membership(s) will be imported.
- Existing group memberships will not be changed or removed.
- The 'Results' screen will show number of duplicate usernames in the CSV file which were ignored i.e. not imported.
- The 'Results' screen will show number of duplicate group names in the CSV file which were ignored i.e. not imported.

Screenshot: 'CSV Importer - Results'



RELATED TOPICS

- Configuring the CSV Importer
- Mapping CSV Fields to Crowd Fields
- Confirming the CSV Importer Configuration
- Viewing the Results of the Import

Crowd Documentation

Importing Users from One Crowd Directory into Another

Once you have added a directory, you can import users, groups and roles into it from an external system or from another directory defined in Crowd. To learn about importing from external systems, refer to Importing Users and Groups into a Directory. Below we tell you how to import from one Crowd directory to another.

You can copy users, groups, roles and memberships:

- From an LDAP directory to a Delegated Authentication directory.
- From one internal Crowd directory to another internal Crowd directory.

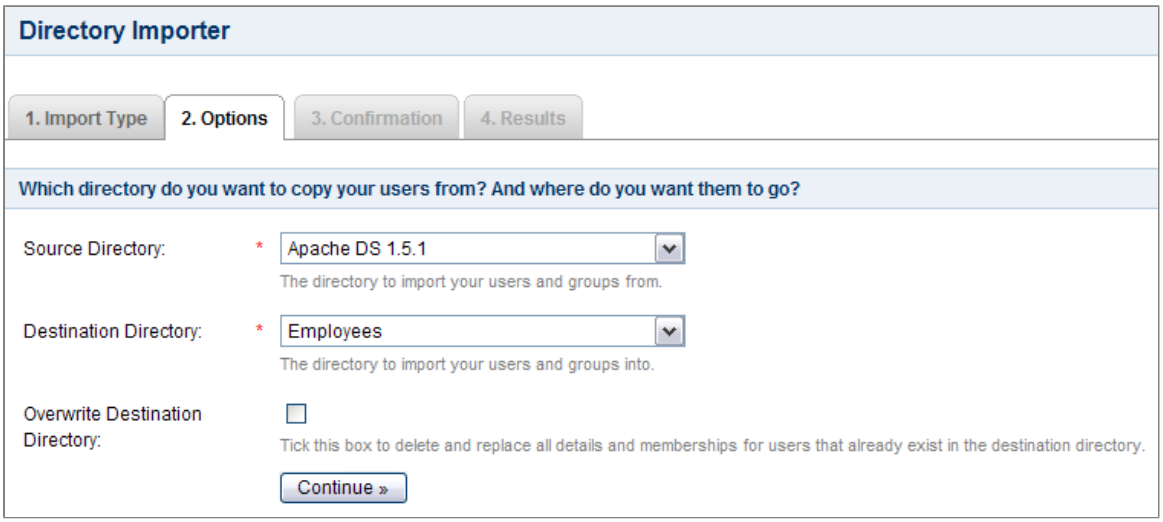
Things to be aware of:

- The '**Password Encryption**' method must be the same in both directories, otherwise you will not be able to copy the users across.
- The directory importer does not support nested groups when importing users, groups and roles from LDAP into a delegated authentication directory. See CWD-1334.
- The '**source directory**' is the directory you want to copy users, groups and roles from. The '**destination directory**' is where you want to copy them to. Both directories must be defined in Crowd before you start the import process.

To import users, groups and roles from one Crowd directory into another,

1. Log in to the Crowd Administration Console.
2. If not already defined, add the source directory to Crowd.
3. If not already defined, add the destination directory to Crowd.
4. Click the '**Users**' link in the top navigation bar.
5. This will display the User Browser. Click the '**Import Users**' link.
6. This will display the '**Import Type**' screen. Click the '**Directory Importer**' button.
7. This will display the '**Options**' screen, shown below. Complete the fields as follows:
    - '**Source Directory**' — Select the directory that contains the users, groups and roles you want to copy.
    - '**Destination Directory**' — Select the directory that you want to copy the users, groups and roles into.
    - '**Overwrite Destination Directory**' — Tick the box if you want to delete and replace all the details and memberships for any user who exists in both source and destination directories:
        - If the checkbox is empty, Crowd will not update the user details for that username in the destination directory, but will add any new group or role memberships for that username.
        - If the checkbox is ticked, Crowd will remove all the details and memberships for that username from the destination directory and replace them with the details and memberships from the source directory.
8. Click the '**Continue**' button.
9. The '**Confirmation**' screen will be displayed. Check the details and click the '**Continue**' button.
10. The '**Results**' screen will be displayed, showing how many users, groups and roles have been imported into your Crowd directory.
    If the import of any users, groups or roles failed, please check the log files to find out why.

Screenshot: 'Import users from one directory to another'



Next Steps

To allow the users to log in to the integrated application(s) via Crowd:

- Map the directory to the application(s), if not already done. See Mapping a Directory to an Application.
- Give the imported groups access to the application(s). See Specifying which Groups can access an Application.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
    - Configuring an Internal Directory
    - Configuring an LDAP Directory Connector
        - Apache Directory Server (ApacheDS)
        - Apple Open Directory
        - Fedora Directory Server
        - Generic LDAP Directories
        - Microsoft Active Directory
            - Configuring an SSL Certificate for Microsoft Active Directory
        - Novell eDirectory
        - OpenDS
        - OpenLDAP
        - OpenLDAP Using Posix Schema
        - Posix Schema for LDAP
        - Sun Directory Server Enterprise Edition (DSEE)
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Configuring Relaxed DN Standardisation
- Specifying Directory Permissions
- Importing Users and Groups into a Directory

- Importing Users from Atlassian Confluence
- Importing Users from Atlassian JIRA
- Importing Users from Atlassian Bamboo
- Importing Users from Jive Forums
- Importing Users from CSV Files
    - Configuring the CSV Importer
    - Mapping CSV Fields to Crowd Fields
    - Confirming the CSV Importer Configuration
    - Viewing the Results of the Import
- Importing Users from One Crowd Directory into Another

Crowd Documentation

Managing Applications

Crowd integrates and provisions applications. Once defined, an application is mapped to a directory(s), whose users are then granted access to the application. Note that an application can only communicate with Crowd when the application uses a known host address.

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Using the Application Browser

This page describes the Application Browser and gives an overview of the types of application you may find in Crowd.

On this page:

- About the Application Browser
- About Applications
    - Default Applications
    - Application Types

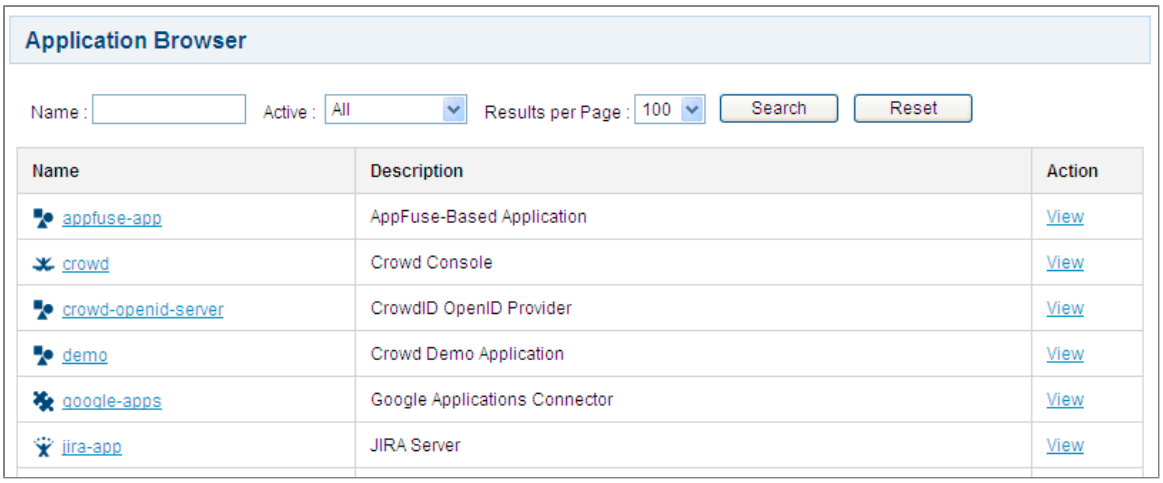
About the Application Browser

The Application Browser allows you to view and search for integrated applications.

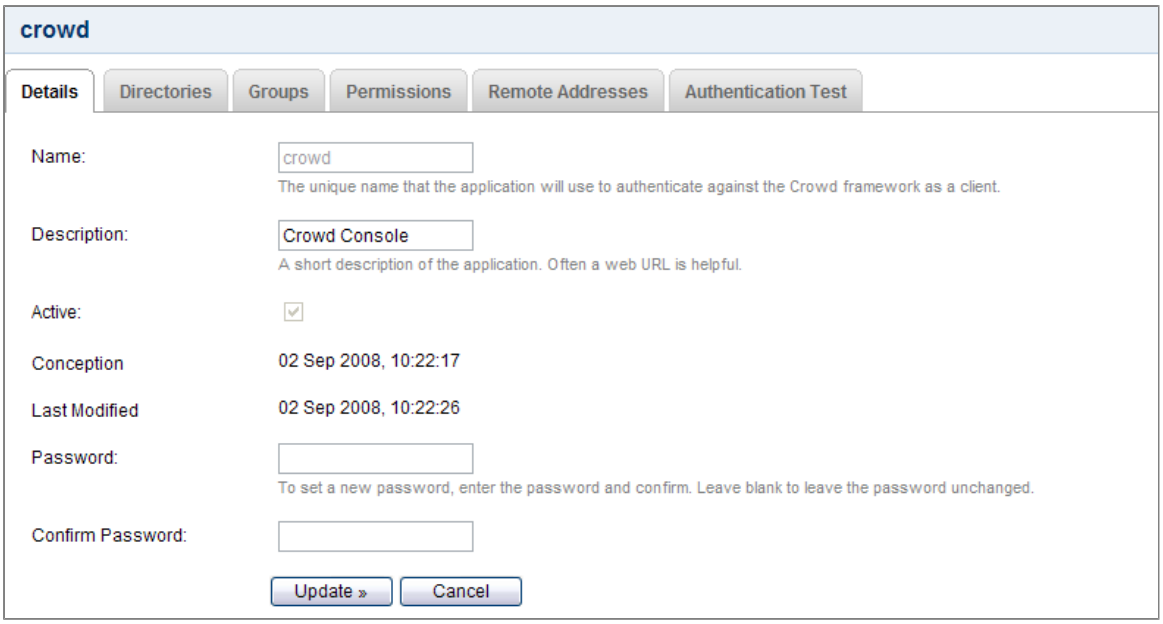
To use the Application Browser,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. This will display the Application Browser, showing all the applications that exist in your Crowd system. You can refine your search by specifying a '**Name**' (note that this is case sensitive), or '**Active**'/'**Inactive**' applications.
4. To view or edit an application's details, click the application name or the '**View**' link next to the specific application.

Screenshot: Application Browser



Screenshot: Example of an application's details



About Applications

Crowd integrates and provisions applications. Once defined, an application is mapped to a directory(s), whose users are then granted access to the application. Note that an application can only communicate with Crowd when the application uses a known host address.

Default Applications

When you first use the Application Browser, you will see a number of default applications, i.e. applications that are shipped with your Crowd installation:

- '**crowd**' — This is the Crowd Administration Console. The Crowd Administration Console is itself a web application that is provisioned by Crowd. The 'crowd' application is mapped to the default directory which you defined during setup, and can be accessed by members of the `crowd-administrators` group.
- '**crowd-openid-server**' — This is the CrowdID application which you (optionally) configured during setup. It allows you to provide OpenID services to your users. For details please see the *CrowdID Administration Guide* and the *CrowdID User Guide*.
  The page How CrowdID works with Crowd does not exist.
- '**demo**' — This is the 'demo' application which you (optionally) configured during setup. Its main purpose is to provide an example of

how to integrate custom applications with Crowd.
The page Sample Application ('demo') does not exist.
- '**google-apps**' — This is the Crowd application connector which allows single sign-on (SSO) to Google Apps. To enable SSO between Crowd-connected applications and Google Apps, you will need to configure the Google Apps connector as described in Configuring the Google Apps Connector.

Application Types

Crowd supports the following application types, as indicated by the application-type icons on the Application Browser:

	This icon marks the Crowd application. • There will be one, and only one, application of this type. • You cannot rename, deactivate or delete this application.
	This marks a Bamboo server connected to Crowd.
	This marks a Confluence server connected to Crowd.
	This marks a Crucible server connected to Crowd.
	This marks a Fisheye server connected to Crowd.
	This marks a JIRA server connected to Crowd.
	These are the 'remote' applications, which you can add to Crowd as described in Adding an Application. This application type does not include plugin applications. You can rename, deactivate or delete remote applications.
	The 'plugin' applications are implemented as plugins to Crowd. • An example of a plugin application is the Google Apps connector, which is shipped with your Crowd installation. To activate the Google Apps connector, you need to configure it. • In future, other plugin applications may become available. You will then be able to install them by copying the relevant jars to your Crowd installation. See Important Directories and Files. • All installed plugin applications are created automatically when the Crowd server starts up, by loading them from the relevant folders in your Crowd Home directory. • You cannot rename or delete plugin applications. You can deactivate them.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Adding an Application

This page gives an overview of the process for adding an application to Crowd, and refers to the application-specific pages for detailed instructions.

Overview

There are two main steps to integrating an application with Crowd:

- **Step 1. Configure Crowd to talk to your application** — that is, set up a directory in Crowd containing your users and groups, and then add the application to Crowd using the 'Add Application' wizard, as described below. The application will now be allowed to authenticate against Crowd.
- **Step 2. Configure the application to talk to Crowd** — that is, install the Crowd client into the application and configure the application to forward users' authentication and security requests to Crowd.

Detailed Instructions

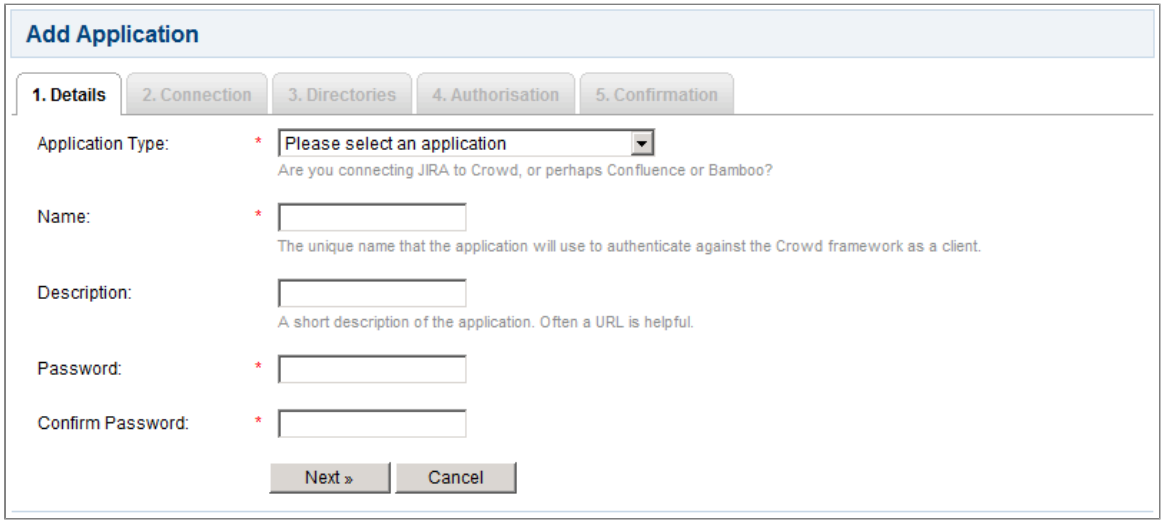
Please refer to the details for your specific application:

- Integrating Crowd with Atlassian Bamboo
- Integrating Crowd with Atlassian Confluence
- Integrating Crowd with Atlassian CrowdID
- Integrating Crowd with Atlassian Crucible
- Integrating Crowd with Atlassian FishEye
- Integrating Crowd with Atlassian JIRA
- Integrating Crowd with Acegi Security
- Integrating Crowd with Apache
- Integrating Crowd with Jive Forums
- Integrating Crowd with Spring Security
- Integrating Crowd with Subversion
- Integrating Crowd with a Custom Application

Using Crowd's 'Add Application' Wizard

Before you start
Before adding the application, consider whether you need to add your directories, users and groups. See the detailed instructions for your application.

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. This will display the Application Browser. Click '**Add Application**' in the left-hand menu.
4. This will display the first screen for the '**Add Application**' wizard for Crowd. Complete the fields as described in the table below.



Attribute	Description
Application Type	This is used to define the type of application you are adding to Crowd. If you cannot see a matching application type, please choose the 'Generic Application' option.
Name	The username which the application will use when it authenticates against the Crowd framework as a client. This value must be unique, i.e. it cannot be used by more than one application client.
Description	A short description of the application. Note: A URL is often helpful.
Password	The password which the application will use when it authenticates against the Crowd framework as a client.
Confirm Password	Retype the same password as above, to confirm it.

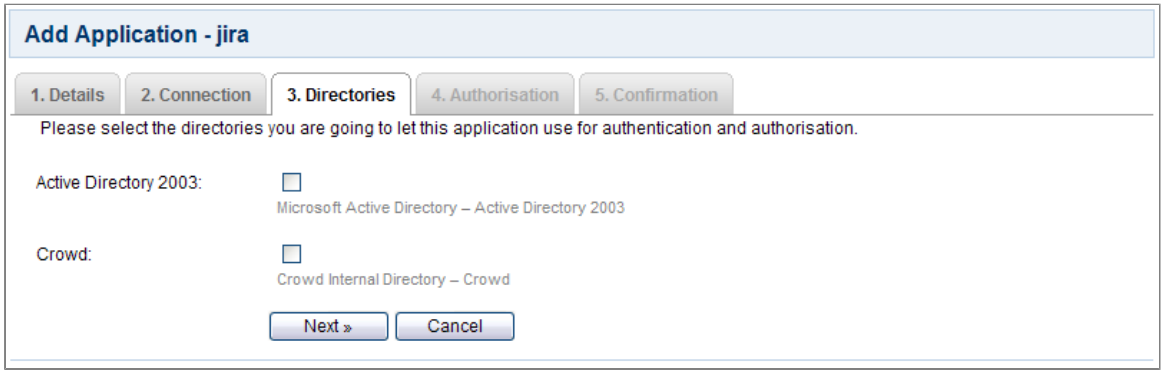
After completing this form, click the 'Next' button to go to the 'Connection' step.

5. Enter the connection details for your application, as described in the table below.

Attribute	Description
URL	The URL of your application. For example this may be http://jira.atlassian.com. After entering the URL for the application, you can click the '**Resolve IP Address**' button. Crowd will attempt to resolve the IP address for your application.
Remote IP Address	This is the IP address of the server where your application exists. To help you work this out, you can click the '**Resolve IP Address**' button once you have entered a URL.

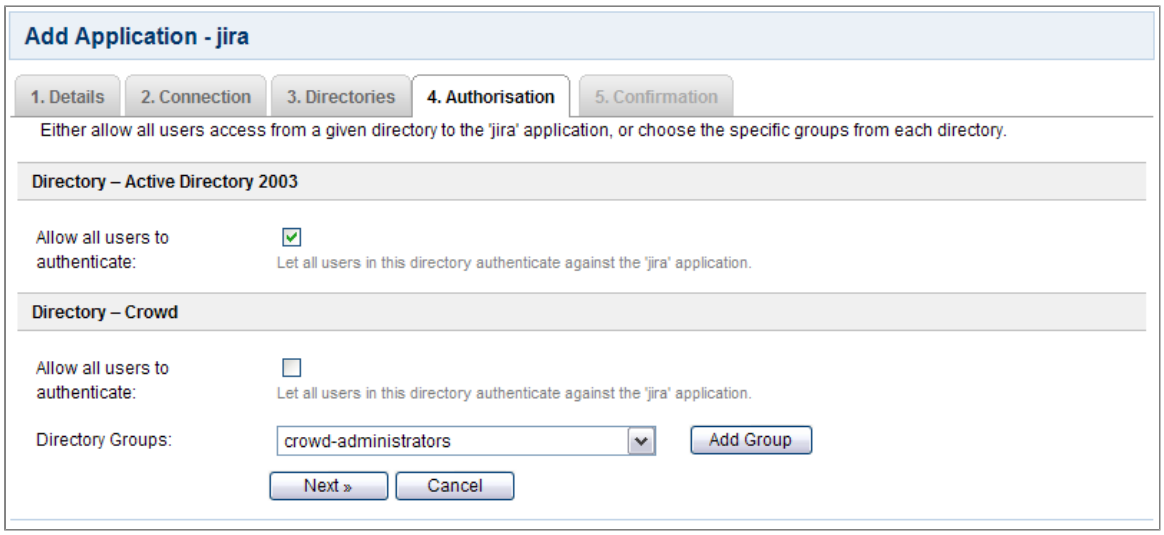
After completing this form, click the 'Next' button to go to the 'Directories' step.

6. Now select the directories that this application can use for authentication and authorisation:



Click the relevant checkbox(es) to select one or more directories.
After completing this form, click the 'Next' button to go to the 'Authorisation' step.

7. In the 'Authorisation' step you will determine the users who are authorised to access the application.



For each directory, you should do one of the following:

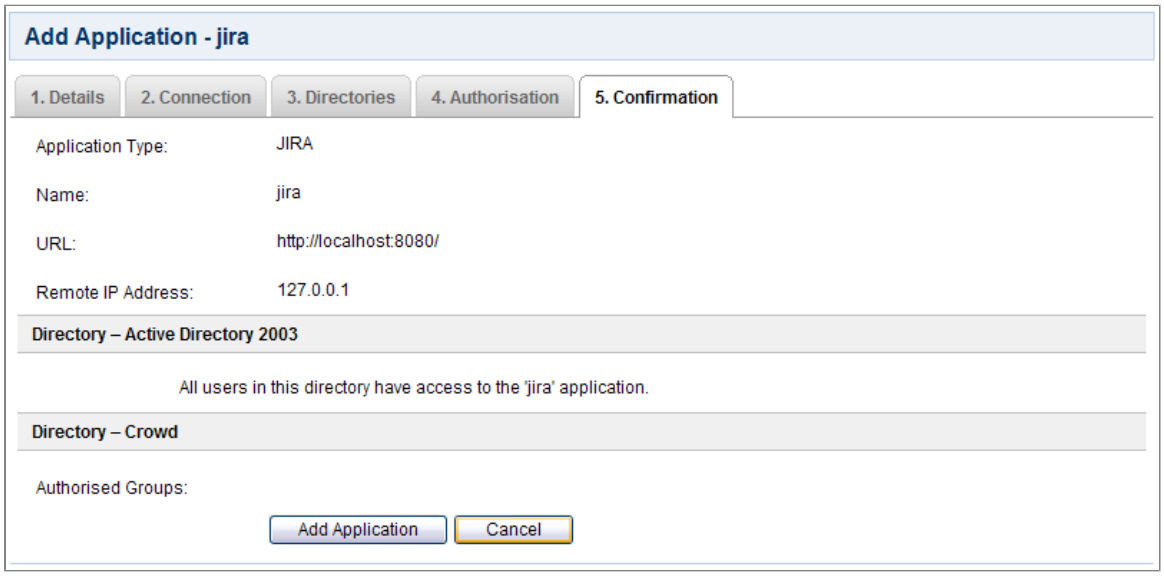
- Either select '**Allow all users to authenticate**', to grant application access to all users defined in the directory.
- Or select one or more groups you wish to have access, and click '**Add Group**' to add each group to the list. The 'Add Group' button appears when you select a group.

  To remove a group from the list after adding it, click the '**remove**' link that will appear next to the authorised groups'

names.

After completing this form, click the 'Next' button to go to the 'Confirmation' step.

8. Now confirm the details for your application.



Check the details of your application.
- If you need to change anything, you can click the tabs to go back to one of the steps in the 'Add Application' wizard.
- When you are happy with the details, click the '**Add Application**' button

You will now be on the 'View Application' page where you can adjust most of the options you have selected during the creation process.

9. After completing the 'Add Application' wizard, remember to configure the application as described in the detailed instructions:
   - Integrating Crowd with Atlassian Bamboo
   - Integrating Crowd with Atlassian Confluence
   - Integrating Crowd with Atlassian CrowdID
   - Integrating Crowd with Atlassian Crucible
   - Integrating Crowd with Atlassian FishEye
   - Integrating Crowd with Atlassian JIRA
   - Integrating Crowd with Acegi Security
   - Integrating Crowd with Apache
   - Integrating Crowd with Jive Forums
   - Integrating Crowd with Spring Security
   - Integrating Crowd with Subversion
   - Integrating Crowd with a Custom Application

Community application connectors
You may also be interested in the Crowd plugins created by community developers. (Please check under '**Plugin Details**' for each plugin to see if the plugin is supported by Atlassian.)

RELATED TOPICS

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with Atlassian Bamboo

This page tells you how to connect Atlassian's Bamboo integration server to one or more directory servers through Crowd.

Currently Crowd supports centralised authentication and single sign-on for Bamboo versions 1.2.2 and later.

> **Please check that this documentation applies to your version of Crowd**
> Please check the Crowd release number in this documentation against your version of Crowd. If you are using a different version of Crowd, you can find the appropriate documentation under 'Previous Versions' on the Crowd documentation homepage.

Prerequisites

> Due to incompatible atlassian-user libraries, Bamboo releases prior to 1.2.2 are not compatible with latest version of Crowd. Please upgrade to the latest version of Bamboo before attempting to integrate Crowd.

1. Download and install Crowd. Refer to the Crowd installation guide for instructions. We will refer to the Crowd root folder as `CROWD`.
2. Download and install Bamboo (version 1.2.2 or later). Refer to the Bamboo Installation Guide for instructions. We will refer to the Bamboo root folder as `BAMBOO`. For the purposes of this document, we will assume that you have used the Standalone (ie. the easier) installation method of Bamboo. If you need to install Bamboo as an EAR/WAR, simply explode the EAR/WAR and make the necessary changes as described below, then repackage the EAR/WAR.
3. Run the Bamboo Setup Wizard, as described in the Bamboo documentation. During this setup process, you will define the Bamboo administrator's username and password. It is easier to do this before you integrate Bamboo with Crowd.
4. After you have installed and set up Bamboo, shut Bamboo down before you begin the integration process described below.

Step 1. Configuring Crowd to Talk to Bamboo

1.1 Prepare Crowd's Directories/Groups/Users for Bamboo

1. **Create a Crowd directory:** The Bamboo application will need to authenticate users against a directory configured in Crowd. You will need to set up a directory in Crowd for Bamboo. For more information on how to do this, see Adding a Directory. We will assume that the directory is called *Crowd Bamboo Directory* for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use *Crowd Bamboo Directory* to house Bamboo users.

2. **Add users and groups:** You can either import them from your Bamboo deployment or add them manually.
   - **Importing users and groups from Bamboo:** If you have an existing Bamboo deployment and would like to import existing users and groups into Crowd, use the Bamboo Importer tool by navigating to **Users** > **Import Users** > **Atlassian Importer**. Select 'Bamboo' as the Atlassian Product and the *Crowd Bamboo Directory* as the directory into which Bamboo users will be imported. For details please see Importing Users from Atlassian Bamboo. If you are going to import users into Crowd, you need to do this now, before you proceed any further.
   - **Adding users and groups manually:** Bamboo needs an administrative group to exist in the directory in order to access the administration features. You can also create an optional additional group for other users. Create the groups in the *Crowd Bamboo Directory*:
     - `bamboo-admin`
     - `bamboo-user` (*optional*)
       See the documentation on Creating Groups for more information on how to define these groups.

     - Create at least one user in the *Crowd Bamboo Directory* and assign the user(s) to both the `bamboo-user` and the `bamboo-admin` groups. The Crowd documentation has more information on creating groups, creating users and assigning users to groups.

1.2 Define the Bamboo Application in Crowd

Crowd needs to be aware that the Bamboo application will be making authentication requests to Crowd. We need to add the Bamboo application to Crowd and map it to the *Crowd Bamboo Directory:*

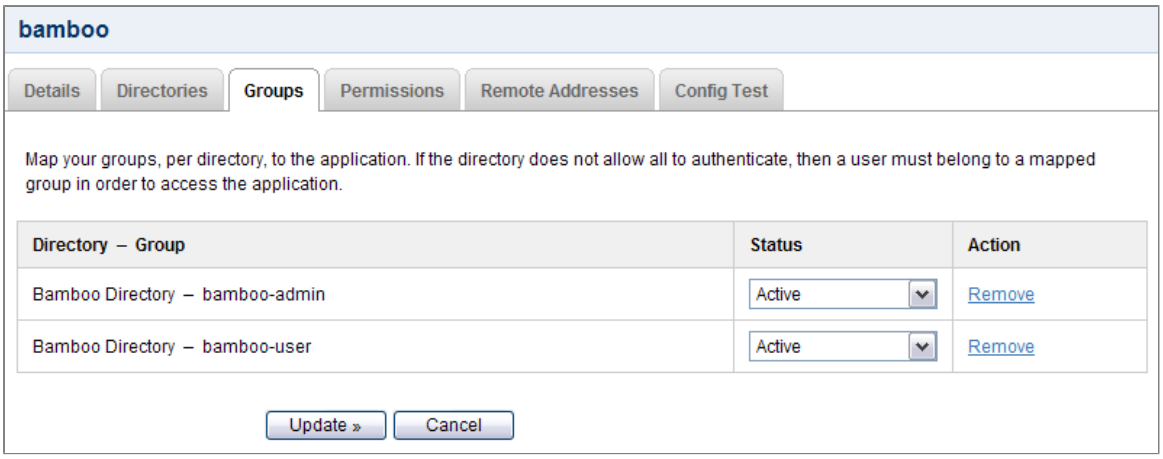
1. Log in to the Crowd Administration Console and navigate to **Applications** > **Add Application**.

2. Complete the **'Add Application' wizard** for the Bamboo application. See the instructions. The **Name** and **Password** values you specify in the 'Add Application' wizard must match the **application.name** and **application.password** that you will set in the `Bamboo/webapp/WEB-INF/classes/crowd.properties` file. (See Step 2 below.)

1.3 Specify which Users can Log In to Bamboo

Once Crowd is aware of the Bamboo application, Crowd needs to know which users can authenticate (log in) to Bamboo via Crowd. As part of the 'Add Application' wizard, you will set up your directories and group authorisations for the application. If necessary, you can adjust these settings after completing the wizard. Below are some examples.

You can either allow entire directories to authenticate, or just particular groups within the directories. In our example, we will allow the `bamboo-user` and `bamboo-admin` groups within the *Crowd Bamboo Directory* to authenticate:



If you are not using a `bamboo-user` group as a security restriction, you will need to set '**Allow all to authenticate**' to 'true' when mapping the directory, otherwise only `bamboo-admin` group members will be able to log in to Bamboo.

1.4 Specify the Address from which Bamboo can Log In to Crowd

As part of the 'Add Application' wizard, you will set up Bamboo's IP address. This is the address which Bamboo will use to authenticate to Crowd. If necessary you can add a hostname, in addition to the IP address, after completing the wizard. See Specifying an Application's Address or Hostname.

Step 2. Configuring Bamboo to Talk to Crowd

If your Bamboo version is **earlier than 1.2.2**, please upgrade to the latest stable version of Bamboo.

2.1 Install the Crowd Client Libraries into Bamboo

Bamboo needs Crowd's client libraries in order to be able to delegate user authentication to the Crowd application. In some cases, you will need to modify the Bamboo application, which is stored in `BAMBOO/webapp`.

1. Please check your versions of Crowd and Bamboo:
    - If you are using **Bamboo 1.2.2** to **1.2.4**, you will need to update the Bamboo libraries as described in this step below.
    - If you are using **Bamboo 2.0** or later, the Crowd client libraries and `crowd.properties` file are included in the Bamboo 2.0 installation download. Please check if your version of Crowd is the same version as the Crowd client library included in the Bamboo 2.x.x installation download (e.g. Bamboo 2.0 currently includes the client library for Crowd 1.3).
        - If the Crowd library versions are different, you will need to update the Bamboo libraries as described in this step below.
        - If the Crowd library versions are the same, you can skip this step.
    - Remove any existing versions of `crowd-integration-client-X.X.X.jar` from your `BAMBOO/webapp/WEB-INF/lib` directory. For example, remove `crowd-integration-client-1.3.jar` and replace it with the client jar provided in your crowd installation.
    - If you are using the Crowd WAR distribution, then you will need to get the CROWD client libraries from the standalone distribution, available on our download site.
    - Copy the Crowd client libraries and configuration files to Bamboo:

Copy From	Copy To
CROWD/client/crowd-integration-client-X.X.X.jar	BAMBOO/webapp/WEB-INF/lib
CROWD/client/conf/crowd.properties	BAMBOO/webapp/WEB-INF/classes
CROWD/client/conf/crowd-ehcache.xml	BAMBOO/webapp/WEB-INF/classes

2. For **Bamboo 1.2.4** only: You will need to remove the `seraph-0.7.23.jar` file from Bamboo's `WEB-INF/lib/` directory and replace it with the following file:
http://repository.atlassian.com/maven2/com/atlassian/seraph/atlassian-seraph/0.10/atlassian-seraph-0.10.jar
(Note: the 0.10 version of the Seraph JAR is newer than 0.7.23.)

2.2 Edit Bamboo's crowd.properties file

Configure the Bamboo application's properties to determine how Crowd will interact with Bamboo.

1. Edit `BAMBOO/webapp/WEB-INF/classes/crowd.properties`. Change the following properties:

Key	Value

application.name	`bamboo` The **application.name** and **application.password** must match the **Name** and **Password** that you specified when defining the application in Crowd (see Step 1 above).
application.password	The **application.name** and **application.password** must match the **Name** and **Password** that you specified when defining the application in Crowd (see Step 1 above).
crowd.server.url	`http://localhost:8095/crowd/services/` If your Crowd server's port is configured differently from the default (8095), set it accordingly.
session.validationinterval	Set to 0, if you want authentication checks to occur on each request. Otherwise set to the number of minutes between requests to validate if the user is logged in or out of the Crowd SSO server. Setting this value to 1 or higher will increase the performance of Crowd's integration.

You can read more about optional settings in the crowd.properties file.

2.3 Configure Bamboo to use Crowd's Authenticator

Now that the Crowd client libraries exist, we need to configure Bamboo to use them.

1. Edit the `Bamboo/webapp/WEB-INF/classes/atlassian-user.xml` file so that the contents of the file is:

```
1.<atlassian-user>
2.          <repositories>
3.
4.                    <crowd key="crowd" name="Crowd Repository"/>
5.
6.          </repositories>
7.</atlassian-user>
```

2. At this stage, Bamboo is set up for **centralised authentication**. If you wish to enable **single sign-on (SSO)** to Bamboo, refer to section 2.5 of this document.

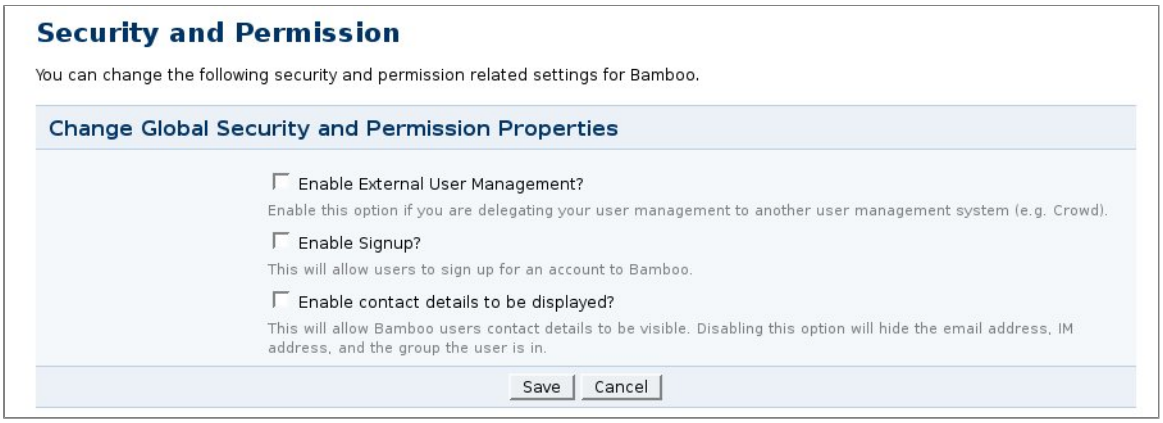
2.4 Configure External User Management in Bamboo

For Bamboo to integrate successfully with Crowd, Bamboo's '**External User Management**' option needs to be:

- **Checked** if you are using an LDAP directory with Crowd and you don't have write-access in LDAP.
- **Unchecked** if you are using internal Crowd directories, or Crowd with LDAP where you do have write-access.
- **Unchecked** if you are using a Delegated Authentication directory.

More information:

- Please ignore the wording on some versions of the Bamboo screens, which may imply that you should check this option.
- In later versions of Bamboo, the option will be called '**Read-Only External User Management**'.
- Refer to the Bamboo documentation for full details of Bamboo's external management configuration.



2.5 (Optional) Enable Single Sign-On

> **SSO is optional**
> Single sign-on (SSO) is optional when integrating Bamboo and other Atlassian products with Crowd. To use centralised authentication *without* SSO, skip the steps below.

To enable single sign-on (SSO), you will configure Bamboo's authentication and access request calls to use Seraph. To configure Seraph-based authentication:

1. Edit the `\BAMBOO\webapp\WEB-INF\classes\seraph-config.xml`
2. Comment out the `authenticator` node :

```
1.<!--<authenticator
class="com.atlassian.bamboo.user.authentication.BambooAuthenticator"/>-->
```

and add a new one:

```
1.<authenticator class="com.atlassian.crowd.integration.seraph.BambooAuthenticator"/>
```

Bamboo's authentication and access request calls will now be performed using Seraph.

2.6 (Optional) Tune the Cache

When using the atlassian-user and Crowd framework together with Bamboo, it is highly recommended that caching be enabled. Multiple redundant calls to the atlassian-user framework are made on any given request. These results can be stored locally between calls by enabling caching via the Crowd Options menu. (Note that this caching in the Crowd application is enabled by default.)

Bamboo will obtain all necessary information for the period specified by the cache configuration - see Configuring Caching for an Application. If a change or addition occurs in Crowd to users, groups and roles, these changes will not be visible in Bamboo until the cache expires for that specific item (i.e. for the particular user, group or role).

The default value for the application cache is 5 minutes (300 seconds). To increase the performance of your application, consider changing the cache value to one or two hours (3600 or 7200 seconds).

See Crowd in Action

Welcome to Bamboo with Crowd!

- Users belonging to the `bamboo-user` group should now be able to log in to Bamboo. Try adding a user to the group using Crowd — you should be able to log in to Bamboo using this newly created user. That's **centralised authentication** in action!
- If you have enabled SSO, you can try adding the *Crowd Bamboo Directory* and `bamboo-admin` group to the *crowd* application (see Mapping a Directory to an Application and Specifying which Groups can access an Application). This will allow Bamboo administrators to log in to the Crowd Administration Console. Try logging in to Crowd as a Bamboo administrator, and then point your browser at Bamboo. You should be logged in as the same user in Bamboo. That's **single sign-on** in action!

RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with Atlassian Confluence

Atlassian's popular Confluence wiki can quickly be configured to use the `atlassian-user` libraries to link in single or multiple directory servers through Crowd.

If you are using NTLM for Windows authentication, you may want to read about configuring Crowd's Confluence NTLM plugin for single sign-on.

Compatibility of Confluence and Crowd Versions

For best performance and support, please ensure that your Crowd and Confluence versions are compatible:

- Crowd versions 1.2 and later support **Confluence 2.6.2 and later**.
- This version of Crowd does not support Confluence 2.6.1 or earlier.
- If you are using **Confluence 2.8 or later**, please upgrade to Crowd 1.3.2 or later.
  Explanation: With Confluence 2.8 the atlassian-user interface has changed, and Crowd 1.3.2 provides the required update to Crowd's atlassian-user integration module.

Prerequisites

1. Download and install Crowd. Refer to the Crowd installation guide for instructions. We will refer to the Crowd root folder as `CROWD`.
2. Download and install Confluence (version 2.6.2 or later). Refer to the Confluence installation guide for instructions. We will refer to the Confluence root folder as `CONFLUENCE`. For the purposes of this document, we will assume that you have used the Standalone (i.e. the easier) installation method of Confluence. If you need to install Confluence as an EAR/WAR, simply explode the EAR/WAR and make the necessary changes as described below, then repackage the EAR/WAR.
3. Run the Confluence Setup Wizard, as described in the Confluence documentation. During this setup process, you will define the Confluence administrator's username and password. It is easier to do this before you integrate Confluence with Crowd.
4. After setting up Confluence, shut down Confluence before you begin the integration process described below.

Step 1. Configuring Crowd to Talk to Confluence

1.1 Prepare Crowd's Directories/Groups/Users for Confluence

The Confluence application will need to authenticate users against a directory configured in Crowd. You will need to set up a directory in Crowd for Confluence. For more information on how to do this, see Adding a Directory. We will assume that the directory is called *Confluence Directory* for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use *Confluence Directory* to house Confluence users.

Confluence also requires particular groups to exist in the directory in order to authenticate users. You will need to create two groups in the *Confluence Directory*:

1. `confluence-users`
2. `confluence-administrators`

See the documentation on Creating Groups for more information on how to define these groups.

You also need to ensure that the *Confluence Directory* contains at least one user who is a member of both groups. Choose one of the two options below:

- If you have an existing Confluence deployment and would like to import existing users and groups into Crowd, use the Confluence Importer tool by navigating to **Users** > **Import Users** > **Atlassian Importer**. Select '**Confluence**' as the Atlassian product, and the *Confluence Directory* as the directory into which Confluence users will be imported. For details please see Importing Users from Atlassian Confluence. If you are going to import users into Crowd, you need to do this now before you proceed any further.
  OR:
- If you don't wish to import your Confluence users, make sure you use Crowd to create at least one user in the *Confluence Directory* and assign them to both the `confluence-users` and the `confluence-administrators` group. The Crowd documentation has more information on creating groups, creating users and assigning users to groups.

1.2 Define the Confluence Application in Crowd

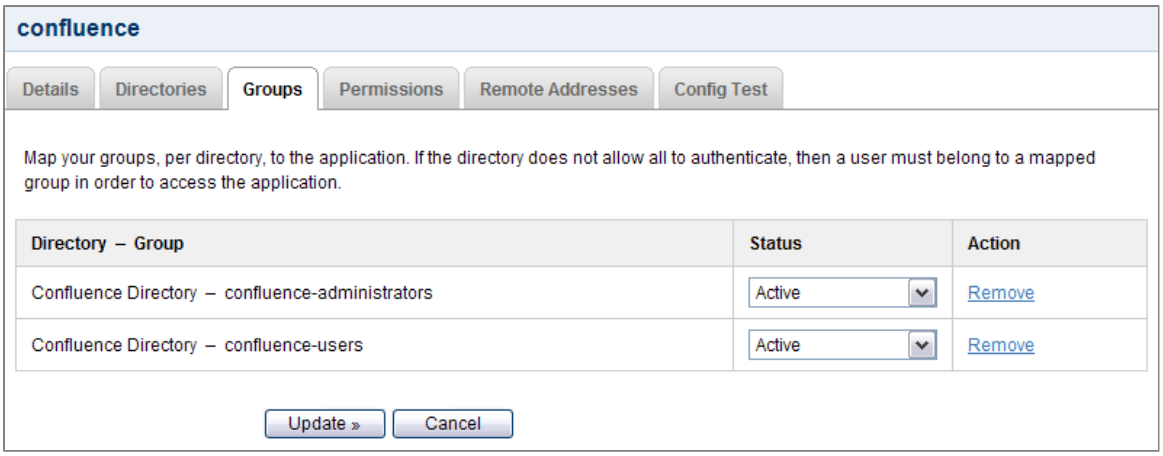
Crowd needs to be aware that the Confluence application will be making authentication requests to Crowd. We need to add the Confluence application to Crowd and map it to the *Confluence Directory*:

1. Log in to the Crowd Administration Console and navigate to **Applications** > **Add Application**.
2. Complete the **'Add Application' wizard** for the Confluence application. See the instructions. The **Name** and **Password** values you specify in the 'Add Application' wizard must match the **application.name** and **application.password** that you will set in the `CONFLUENCE/confluence/WEB-INF/classes/crowd.properties` file. (See Step 2 below.)

1.3 Specify which Users can Log In to Confluence

Once Crowd is aware of the Confluence application, Crowd needs to know which users can authenticate (log in) to Confluence via Crowd. As part of the 'Add Application' wizard, you will set up your directories and group authorisations for the application. If necessary, you can adjust these settings after completing the wizard. Below are some examples.

You can either allow entire directories to authenticate, or just particular groups within the directories. In our example, we will allow the `confluence-users` and `confluence-administrators` groups within the *Confluence Directory* to authenticate:



For details please see Specifying which Groups can access an Application.

1.4 Specify the Address from which Confluence can Log In to Crowd

As part of the 'Add Application' wizard, you will set up Confluence's IP address. This is the address which Confluence will use to authenticate to Crowd. If necessary you can add a hostname, in addition to the IP address, after completing the wizard. See Specifying an Application's Address or Hostname.

Step 2. Configuring Confluence to Talk to Crowd

2.1 Install the Crowd Client Library into Confluence

Confluence needs Crowd's client library and configuration file in order to be able to delegate user authentication to the Crowd application. As stated earlier, we will modify the Confluence application by editing the standalone application, which is an exploded WAR stored in `CONFLUENCE/confluence`.

1. If you are using the Crowd WAR distribution, then you will need to get the CROWD client libraries from the standalone distribution, available on our download site.
2. Copy the Crowd client library and configuration file to Confluence:

Copy From	Copy To
CROWD/client/crowd-integration-client-X.X.X.jar	CONFLUENCE/confluence/WEB-INF/lib
CROWD/client/conf/crowd.properties	CONFLUENCE/confluence/WEB-INF/classes

There is no need to copy across anything from `CROWD/client/lib`. All the required libraries from that directory already exist in Confluence versions 2.3 and later.

> **A note about older Confluence versions**
> Confluence **2.5.6 to 2.6.1** are not compatible with Crowd 1.2 and later. We recommend that you upgrade to Confluence **2.6.2 or later**.
> If you can not upgrade your Confluence instance, you will need to remove the `seraph-X.X.X.jar` file from Confluence's `<CONFLUENCE-INSTALLATION>/confluence/WEB-INF/lib/seraph-X.X.X.jar` and replace it with the following file:
> http://repository.atlassian.com/maven2/com/atlassian/seraph/atlassian-seraph/0.10/atlassian-seraph-0.10.jar

3. Replace Confluence's cache configuration file:

Copy From	Replace File
CROWD/client/conf/crowd-ehcache.xml	CONFLUENCE/confluence/WEB-INF/classes/crowd-ehcache.xml

4. Edit `CONFLUENCE/confluence/WEB-INF/classes/crowd.properties`. Change the following properties:

Key	Value

application.name	`confluence` The **application.name** and **application.password** must match the **Name** and **Password** that you specified when defining the application in Crowd (see Step 1 above).
application.password	The **application.name** and **application.password** must match the **Name** and **Password** that you specified when defining the application in Crowd (see Step 1 above).
crowd.server.url	`http://localhost:8095/crowd/services/` If your Crowd server's port is configured differently from the default (i.e. 8095), set it accordingly.
session.validationinterval	This is the number of minutes between validation requests, when Crowd validates whether the user is logged in to or out of the Crowd SSO server. Set this value to 0 if you want authentication checks to occur on each request. Otherwise set to the required number of minutes between validation requests. Setting this value to 1 or higher will increase the performance of Crowd's integration.

You can read more about optional settings in the crowd.properties file.

2.2 Configure Confluence to use Crowd's Authenticator

Now that the Crowd client libraries exist, we need to configure Confluence to use them.

1. Edit the `CONFLUENCE/confluence/WEB-INF/classes/atlassian-user.xml` file so that the contents of the file is:

```
1.<atlassian-user>
2.          <repositories>
3.
4.                    <crowd key="crowd" name="Crowd Repository"/>
5.
6.          </repositories>
7.</atlassian-user>
```

2. At this stage, Confluence is set up for **centralised authentication**. If you wish, you can now enable **single sign-on (SSO)** to Confluence.
   Skip this step if you are using the Confluence NTLM plugin to enable SSO. Instead, follow the instructions on configuring Confluence for NTLM SSO.

   Edit `CONFLUENCE/confluence/WEB-INF/classes/seraph-config.xml`. Comment out the `authenticator` node :

```
1.<!--<authenticator class="com.atlassian.confluence.user.ConfluenceAuthenticator"/>-->
```

   and add a new one:

```
1.<authenticator class="com.atlassian.crowd.integration.seraph.ConfluenceAuthenticator"/>
```

   Confluence's authentication and access request calls will now be performed using Seraph.

2.3 Enable Confluence's External User Management

Once the setup is complete, you may wish to turn 'External User Management' **on** in Confluence. This will prevent Confluence administrators from being able to add or update users. For more information please see the Confluence documentation regarding External User Management.

- If you are using Confluence 2.6.2 or earlier, this step is required i.e. you must turn on external user management in Confluence.
- If your Crowd directory permissions are configured so that Confluence cannot update the Crowd directories, this step is required i.e. you must turn on external user management in Confluence. Otherwise, an error will occur when Confluence attempts to write data into Crowd.
- If you have imported Confluence users into Crowd, you may want to delay turning on 'External User Management' for a week or two, to give users time to reset their passwords. (Because users' passwords are encrypted in Confluence's database, they will not be copied across to Crowd.)

2.4 (Optional) Tune the Cache

Enabling caching on the Crowd server: When using the Atlassian-User and Crowd framework together with Confluence, it is highly recommended that caching be enabled on the Crowd server. Multiple redundant calls to the Atlassian-User framework are made on any given request. These results can be stored locally between calls by enabling caching via the Crowd Options menu. Note that this caching on the Crowd server is enabled by default.

Enabling application caching for Confluence: If application caching is enabled for Confluence, Confluence will obtain all necessary information for the period specified by the cache configuration. See Configuring Caching for an Application. If a change or addition occurs to Crowd users, groups and roles, these changes will not be visible in Confluence until the cache expires for that specific item, i.e. for the

particular user, group or role.

The default period for the application cache is 5 minutes (300 seconds). To increase the performance of your application, consider changing the cache value to one or two hours (3600 or 7200 seconds).

See Crowd in Action

- Users belonging to the `confluence-users` group should now be able to log in to Confluence.
- Try adding a user to the `confluence-users` group using Crowd — you should be able to log in to Confluence using this newly created user. That's **centralised authentication** in action!
- If you have enabled SSO, you can try adding the *Confluence Directory* and `confluence-administrators` group to the *crowd* application (see Mapping a Directory to an Application and Specifying which Groups can access an Application). This will allow Confluence administrators to log in to the Crowd Administration Console. Try logging in to Crowd as a Confluence administrator, and then point your browser at Confluence. You should be logged in as the same user in Confluence. That's **single sign-on** in action!

RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Configuring Confluence for NTLM SSO

> **Confluence NTLM plugin not officially supported by Atlassian**
> The Confluence NTLM plugin was written by a third party. Atlassian does not officially support the plugin. The Atlassian Crowd team will do our best to advise on any Crowd integration problems. Please refer to the plugin documentation for installation instructions and further support.

Out of the box, Confluence does not support Single Sign On (SSO) functionality. This page describes how to set up Confluence with NTLM SSO functionality using the Confluence NTLM plugin, Crowd, and Active Directory (AD) as your LDAP user repository.

Summary

The Confluence NTLM plugin enables the following authentication scenario:

- A user in a Windows domain logs into the Windows network, using their Active Directory username/password.
- Then, when they open Confluence in an Internet Explorer browser, they are seamlessly logged into Confluence.

The Crowd component then allows you to manage all users and groups in Active Directory. Crowd automatically ensures that users and groups are synchronised between AD and Confluence. For example, if a user/group is added/deleted from AD it will be automatically added/deleted from Confluence.

Components

Confluence NTLM plugin	NTLM is the protocol used by Windows for authentication. The Confluence NTLM plugin takes care of the Windows domain / Active Directory login to Confluence. You must be running a Windows Domain Controller with accounts set up in AD in order to use this plugin. If NTLM authentication is not available, the plugin allows standard form-based login to Confluence. **Note:** This plugin is not officially supported by Atlassian.
Crowd	Crowd takes care of the synchronisation of users/groups between Active Directory and Confluence. You will need to create an SSL connection between Crowd and the AD server if you would like to create users through Crowd. AD will not allow Crowd to add users or change their passwords unless the communication occurs over a secure connection.
Active Directory (AD) on Windows 2003 Server	Active Directory (AD) on Windows 2003 Server — you must already have an AD instance set up and running with a domain controller.
Confluence	The machine running Confluence must be part of the Windows domain or installed on the same box as the domain controller.

Steps

1. Back up your Confluence installation files and data:
   - Confluence Home directory. (See Confluence's Important Directories and Files for how to locate this).
   - Confluence installation directory (if you are using Confluence Standalone) or your Confluence webapp (if you are using Confluence EAR-WAR).
   - Your database (if you are not using the embedded database).
2. Download the Confluence NTLM plugin.
3. Install the plugin, following the instructions on the plugin documentation page.
4. In the `ldaputil.properties` file, insert the appropriate LDAP and Domain Controller information along with other parameters.
5. Install and configure Crowd.
6. Create a directory in Crowd for the AD LDAP server.
7. Create the Confluence application in Crowd and configure Crowd and Confluence to talk to each other, as described in Integrating Crowd with Atlassian Confluence.

   When following the above instructions, **do not** change the `seraph-config.xml` file to enable Crowd's SSO functionality. (I.e. don't change the `authenticator` node to read `<authenticator class="com.atlassian.crowd.integration.seraph.ConfluenceAuthenticator"/>`. Instead of Crowd's SSO authentication, we'll be using the Confluence NTLM plugin.

8. In AD, create the groups **confluence-users** and **confluence-administrators**. They should then appear in Crowd.
9. In AD, create an admin user and make them a member of the above groups in AD.
10. Create any additional groups that you would like in AD.
11. Log in to the Windows domain using your desktop login and then open Confluence in an Internet Explorer browser. You should be logged in automatically.

Additional Crowd Performance Tips

- Change the default cache setting timeout in the file `<CONFLUENCE>\WEB-INF\classes\crowd-ehcache.xml`. For performance reasons, increase the object caching to 7,200 seconds (2 hours):
  `timeToIdleSeconds="7200" timeToLiveSeconds="7200".`
  This reduces the frequency of the requests from Crowd to the LDAP server when changes to LDAP objects (such as a group name or user attribute) are made, thus reducing the performance overhead.
- Turn on the '**Use Paged Results**' option in the directory connector tab for the directory you've set up in Crowd.

Integrating Crowd with Atlassian CrowdID

Atlassian CrowdID is a free add-on to Crowd. It gives administrators a secure way to provide OpenID accounts for their users.

When installing Crowd 1.1+ the Crowd Setup Wizard allows you to install CrowdID with Crowd. If you chose to install CrowdID as part of the Setup Wizard, there is no need for further configuration. The CrowdID server will be up and running at http://localhost:8095/openidserver

If you have not already installed CrowdID, follow the instructions below to install it now.

Prerequisites

1. Download and install Crowd. Refer to the Crowd installation guide for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.

2. This guide assumes that CrowdID was NOT installed with the installation of Crowd. If CrowdID was installed using the Crowd Setup Wizard, there is no need for further configuration.

Step 1. Configuring Crowd to Talk to CrowdID

1.1 Prepare Crowd's Directories/Groups/Users for CrowdID

The CrowdID application will need to locate users from a directory configured in Crowd. You will need to set up a directory in Crowd for CrowdID. For information on how to do this, see Adding a Directory. We will assume that the directory is called *CrowdID Directory* for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use *CrowdID Directory* to house CrowdID users.

CrowdID also requires an administrator group to exist in the directory. You need to ensure that a `crowd-administrators` groups exist in the *CrowdID Directory*. Any user in this group will have CrowdID administrator access.

The Crowd documentation has more information on creating groups, creating users and assigning users to groups.

1.2 Define the CrowdID Application in Crowd

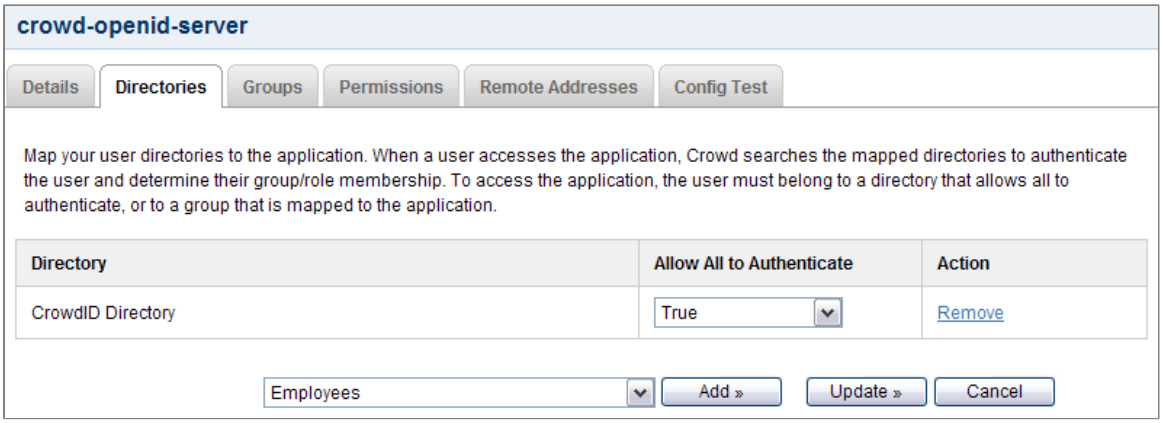
Crowd needs to be aware that the CrowdID application will be making authentication requests to Crowd. We need to add the CrowdID application to Crowd and map it to the *CrowdID Directory*.

1. Log in to the Crowd Administration Console and navigate to **Applications** > **Add Application**.
2. Complete the **'Add Application' wizard** for the CrowdID application. See the instructions. The **Name** and **Password** values you specify in the 'Add Application' wizard must match the **application.name** and **application.password** that you will set in the `CROWD/crowd-openidserver-webapp/WEB-INF/classes/crowd.properties` file. (See Step 2 below.)

1.3 Specify which Users can Log In to CrowdID

Once Crowd is aware of the CrowdID application, Crowd needs to know which users can authenticate (log in) to CrowdID via Crowd. As part of the 'Add Application' wizard, you will set up your directories and group authorisations for the application. If necessary, you can adjust these settings after completing the wizard. Below are some examples.

You can either allow entire directories to authenticate, or just particular groups within the directories. In our example, we will allow the entire *CrowdID Directory* to authenticate:



For details please see Specifying which Groups can access an Application.

1.4 Specify the Address from which CrowdID can Log In to Crowd

As part of the 'Add Application' wizard, you will set up CrowdID's IP address. This is the address which CrowdID will use to authenticate to Crowd. If necessary you can add a hostname, in addition to the IP address, after completing the wizard. See Specifying an Application's Address or Hostname.

Step 2. Configuring CrowdID to Talk to Crowd

Edit `CROWD/crowd-openidserver-webapp/WEB-INF/classes/crowd.properties`. Change the following properties:

Key	Value
application.name	`crowd-openid-server` The **application.name** and **application.password** must match the **Name** and **Password** that you specified when you defined the application in Crowd (see Step 1 above).
application.password	The **application.name** and **application.password** must match the **Name** and **Password** that you specified when you defined the application in Crowd (see Step 1 above).

application.login.url	`http://localhost:8095/openidserver` The **application.login.url** should point to the correct host and port of the CrowdID application.
crowd.server.url	`http://localhost:8095/crowd/services/` If your Crowd server's port is configured differently from the default (i.e. 8095), set it accordingly.
session.validationinterval	This is the number of minutes between validation requests, when Crowd validates whether the user is logged in to or out of the Crowd SSO server. Set this value to 0 if you want authentication checks to occur on each request. Otherwise set to the required number of minutes between validation requests. Setting this value to 1 or higher will increase the performance of Crowd's integration.

You can read more about optional settings in the crowd.properties file.

See CrowdID in Action

- Go to http://localhost:8095/openidserver and log in with any user in the *CrowdID Directory*.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with Atlassian Crucible

You can use Crowd to provide external authentication and authorisation for Atlassian's Crucible code review tool.

Crucible and FishEye
When you purchase and install Crucible, you may also purchase Atlassian's FishEye source-repository viewer. If you have both FishEye and Crucible, they will share a common authentication mechanism and integration with Crowd. Crucible and FishEye will authenticate to Crowd using the same application name and password. See Integrating Crowd with Atlassian FishEye. If you have Crucible only (available from **Crucible 1.6**), you will need to set up the Crowd directory and application in the same way, following the instructions in Integrating Crowd with Atlassian FishEye.

Prerequisites

1. Download and install Crowd. Refer to the Crowd installation guide for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.
2. Download and install Crucible. Refer to the Crucible Installation Guide for detailed information on how to do this.
3. Follow the instructions on integrating Crowd with FishEye.
   For **Crucible versions 1.2.x and later**, refer to the instructions for FishEye 1.4. For **Crucible 1.1.x and earlier**, refer to the the instructions for FishEye 1.3.

Configure Authorisation in Crucible Projects (If Required)

Optionally, you can now use the Crowd users and/or groups in the permission schemes for your Crucible projects. If you have created groups in the Crowd directory which is mapped to your FishEye application (see Integrating Crowd with Atlassian FishEye), the Crowd groups can be seen in Crucible.

Please refer to the Crucible documentation for instructions on:

- Creating projects in Crucible (here).
- Creating permission schemes and assigning them to users and/or groups (here).
- Linking the permission scheme to a Crucible project (here).

RELATED TOPICS

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with Atlassian FishEye

You can use Crowd to provide external authentication and authorisation for Atlassian's FishEye source-repository viewer.

Crowd supports centralised authentication and single sign-on (SSO) for **FishEye versions 1.3.1 and later**.

> **Crucible and FishEye**
> If you are using Atlassian's Crucible code review tool, you will need to follow the instructions below on integrating Crowd with FishEye. If you have the standalone version of Crucible without FishEye (available from **Crucible 1.6**), please follow the instructions below to set up the Crowd directory and application for Crucible instead of FishEye. If preferred, you can change the name of your Crowd application and directory to 'Crucible' rather than 'FishEye'. Then follow the further instructions to integrate Crowd with Crucible.

On this page:

- Prerequisites
- Step 1. Configuring Crowd to Talk to FishEye
  - 1.1 Prepare Crowd's Directories/Groups/Users for FishEye
  - 1.2 Define the FishEye Application in Crowd
  - 1.3 Specify which Users can Log In to FishEye
  - 1.4 Specify the Address from which FishEye can Log In to Crowd
- Step 2. Configuring FishEye to Talk to Crowd
  - 2.1 Change the Details of your Existing FishEye Users
  - 2.2 Configure FishEye to use Crowd's Authenticator
  - 2.3 Configure Group Authorisation in FishEye (If Required)
- Next Step for Crucible Users

Prerequisites

1. Download and install Crowd. Refer to the Crowd installation guide for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.
2. Download and install FishEye. Refer to the FishEye Installation Guide for detailed information on how to do this. We will refer to the FishEye root folder as `FISHEYE`.
   If you have the standalone version of Crucible (available from **Crucible 1.6**), there is no need to download or install FishEye.
3. After FishEye is set up, make sure FishEye is not running when you begin the integration process described below.

Step 1. Configuring Crowd to Talk to FishEye

1.1 Prepare Crowd's Directories/Groups/Users for FishEye

The FishEye application will need to authenticate users against a directory configured in Crowd. You will need to set up a directory in Crowd for FishEye. For more information on how to do this, see Adding a Directory. We will assume that the directory is called *FishEye Directory* for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use

FishEye Directory to house FishEye users.

If you wish to use Crowd groups to control access to your FishEye repositories, you should set up your groups in Crowd. See the documentation on Creating Groups for more information on how to define these groups.

Use Crowd to create at least one user in the *FishEye Directory*. If you are using groups, assign your user(s) to the appropriate groups. The Crowd documentation has more information on creating users and assigning users to groups.

1.2 Define the FishEye Application in Crowd

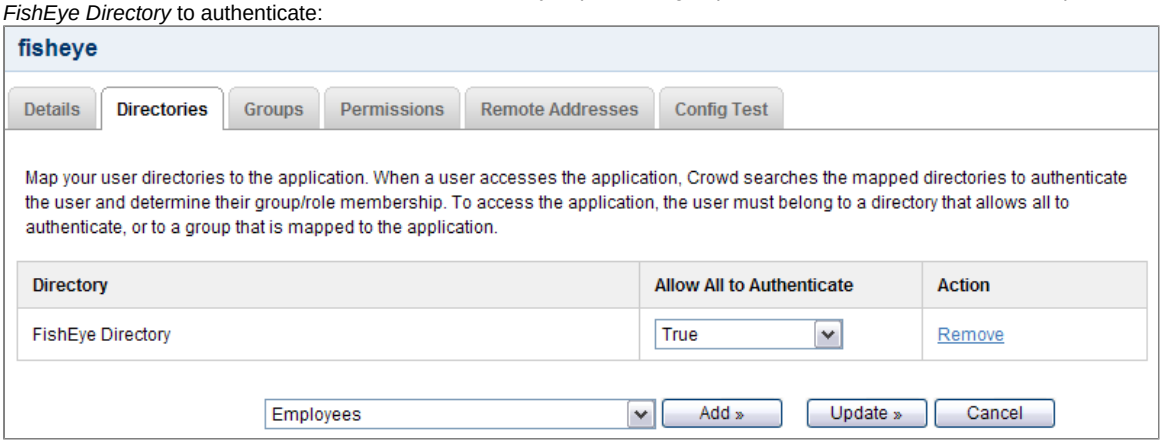
Crowd needs to be aware that the FishEye application will be making authentication requests to Crowd. We need to add the FishEye application to Crowd and map it to the *FishEye Directory:*

1. Log in to the Crowd Administration Console and navigate to **Applications** > **Add Application**.
2. Complete the **'Add Application' wizard** for the FishEye application. See the instructions. The **Name** and **Password** values you specify in the 'Add Application' wizard must match the '**Application name**' and '**Application password**' that you will set in FishEye's '**Crowd Authentication Settings**' screen. (See Step 2 below.)

1.3 Specify which Users can Log In to FishEye

Once Crowd is aware of the FishEye application, Crowd needs to know which users can authenticate (log in) to FishEye via Crowd. As part of the 'Add Application' wizard, you will set up your directories and group authorisations for the application. If necessary, you can adjust these settings after completing the wizard. Below are some examples.

You can either allow entire directories to authenticate, or just particular groups within the directories. In our example, we will allow the entire *FishEye Directory* to authenticate:



If you wish to authorise specific groups only, please see Mapping a Directory to an Application and Specifying which Groups can access an Application.

1.4 Specify the Address from which FishEye can Log In to Crowd

As part of the 'Add Application' wizard, you will set up FishEye's IP address. This is the address which FishEye will use to authenticate to Crowd. If necessary you can add a hostname, in addition to the IP address, after completing the wizard. See Specifying an Application's Address or Hostname.

Step 2. Configuring FishEye to Talk to Crowd

The instructions below are for **FishEye 1.4.x and later**. If you are using FishEye 1.3.x, please follow the guide for earlier versions of FishEye.

2.1 Change the Details of your Existing FishEye Users

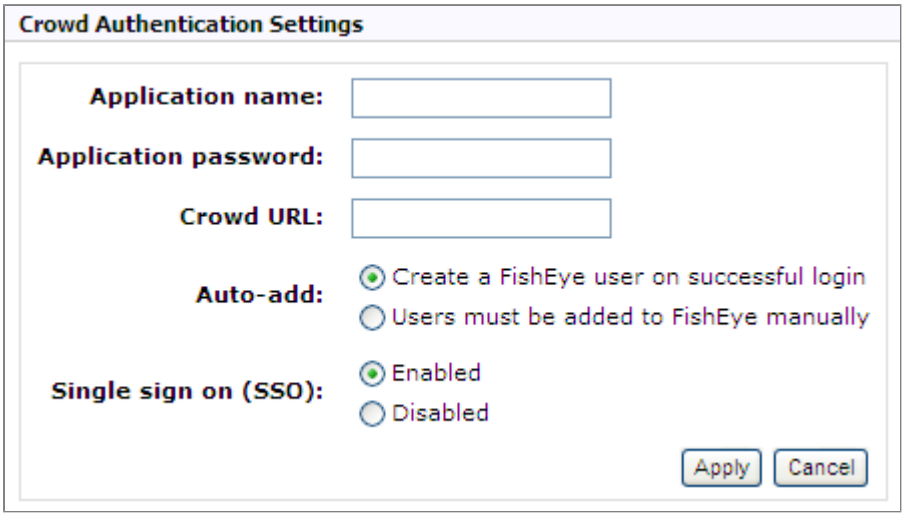
If you have an existing FishEye installation with existing built-in users, please do the following for each username in FishEye:

- Change the account type from '**built-in**' to '**crowd**'. This is required for the new authorisation through Crowd to work properly. For details please see the Fisheye documentation.
- Ensure that the username in FishEye is the same as in Crowd. If necessary, rename the user in FishEye. See the FishEye documentation for details.

2.2 Configure FishEye to use Crowd's Authenticator

1. Log in to the FishEye Administration screens and navigate to '**Security**'.
2. Select '**Setup Crowd authentication**'.
   FishEye allows only one authentication method to be configured at any one time. If you have already configured a different authentication source, click the '**Remove**' link to remove that authentication method. You will then be presented with the options for different authentication methods – one will be the option to set up Crowd authentication.
3. The '**Crowd Authentication Settings**' screen will appear, as shown below. Enter the following information:

- **Application name** – The name for the FishEye application you specified in Step 1 above.
- **Application password** – The password you specified in Step 1 above.
- **Crowd URL** – `http://localhost:8095/crowd/services/`
  \(i) The trailing slash is required.
- **Auto-add** – Select '**Create a FishEye user on successful login**' (default) to ensure that your Crowd users will be automatically enrolled into FishEye when they first log in via Crowd.
- **Single sign on (SSO)** — Controls whether FishEye should attempt to participate in a single sign on (SSO) environment.
  This SSO option is available only with **FishEye 1.5.1** and later.
  - Select '**Enabled**' (default) if you want FishEye to use Crowd's SSO capability.
  - Select '**Disabled**' if you want FishEye to use Crowd to check username/passwords and group membership, without participating in SSO. In this mode, FishEye will not read or set `crowd.token` cookies. This is useful in environments where you want FishEye to ignore `crowd.token` cookies set by other Crowd-enabled applications.



For more information, please see the Fisheye documentation on configuring external authentication sources.

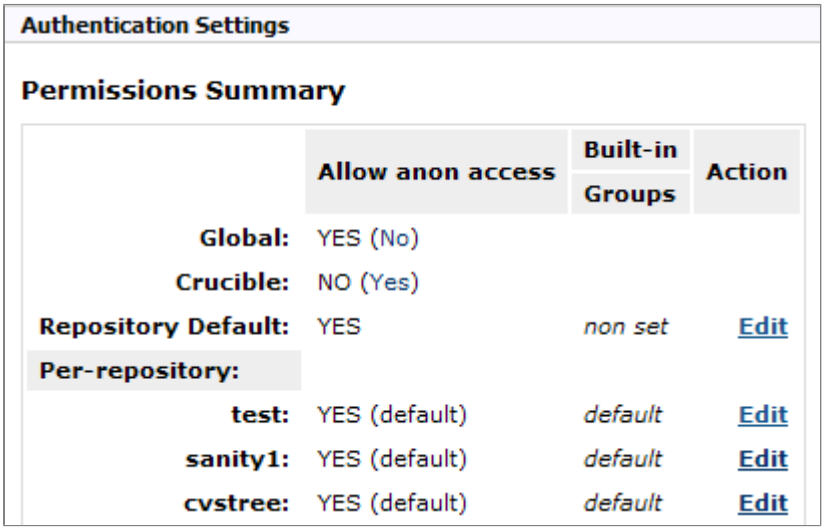
2.3 Configure Group Authorisation in FishEye (If Required)

If you have created groups in the Crowd directory which is mapped to your FishEye application (see Step 1 above), the Crowd groups can be seen in FishEye. Now you can set up group authorisation for your FishEye repositories.

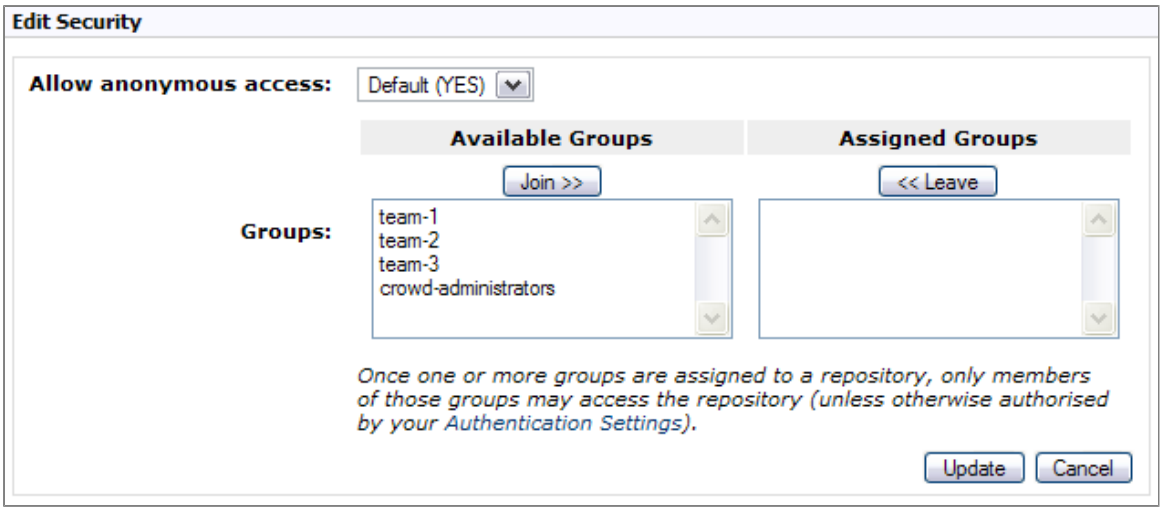
Allow the groups to access your FishEye repositories as follows:

1. In the FishEye Administration menu, select '**Security**' under '**Global Settings**'.
2. This will display the '**Authentication Settings**' screen. In the '**Permissions Summary**' section, click '**Edit**' next to the required repository name under '**Per-repository**'.
3. The '**Edit Security**' screen will appear. Select the group name(s) and click the '**Join**' button. Click '**Update**'. The group(s) will appear in the '**Built-in Groups**' section of the 'Authentication Settings' screen.

Screenshot 1: 'Authentication Settings'



Screenshot 2: 'Edit Security'



Next Step for Crucible Users

If you are using Atlassian's Crucible code review tool, please take a look at the further instructions on integrating Crowd with Crucible.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Configuring FishEye 1.3.x to talk to Crowd

This page forms part of the guide on Integrating Crowd with Atlassian FishEye and Crucible.

Use the instructions below if you are integrating Crowd with **FishEye version 1.3.x**. If you are using FishEye 1.4.x or later, refer to the instructions for later versions of FishEye.

Step 1. Configuring Crowd to talk to FishEye

Please complete **Step 1** in the full Crowd/FishEye integration instructions.

Step 2. Configuring FishEye to talk to Crowd

Before you begin
For any usernames that are already configured through the Fisheye Administration console, you will need to change the account type from 'built-in' to 'custom'. This is required for the new authorisation through Crowd to work properly.

For details please see the Fisheye documentation.

2.1 Install the Crowd Client Libraries into FishEye

Copy the Crowd integration libraries and configuration files as described in Integrating Crowd with a Custom Application. This involves copying all client library JARs to the library folder of FishEye:

The version numbers have been omitted. Select the JAR which matches the name. This listing has been verified with FishEye 1.3.1.

Files to Copy	Destination
CROWD/client/crowd-integration-client-X.X.X.jar	$FISHEYE_INST/lib

CROWD/client/lib/commons-codec-1.3.jar	$FISHEYE_INST/lib
CROWD/client/lib/commons-httpclient-3.0.jar	$FISHEYE_INST/lib
CROWD/client/lib/commons-lang-2.3.jar	$FISHEYE_INST/lib
CROWD/client/lib/jdom-1.0.jar	$FISHEYE_INST/lib
CROWD/client/lib/stax-api-1.0.1.jar	$FISHEYE_INST/lib
CROWD/client/lib/wsdl4j-1.6.1.jar	$FISHEYE_INST/lib
CROWD/client/lib/wstx-asl-3.2.0.jar	$FISHEYE_INST/lib
CROWD/client/lib/xfire-core-1.2.6.jar	$FISHEYE_INST/lib

2.2 Configure FishEye to use Crowd's Authenticator

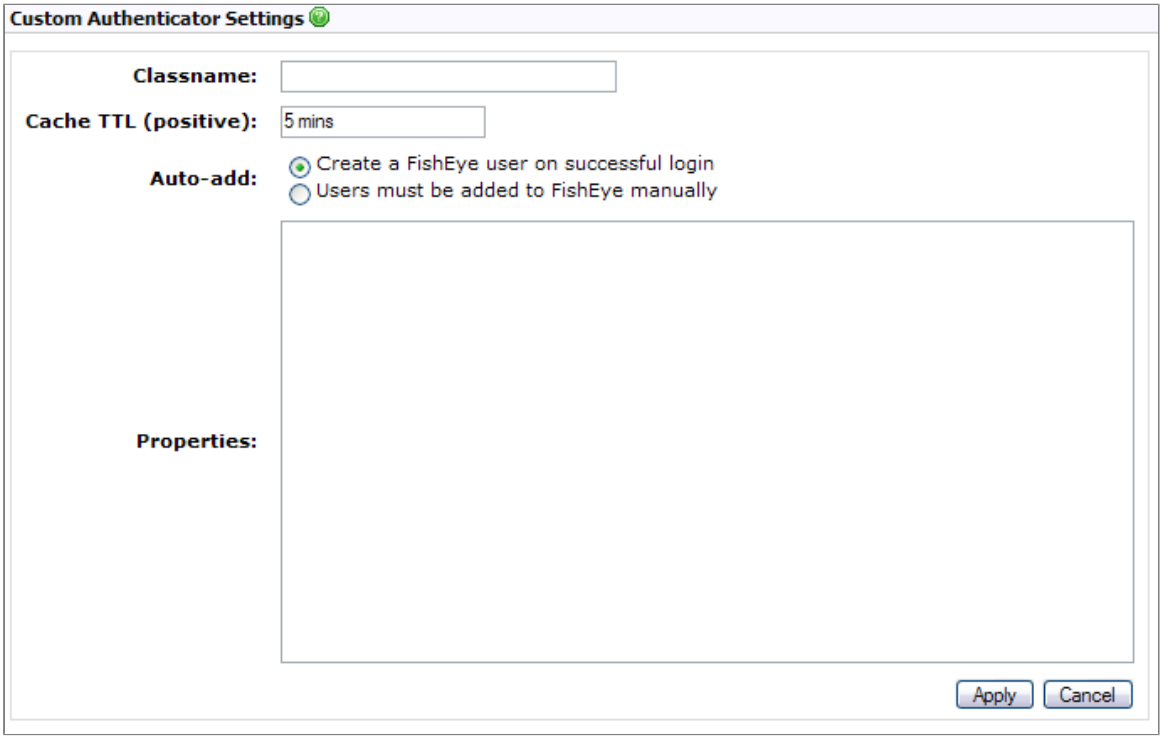
1. Log in as an administrator to FishEye and navigate to '**Users/Security**'. Select '**Setup Custom authentication**'.
   - Enter the following '**Classname**' for the authenticator:

     ```
     1.com.atlassian.crowd.integration.fisheye.FisheyeAuthenticator
     ```

     Leave the cache and auto-add settings at their default values. This will mean authentication calls to Crowd will be cached (improves performance) and that users will be automatically enrolled into FishEye after their initial login to FishEye via Crowd.

   - Fisheye requires you to pass in the configuration attributes for Crowd. Add the following information in the '**Properties**' text box, replacing the information with your own configuration data – match the values set in Step 1.

     ```
     1.application.name            fisheye
     2.application.password        password
     3.application.login.url       http://localhost:8080/
     4.crowd.server.url            http://localhost:8095/crowd/services/
     5.session.isauthenticated     session.isauthenticated
     6.session.tokenkey            session.tokenkey
     7.session.validationinterval  0
     8.session.lastvalidation      session.lastvalidation
     ```



Refer to the FishEye documentation for further details on using the FishEye setup screens.

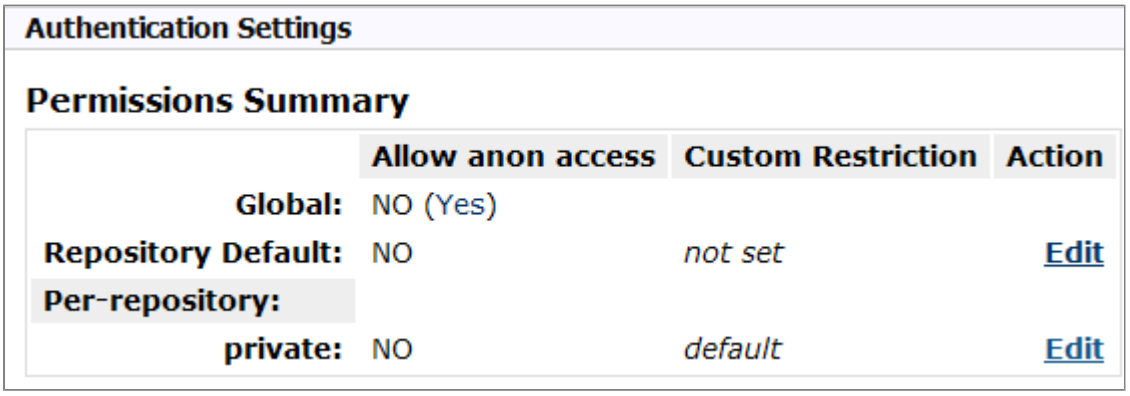
2.3 Configure Groups for FishEye Source Repositories (If Required)

If you are using any FishEye groups to control access to particular source repositories, you will need to create the groups in Crowd and then configure FishEye as follows:

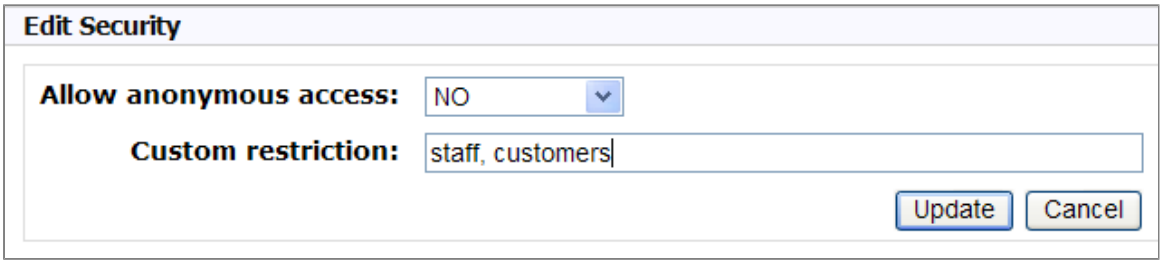
1. In the FishEye Administration menu, select '**Global Settings**', then '**Users/Security**'.
2. This will display the '**Authentication Settings**' screen. In the '**Permissions Summary**' section, edit the '**Per-repository**' field and

enter the group names (separated by commas) in the '**Custom restriction**' field.

Screenshot 1: 'Authentication Settings'



Screenshot 2: 'Custom Restriction'



Related Topics

Unable to render {children}. Page not found: 3. Managing Applications.

Crowd Documentation

Integrating Crowd with Atlassian JIRA

Atlassian's popular JIRA issue management system takes advantage of the OSUser framework and can quickly be configured to use OSUser to link in single or multiple directory servers through Crowd. Crowd provides integration libraries for the OpenSymphony OSUser module, which has a simple-to-use API for user management that allows pluggable implementations. You can read more about the OSUser API at http://www.opensymphony.com/osuser/.

Currently Crowd supports centralised authentication and single sign-on for JIRA versions 3.7.4 and later.

Please check that this documentation applies to your version of Crowd
Please check the Crowd release number in this documentation against your version of Crowd. If you are using a different version of Crowd, you can find the appropriate documentation under 'Previous Versions' on the Crowd documentation homepage.

NTLM SSO
If you are using NTLM for Windows authentication, you may want to read about configuring Crowd's JIRA NTLM plugin for single sign-on.

Prerequisites

1. Download and install Crowd. Refer to the Crowd installation guide for instructions. We will refer to the Crowd root folder as `CROWD`.
2. Download and install JIRA (version 3.7.4 or later). Refer to the JIRA installation guide for instructions. We will refer to the JIRA root folder as `JIRA`. For the purposes of this document, we will assume that you have used the 'Standalone' (i.e. the easier and recommended) installation method of JIRA. If you need to install JIRA as an EAR/WAR, simply explode the EAR/WAR and make the necessary changes as described below, then repackage the EAR/WAR.
3. Run the JIRA Setup Wizard, as described in the JIRA documentation. During this setup process, you will define the JIRA administrator's username and password. It is easier to do this before you integrate JIRA with Crowd.
4. After setting up JIRA, shut down JIRA before you begin the integration process described below.

Step 1. Configuring Crowd to talk to JIRA

1.1 Prepare Crowd's Directories/Groups/Users for JIRA

1. The JIRA application will need to locate users from a directory configured in Crowd. You will need to set up a directory in Crowd for JIRA. This directory may be any Crowd-configured directory, such as an LDAP directory hooked up to Crowd or a Crowd internal directory. For information on how to do this, see Adding a Directory.

   We will assume that the directory is called *JIRA Directory in Crowd* for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use *JIRA Directory in Crowd* to house JIRA users.

2. JIRA also requires particular groups to exist in the directory in order to authenticate users. You need to ensure that these three groups exist in the *JIRA Directory in Crowd*:
   - `jira-users`
   - `jira-developers`
   - `jira-administrators`

3. You also need to ensure that the *JIRA Directory in Crowd* contains at least one user who is a member of all three groups. You can either:
   - If you have an existing JIRA deployment and would like to import existing groups and users into Crowd, use the JIRA Importer tool by navigating to **Users** > **Import Users** > **Atlassian Importer**. Select 'JIRA' as the Atlassian Product and the *JIRA Directory in Crowd* as the directory into which JIRA users will be imported. For details please see Importing Users from Atlassian JIRA. If you are going to import users into Crowd, you need to do this now before you proceed any further.
     OR:
   - If you don't wish to import your JIRA users, use the Crowd Administration Console to create the three groups, then create at least one user in the *JIRA Directory in Crowd* and add them to the three JIRA-specific groups (above). The Crowd documentation has more information on creating groups, creating users and assigning users to groups.

Error will occur in JIRA if the required groups do not exist
JIRA expects that the group names mentioned above will exist. If you need to use different group names, you may want to remove the above pre-existing groups from JIRA's Global Permissions. If the above groups do not exist somewhere in Crowd, you will receive an error when you try to remove the groups from JIRA's Global Permissions.

1.2 Define the JIRA Application in Crowd

Crowd needs to be aware that the JIRA application will be making authentication requests to Crowd. We need to add the JIRA application to Crowd and map it to the *JIRA Directory in Crowd*.

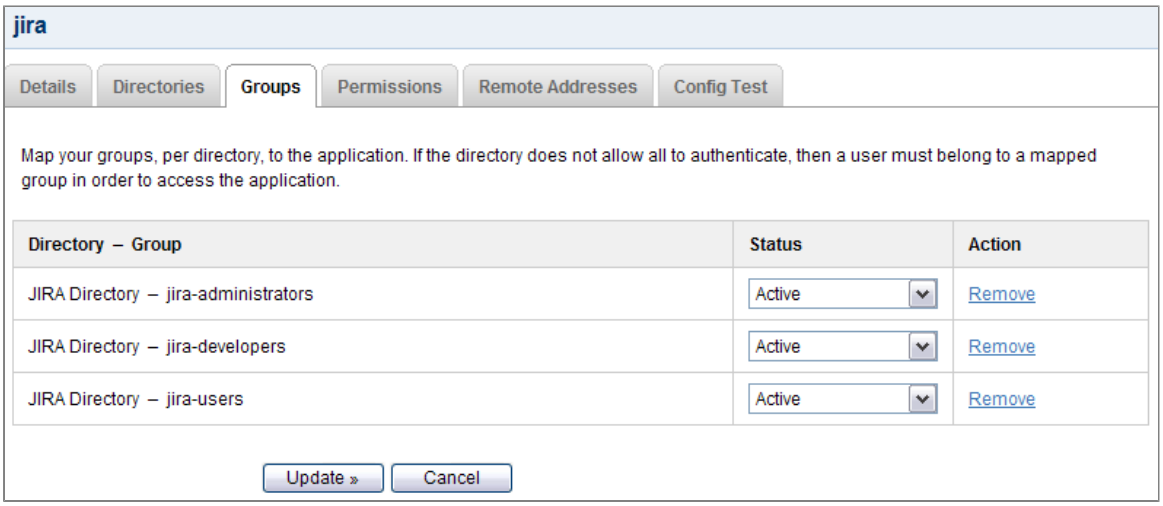
1. Log in to the Crowd Administration Console and navigate to **Applications** > **Add Application**.

2. Complete the **'Add Application' wizard** for the JIRA application. See the instructions. The **Name** and **Password** values you specify in the 'Add Application' wizard must match the **application.name** and **application.password** that you will set in the `JIRA/atlassian-jira/WEB-INF/classes/crowd.properties` file. (See Step 2 below.)

1.3 Specify which users can log in to JIRA

Once Crowd is aware of the JIRA application, Crowd needs to know which users can authenticate (log in) to JIRA via Crowd. As part of the 'Add Application' wizard, you will set up your directories and group authorisations for the application. If necessary, you can adjust these settings after completing the wizard. Below are some examples.

You can either allow entire directories to authenticate, or just particular groups within the directories. In our example, we will allow the `jira-users`, `jira-developers` and `jira-administrators` groups within the *JIRA Directory in Crowd* to authenticate:



With this example, only users who are members of the `jira-users`, `jira-developers` and `jira-administrators` groups will be able to authenticate against JIRA.

For details please see Specifying which Groups can access an Application.

1.4 Specify the address from which JIRA can log in to Crowd

As part of the 'Add Application' wizard, you will set up JIRA's IP address. This is the address which JIRA will use to authenticate to Crowd. If necessary you can add a hostname, in addition to the IP address, after completing the wizard. See Specifying an Application's Address or Hostname.

Step 2. Configuring JIRA to talk to Crowd

2.1 Install the Crowd Client Libraries into JIRA

JIRA needs Crowd's client libraries in order to be able to delegate user authentication to the Crowd application. As stated earlier, we are going to be modifying the JIRA application by editing the standalone application, which is an exploded WAR stored in `JIRA/atlassian-jira`.

1. If you are using the Crowd WAR distribution, then you will need to get the CROWD client libraries from the standalone distribution, available on our download site.
2. Copy the Crowd client libraries and configuration files to JIRA:

Copy From	Copy To
CROWD/client/crowd-integration-client-X.X.X.jar	JIRA/atlassian-jira/WEB-INF/lib
CROWD/client/conf/crowd.properties	JIRA/atlassian-jira/WEB-INF/classes

Duplicate Crowd Client libraries in your classpath
You need to delete the existing crowd-integration-clien-X.X.X.jar file from JIRA's WEB-INF/lib directory and replace it with CROWD/client/crowd-integration-client-X.X.X.jar instead of just copying it over. Also, renaming the existing crowd-integration-client jar will not work as JIRA will start with duplicate Crowd Client libraries in its classpath.

3. If you are using **JIRA 3.11 or earlier**, you will need to remove the `seraph-0.7.12.jar` file from JIRA's `WEB-INF/lib/` directory and replace it with the following file:
http://repository.atlassian.com/maven2/com/atlassian/seraph/atlassian-seraph/0.10/atlassian-seraph-0.10.jar
4. If you are using **JIRA 3.12.2 or earlier**, you will need to update JIRA's xfire libraries:
   - Remove the `xfire-all-1.2.1.jar` file from JIRA's `WEB-INF/lib/` directory.
   - Copy the following two files from Crowd's `client/lib/` directory to JIRA's `WEB-INF/lib/` directory:
     - `xfire-aegis-1.2.6.jar`
     - `xfire-core-1.2.6.jar`
5. Replace JIRA's cache configuration file:

Copy From	Replace File
CROWD/client/conf/crowd-ehcache.xml	JIRA/atlassian-jira/WEB-INF/classes/crowd-ehcache.xml

6. Edit `JIRA/atlassian-jira/WEB-INF/classes/crowd.properties`. Change the following properties:

Key	Value
application.name	`jira` The application name must match the name that you specified when you defined the application in Crowd (see *Step 1* above).
application.password	The password must match the one that you specified when you defined the application in Crowd (see *Step 1* above).
crowd.server.url	http://localhost:8095/crowd/services/ If your Crowd server's port is configured differently from the default (i.e. 8095), set it accordingly.
session.validationinterval	Set to 0, if you want authentication checks to occur on each request. Otherwise set to the number of minutes between request to validate if the user is logged in or out of the Crowd SSO server. Setting this value to 1 or higher will increase the performance of Crowd's integration.

You can read more about optional settings in the crowd.properties file.

2.2 Configure JIRA to use Crowd's Authenticator

Now that the Crowd client libraries exist, we need to configure JIRA to use them.

1. Edit the JIRA config file `JIRA/atlassian-jira/WEB-INF/classes/osuser.xml`. Comment out any existing authentication providers and uncomment/insert the Crowd providers:

```
<!-- This is where JIRA's credentials checking can be configured.  For instance, see
http://www.atlassian.com/software/jira/docs/latest/ldap.html -->
<opensymphony-user>
    <authenticator class="com.opensymphony.user.authenticator.SmartAuthenticator" />

<!-- You will need to uncomment the Crowd providers below to enable Crowd integration -->
    <provider class="com.atlassian.crowd.integration.osuser.CrowdCredentialsProvider"/>
    <provider class="com.atlassian.crowd.integration.osuser.CrowdAccessProvider"/>
    <provider class="com.atlassian.crowd.integration.osuser.DelegatingProfileProvider">
        <property name="provider-1"
>com.atlassian.crowd.integration.osuser.CrowdProfileProvider</property>
        <property name="provider-2"
>com.atlassian.jira.user.ExternalEntityJiraProfileProvider</property>
        <property name="provider-2-exclusive-access">true</property>
    </provider>

<!-- CROWD:START  - The providers below here will need to be commented out for Crowd
integration -->
<!--
    <provider class="com.atlassian.core.ofbiz.osuser.CoreOFBizCredentialsProvider">
        <property name="exclusive-access">true</property>
    </provider>

    <provider class="com.opensymphony.user.provider.ofbiz.OFBizProfileProvider">
        <property name="exclusive-access">true</property>
    </provider>

    <provider class="com.opensymphony.user.provider.ofbiz.OFBizAccessProvider">
        <property name="exclusive-access">true</property>
    </provider>
-->
<!-- CROWD:END -->

</opensymphony-user>
```

2. View `JIRA/atlassian-jira/WEB-INF/classes/propertyset.xml`. If there is no entry for the CrowdPropertySet, add the following <propertyset> item at the end of the file as the last <propertyset> item:

```
<propertyset name="crowd" class="com.atlassian.crowd.integration.osuser.CrowdPropertySet"/>
```

3. At this stage, JIRA is set up for **centralised authentication**. If you wish, you can now enable **single sign-on (SSO)** to JIRA.

   Skip this step if you are using the JIRA NTLM plugin to enable SSO. Instead, follow the instructions on configuring JIRA for NTLM SSO.

   Edit `JIRA/atlassian-jira/WEB-INF/classes/seraph-config.xml`. Change the `authenticator` node to read:

```
<authenticator class="com.atlassian.crowd.integration.seraph.JIRAAuthenticator"/>
```

   JIRA's authentication and access request calls will now be performed using Seraph. Now when authentication or access request calls are performed versus the OSUser framework, the JIRA stack will call the Crowd providers and propertyset implementations.

2.3 Enable JIRA's 'External User Management'

Once the setup is complete, you can configure JIRA to allow external user management. Go to the JIRA Administration Console. In the General Configuration section, turn '**External user management**' and '**External password management**' on or off. (See the JIRA Administrator's Guide for details).

JIRA with external user management ON:

This is recommended, because it allows you to use Crowd's powerful cross-directory user administration features.

Crowd allows you to automatically assign new users to groups. You can define default groups for each directory. Every new user automatically becomes a member of these groups.

If you turn external user management on, the following functions can no longer be performed from within the JIRA administration interface: adding users, adding groups, editing users, editing groups.

If you are using Crowd 1.1.1 or earlier, you must turn external user management **on** in JIRA.

JIRA with external user management OFF:

This means that you can allow signup via JIRA, and you can manage your users within JIRA. Changes will flow through to Crowd.

JIRA has an automatic group membership feature. This means that any new user added through JIRA will automatically be a member of all groups which have the **JIRA Users** permission. In this way, you can ensure that a new user is automatically added to several groups when they sign up with JIRA.

Any group or user changes will cascade to all directories assigned to the JIRA application in Crowd. For example, if user 'jbloggs' registers in JIRA, 'jbloggs' will be added to every Crowd directory linked with the JIRA application.

2.4 (Optional) Tune the Cache

Enabling caching on the Crowd server: When using the Atlassian-User and Crowd framework together with JIRA, it is highly recommended that caching be enabled on the Crowd server. Multiple redundant calls to the Atlassian-User framework are made on any given request. These results can be stored locally between calls by enabling caching via the Crowd Options menu. Note that this caching on the Crowd server is enabled by default.

Enabling application caching for JIRA: If application caching is enabled for JIRA, JIRA will obtain all necessary information for the period specified by the cache configuration. See Configuring Caching for an Application. If a change or addition occurs to Crowd users, groups and roles, these changes will not be visible in JIRA until the cache expires for that specific item, i.e. for the particular user, group or role.

From JIRA 3.13, the default cache is two hours. In earlier versions, the default value for the application cache is 5 minutes (300 seconds) — increasing this to one or two hours (3600 or 7200 seconds) will improve the performance of your JIRA site.

2.5 (Optional) Disable the Auto-Complete Function in JIRA's User Picker

To improve performance on page-loading in JIRA, we recommend that you disable the auto-complete function in JIRA's 'User Picker' popup screens. Follow the instructions in the JIRA documentation.

More information: In our experience, disabling this feature in JIRA helps performance for customers with extremely large user bases. If you leave this feature enabled and have adequate performance results in JIRA, feel free to leave it enabled.

See Crowd in Action

- You should now be able to login using users belonging to the `jira-users` group. Try adding a user to the group using Crowd — you should be able to login to JIRA using this newly created user. That's **centralised authentication** in action!
- If you have enabled SSO, you can try adding the *JIRA Directory in Crowd* and `jira-administrators` group to the *crowd* application (see Mapping a Directory to an Application and Specifying which Groups can access an Application). This will allow JIRA administrators to log in to the Crowd Administration Console. Try logging in to Crowd as a JIRA administrator, and then point your browser at JIRA. You should be logged in as the same user in JIRA. That's **single sign-on** in action!

Known Limitations

A problem occurs in JIRA if a user is removed after that user has participated in an issue i.e. if JIRA refers to the user. If the user is internally managed by JIRA, JIRA will prevent the removal of the user but if the user is managed by an external system such as Crowd, JIRA will throw a `DataAccessException`.

The current workaround for this is to deactivate the user's account (by removing them from the `jira-users` group). This issue can be tracked here: http://jira.atlassian.com/browse/CWD-202

RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application

- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Configuring JIRA for NTLM SSO

> **JIRA NTLM plugin not officially supported by Atlassian**
> The JIRA NTLM plugin was written by a third party. Atlassian does not officially support this SSO plugin. Crowd implements SSO itself, but does not natively support the NTLM protocol without this plugin. Therefore, NTLM is not a pre-requisite for the use of Crowd. The Atlassian Crowd team will do our best to advise on any Crowd integration problems. Please refer to the plugin documentation for installation instructions and further support.

Out of the box, JIRA does not support Single Sign On (SSO) functionality. This page describes how to set up JIRA with NTLM SSO functionality using the JIRA NTLM plugin, Crowd, and Active Directory (AD) as your LDAP user repository.

Summary

The JIRA NTLM plugin enables the following authentication scenario:

- A user in a Windows domain logs into the Windows network, using their Active Directory username/password.
- Then, when they open JIRA in an Internet Explorer browser, they are seamlessly logged into JIRA.

The Crowd component then allows you to manage all users and groups in Active Directory. Crowd automatically ensures that users and groups are synchronised between AD and JIRA. For example, if a user/group is added/deleted from AD it will be automatically added/deleted from JIRA.

Components

JIRA NTLM plugin	NTLM is the protocol used by Windows for authentication. The JIRA NTLM plugin takes care of the Windows domain / Active Directory login to JIRA. You must be running a Windows Domain Controller with accounts set up in AD in order to use this plugin. If NTLM authentication is not available, the plugin allows standard form-based login to JIRA. **Note:** This plugin is not officially supported by Atlassian.
Crowd	Crowd takes care of the synchronisation of users/groups between Active Directory and JIRA. You will need to create an SSL connection between Crowd and the AD server if you would like to create users through Crowd. AD will not allow Crowd to add users or change their passwords unless the communication occurs over a secure connection.
Active Directory (AD) on Windows 2003 Server	Active Directory (AD) on Windows 2003 Server — you must already have an AD instance set up and running with a domain controller.
JIRA	The machine running JIRA must be part of the Windows domain or installed on the same box as the domain controller.

Steps

1. Back up your entire JIRA installation directory and run an XML backup of your data.
2. Download the JIRA NTLM plugin.
3. Read the `README` file included in the plugin zip file, and then follow the instructions in the `INSTALL` file to install the plugin.
4. In the `ntlm_ldap.properties` file, insert the appropriate LDAP and Domain Controller information along with other parameters.
5. Install and configure Crowd.
6. Create a directory in Crowd for the AD LDAP server.
7. Create the JIRA application in Crowd and configure Crowd and JIRA to talk to each other, as described in Integrating Crowd with Atlassian JIRA.

> When following the above instructions, **do not** change the `seraph-config.xml` file to enable Crowd's SSO functionality. (I.e. don't change the `authenticator` node to read `<authenticator class="com.atlassian.crowd.integration.seraph.JIRAAuthenticator"/>`. Instead of Crowd's SSO authentication, we'll be using the JIRA NTLM plugin.

8. In AD, create the groups **jira-users**, **jira-developers**, and **jira-administrators**. They should then appear in Crowd.
9. In AD, create an admin user and make them a member of the above three groups in AD.
10. Create any additional groups that you would like in AD.
11. Log into the Windows domain using your desktop login and then open JIRA in an Internet Explorer browser. You should be logged in automatically.

Additional Crowd Performance Tips

- Change the default cache setting timeout in the file <JIRA>\WEB-INF\classes\crowd-ehcache.xml. For performance reasons, increase the object caching to 7,200 seconds (2 hours):
  timeToIdleSeconds="7200" timeToLiveSeconds="7200".
  This reduces the frequency of the requests from Crowd to the LDAP server when changes to LDAP objects (such as a group name or user attribute) are made, thus reducing the performance overhead.

- Turn on the '**Use Paged Results**' option in the directory connector tab for the directory you've set up in Crowd.

Integrating Crowd with Acegi Security

Crowd provides **centralised authentication** and **single sign-on** connectors for the web security framework Acegi. Acegi provides a modular and highly configurable approach to authentication and authorisation for J2EE applications.

If your web application already makes use of the Acegi framework for authentication and authorisation, you can use the Crowd Acegi connector to allow your application to easily delegate authentication and authorisation requests to Crowd.

The connectors are available with **Crowd 1.2 and later** and have been developed and tested with Acegi 1.0.5.

Please consult the Acegi quick start guide or reference guide for a thorough insight into the Acegi framework. You might also find useful information in our Crowd-Acegi integration tutorial.

This guide assumes developer-level knowledge and an Acegi-based web application
This guide is for developers rather than administrators. This guide assumes you have **Crowd 1.5.1** or later installed and that you want to integrate your Acegi-based web application with Crowd's security server. The documentation below describes how to integrate Crowd with your own application that uses the Acegi framework. It assumes you already use Acegi in your application. If you need help integrating the Acegi framework with your web application, have look at some of the Acegi documentation.

Spring Security 2
If you're working with Spring Security, we have a separate tutorial.

Prerequisites

1. Download and configure Crowd. Refer to the Crowd Installation Guide for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.
2. Have your Acegi-based custom application ready for tweaking. We will refer to your custom application as '**AcegiApp**'.

Step 1. Configuring Crowd to Talk to your Acegi Application

Crowd needs to be aware that AcegiApp will be making authentication requests to Crowd. In brief, you will need to do the following:

1. Add the AcegiApp application to Crowd.
2. Add and configure the directories visible to AcegiApp.
3. Add and map the groups which are allowed to authenticate with AcegiApp.

Please see Adding an Application for a detailed guide.

Step 2. Installing the Crowd Acegi Connector

2.1 Adding the Crowd Acegi Connector to your Acegi Application

You will need to add the Crowd Acegi connector library and its associated dependencies to your Acegi application. You can do this manually by copying over the JAR files to your Acegi application or, if your Acegi application is a Maven project, you can add the Crowd Acegi connector as a project dependency. Both methods are described below.

2.1.1 Manually Adding the Crowd Acegi Connector Libraries

Follow *either* 2.1.1 *or* 2.1.2 (not both).

Copy the Crowd integration libraries and configuration files. This is described in the Client Configuration documentation. You will need to copy at least the following file to your Acegi application:

Copy From	Copy To
CROWD/client/crowd-integration-client-X.X.X.jar	AcegiApp/WEB-INF/lib
CROWD/client/lib/*.jar	AcegiApp/WEB-INF/lib

2.1.2 Adding the Crowd Acegi Connector as a Maven Dependency

Follow *either* 2.1.1 *or* 2.1.2 (not both).

The page Maven 2 integration does not exist.

See more information on Maven 2 integration.

2.2 Adding the Cache Configuration File

Copy the following file into your application's classpath:

Copy From	Copy To
CROWD/client/conf/crowd-ehcache.xml	AcegiApp/WEB-INF/classes/crowd-ehcache.xml

This file can be tweaked to change the cache behaviour.

2.3 Configuring the Crowd Acegi Connector Properties

The Crowd Acegi connector needs to be configured with the details of the Crowd server.

1. Copy the default `crowd.properties` file to the classpath of your Acegi application:

Copy From	Copy To
CROWD/client/conf/crowd.properties	AcegiApp/WEB-INF/classes

2. Edit the `crowd.properties` and populate the following fields appropriately:

Key	Value
application.name	Same as application name defined when adding the application to Crowd in Step 1.
application.password	Same as application password defined when adding the application to Crowd in Step 1.
crowd.server.url	http://localhost:8095/crowd/services/
session.validationinterval	This is the time interval between requests which validate whether the user is logged in or out of the Crowd SSO server. Set to 0, if you want authentication checks to occur on each request. Otherwise set to the number of minutes you wish to wait between requests. Setting this value to 1 or higher will increase the performance of Crowd's integration.

You can read more about the crowd.properties file.

Step 3. Configuring your Acegi Application to Use the Crowd Acegi Connector

There are two ways you can integrate your application with Crowd:

- **Centralised user management**: The user repository available to your application will be the user repository allocated to your application via Crowd. This means that your application will use the centralised user repository for retrieving user details as well as performing authentication.
- **Single sign-on**: In addition to centralised authentication, SSO will be available to your application. If any other SSO-enabled applications (such as JIRA, Confluence, or your own custom applications) are integrated with Crowd, then SSO behaviour will be established across these applications. If you sign in to one application, you are signed in to all applications. If you sign out of one application, you are signed out of all applications.

First, you will need to add the Crowd client application context to wire up the Crowd beans that manage the communication to Crowd. You can do this by including the `applicationContext-CrowdClient.xml` Spring configuration file, found in `crowd-integration-client.jar`. For example, if you are configuring Spring using a context listener, you can add the following parameter in your your Acegi application's `WEB-INF/web.xml`:

```
1.<context-param>
2.          <param-name>contextConfigLocation</param-name>
3.          <param-value>
4.                    ...
5.                    classpath:/applicationContext-CrowdClient.xml
6.                    ...
7.          </param-value>
8.</context-param>
```

Next, open the `applicationContext.xml` file relevant to your application, which contains the Acegi configuration. This is the file in your application that defines the Acegi beans. You are likely to have a bean configuration similar to this snippet:

```
<bean id="filterChainProxy" class="org.acegisecurity.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
        <value>
            CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
            PATTERN_TYPE_APACHE_ANT
            /images/**=#NONE#
            /scripts/**=#NONE#
            /styles/**=#NONE#

/**=httpSessionContextIntegrationFilter,logoutFilter,authenticationProcessingFilter,securityContextHo
        </value>
    </property>
</bean>
```

3.1 Configuring Centralised User Management

Perform the following updates to your Acegi Spring configuration:

1. Add the definition of the CrowdUserDetailsService:

```
<bean id="crowdUserDetailsService" class=
"com.atlassian.crowd.integration.acegi.user.CrowdUserDetailsServiceImpl">
        <property name="authenticationManager" ref="crowdAuthenticationManager"/>
        <property name="groupMembershipManager" ref="crowdGroupMembershipManager"/>
        <property name="userManager" ref="crowdUserManager"/>
        <property name="authorityPrefix" value="ROLE_"/>
</bean>
```

2. Add the definition of the RemoteCrowdAuthenticationProvider:

```
<bean id="crowdAuthenticationProvider" class=
"com.atlassian.crowd.integration.acegi.RemoteCrowdAuthenticationProvider">
                <constructor-arg ref="crowdAuthenticationManager"/>
                <constructor-arg ref="httpAuthenticator"/>
                <constructor-arg ref="crowdUserDetailsService"/>
</bean>
```

3. Update the definition of your AuthenticationManager / ProviderManager to use the CrowdAuthenticationProvider. If you need multiple authentication providers, you can append the CrowdAuthenticationProvider to your list.

```
<bean id="authenticationManager" class="org.acegisecurity.providers.ProviderManager">
        <property name="providers">
                <list>
                        <ref local="crowdAuthenticationProvider"/>
                        ....
                </list>
        </property>
</bean>
```

Further extensions

- If you have an existing user data model, then you can extend or wrap the CrowdDetailsService to cater for user objects within your application domain.
- If you require users within Crowd to be created in your application's persistence model so that you can store application-specific user data, you can extend the CrowdAuthenticationProvider to create records for successfully authenticated Crowd users.

Crowd's remote API
We recommend that applications do not store the Crowd users locally. Rather, applications should query users via Crowd's [remote API].

3.2 Configuring Single Sign-On (SSO)

SSO is optional and requires centralised user management
Single sign-on is optional. If you wish to configure SSO you must first configure centralised user management as described in step 3.1 above.

Perform the following additional updates to your Acegi Spring configuration:

1. Update the definition of the AuthenticationProcessingFilter to use the CrowdAuthenticationProcessingFilter:

```
<bean id="authenticationProcessingFilter" class=
"com.atlassian.crowd.integration.acegi.CrowdSSOAuthenticationProcessingFilter">
        <property name="httpAuthenticator" ref="httpAuthenticator"/>
        <property name="authenticationManager" ref="authenticationManager"/>
        <property name="filterProcessesUrl" value="/console/j_security_check"/>
        <property name="authenticationFailureUrl" value="/login.jsp?error=true"/>
        <property name="defaultTargetUrl" value="/"/>
        ...
</bean>
```

2. Add the definition of the CrowdLogoutHandler:

```
<bean id="crowdLogoutHandler" class=
"com.atlassian.crowd.integration.acegi.CrowdLogoutHandler">
        <property name="httpAuthenticator" ref="httpAuthenticator"/>
</bean>
```

3. Update the definition of the LogoutFilter to use the CrowdLogoutHandler:

```
<bean id="logoutFilter" class="org.acegisecurity.ui.logout.LogoutFilter">
          <constructor-arg value="/index.jsp"/>
          <constructor-arg>
                    <list>
                              ...
                              <ref bean="crowdLogoutHandler"/>
                              <bean class=
"org.acegisecurity.ui.logout.SecurityContextLogoutHandler"/>
                    </list>
          </constructor-arg>
          <property name="filterProcessesUrl" value="/logout.jsp"/>
 </bean>
```

Step 4. Restarting your Acegi Application

Bounce your application. You should now have centralised authentication and single sign-on with Crowd.

Authorisation

For the purposes of Crowd integration with Acegi, you should map Acegi's roles to Crowd's groups. To put it another way: in order to use Acegi's authorisation features, users in Crowd will have their Acegi roles specified by their group names.

For example if user 'admin' is in the 'crowd-admin' group, then the user 'admin' will be authorised to view pages restricted to the 'crowd-admin' role in Acegi.

```
<!-- authorisation -->
<bean id="filterInvocationInterceptor" class=
"org.acegisecurity.intercept.web.FilterSecurityInterceptor">
        <property name="authenticationManager" ref="authenticationManager"/>
        <property name="accessDecisionManager" ref="accessDecisionManager"/>
        <property name="objectDefinitionSource">
                 <value>
                          CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
                          PATTERN_TYPE_APACHE_ANT
                          /console/secure/**=ROLE_crowd-admin
                          /console/user/**=IS_AUTHENTICATED_FULLY
                 </value>
        </property>
</bean>

<bean id="accessDecisionManager" class="org.acegisecurity.vote.AffirmativeBased">
        <property name="allowIfAllAbstainDecisions" value="false"/>
        <property name="decisionVoters">

                 <list>
                          <bean class="org.acegisecurity.vote.RoleVoter"/>
                          <bean class="org.acegisecurity.vote.AuthenticatedVoter"/>
                 </list>
        </property>
</bean>
```

RELATED TOPICS

- Integrating AppFuse - a Crowd-Acegi Integration Tutorial

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application

- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating AppFuse - a Crowd-Acegi Integration Tutorial

AppFuse provides a sweet starting point for developing web applications. You choose the frameworks, AppFuse generates the skeleton application.

At its core, the web security of AppFuse 2.0.1 and earlier applications relies on the modular and extensible Acegi authentication framework. In this tutorial, we look at a basic integration of Crowd with Acegi, using an application generated by AppFuse.

If you're working with AppFuse 2.0.2 or later, it uses Spring Security instead of Acegi. Please see our separate tutorial.

This tutorial assumes you have installed Crowd 1.5.1 or later.

Step 1. Get AppFuse

In this tutorial, we will be using the Struts2-basic archetype to create the project, but the other types should be similar. For more information, consult the AppFuse quickstart guide. In particular, it outlines the database requirements for AppFuse.

1. Create the project.

```
1.mvn archetype:create -DarchetypeGroupId=org.appfuse.archetypes \
2.-DarchetypeArtifactId=appfuse-basic-struts \
3.-DremoteRepositories=http://static.appfuse.org/releases -DarchetypeVersion=2.0 \
4.-DgroupId=com.mycompany.app -DartifactId=myproject
```

2. Since we will be editing the core Acegi configuration, we will need the full source code of the application.

```
1.cd myproject
2.mvn appfuse:full-source
```

3. Build it.

```
1.mvn clean install
```

4. Run it.

```
1.mvn jetty:run-war -Dmaven.test.skip
```

5. Play with it.

```
1.http://localhost:8080/
```

6. Shut it down.

```
1.ctrl+c
```

Step 2. Let Crowd Know about AppFuse

Add `appfuse` as an application via the Crowd Console. See Adding an Application for more information.

Step 3. Add the Crowd Acegi Connector to AppFuse

Open up the `pom.xml` and add the Crowd client libraries as a project dependency:

```
1.<dependencies>
2.     <dependency>
3.          <groupId>com.atlassian.crowd</groupId>
4.          <artifactId>crowd-integration-client</artifactId>
5.          <version>1.5.1</version>
6.     </dependency>
7....
8.</dependencies>
```

You will also need to create the file `myproject/src/main/resources/crowd.properties`:

```
1.application.name                          appfuse
2.application.password                      password
3.application.login.url                     http://localhost:8095/crowd/
4.crowd.server.url                          http://localhost:8095/crowd/services/
5.session.isauthenticated                   session.isauthenticated
6.session.tokenkey                          session.tokenkey
7.session.validationinterval                0
8.session.lastvalidation                    session.lastvalidation
```

In particular, the application name and password must match the values defined for the application added in Step 2.

Finally, copy the `STANDALONE/client/conf/crowd-ehcache.xml` to `myproject/src/main/resources/crowd-ehcache.xml`. This file defines the cache properties, such as cache timeouts, used when accessing data from the Crowd server.

Step 4. Hook Up Centralised Authentication

Before modifying the security configuration, you will need to add the Spring configuration file to wire up the Crowd client beans. Add the `applicationContext-CrowdClient.xml` configuration file to the list of `contextConfigLocations` in `WEB-INF/web.xml`:

```
01.<context-param>
02.          <param-name>contextConfigLocation</param-name>
03.          <param-value>
04.                    classpath:/applicationContext-resources.xml
05.                    classpath:/applicationContext-dao.xml
06.                    classpath:/applicationContext-service.xml
07.                    classpath*:/applicationContext.xml
08.                    classpath:/applicationContext-CrowdClient.xml
09.                    /WEB-INF/applicationContext*.xml
10.                    /WEB-INF/xfire-servlet.xml
11.                    /WEB-INF/security.xml
12.          </param-value>
13.</context-param>
```

AppFuse neatly stores all the Acegi configuration in `myproject/src/main/webapp/WEB-INF/security.xml`. In order to get centralised authentication, we will need to set up Acegi to use the wrapped authenticator class we just created. Edit the Acegi beans in `security.xml`:

1. Add the definition of the CrowdUserDetailsService:

```
1.<bean id="crowdUserDetailsService" class=
"com.atlassian.crowd.integration.acegi.user.CrowdUserDetailsServiceImpl">
2.                    <property name="authenticationManager" ref="crowdAuthenticationManager"
/>
3.                    <property name="groupMembershipManager" ref=
"crowdGroupMembershipManager"/>
4.                    <property name="userManager" ref="crowdUserManager"/>
5.                    <property name="authorityPrefix" value="ROLE_"/>
6.</bean>
```

2. Add the definition of the RemoteCrowdAuthenticationProvider which will delegate Acegi's authentication requests to Crowd:

```
1.<bean id="crowdAuthenticationProvider" class=
"com.atlassian.crowd.integration.acegi.RemoteCrowdAuthenticationProvider">
2.                    <constructor-arg ref="crowdAuthenticationManager"/>
3.                    <constructor-arg ref="httpAuthenticator"/>
4.                    <constructor-arg ref="crowdUserDetailsService"/>
5.</bean>
```

3. Replace the DaoAuthenticationProvider with our authenticator in the authentication manager:

```
01.<bean id="authenticationManager" class="org.acegisecurity.providers.ProviderManager">
02.          <property name="providers">
03.                    <list>
04.                              <ref local="crowdAuthenticationProvider"/>
05.                              <!--ref local="daoAuthenticationProvider"-->
06.                              <ref local="anonymousAuthenticationProvider"/>
07.                              <ref local="rememberMeAuthenticationProvider"/>
08.                    </list>
09.          </property>
10.</bean>
```

4. Now do a:

```
1.mvn jetty:run-war -Dmaven.test.skip
```

5. Head over to `http://localhost:8080/`.

You should now be able to authenticate the users in your Crowd repository that **meet all of the following conditions**:

- They are in a Crowd directory assigned to the AppFuse application in Crowd. See more information.
- They are in Crowd groups named `USER` and `ADMIN`. You will need to add these groups and assign the user as a member of the groups. These Crowd group names map to the Acegi authorisation roles defined in the AppFuse application.
- They are allowed to authenticate with the AppFuse application because EITHER they are in a group allowed to authenticate with Crowd see more OR their container directory allows all users to authenticate see more.

Congratulations. You have **centralised authentication**

Application-level centralised user management
One quirk you may notice is that you can't view the profile details of users who exist in Crowd, but did not exist in AppFuse prior to the Crowd integration. Although it's possible to authenticate a Crowd user 'dude' and still run AppFuse as 'dude', 'dude' will not be in AppFuse's local database. AppFuse makes use of a database-backed user management system. In order to achieve application-level **centralised user management**, AppFuse will need to delegate its calls to create, retrieve, update and delete users to Crowd via [Crowd's remote API]. This will prevent data redundancy and eliminate the hassle of data synchronisation. This is beyond the scope of this short tutorial.

Step 5. Hook Up Single Sign-On

Enabling single sign-on (SSO) requires a little more tweaking of the `security.xml`:

1. Change the default processing filter to Crowd's SSO filter:

```
1.<bean id="authenticationProcessingFilter" class=
"com.atlassian.crowd.integration.acegi.CrowdSSOAuthenticationProcessingFilter">
2.          <property name="httpAuthenticator" ref="httpAuthenticator"/>
3.          <property name="authenticationManager" ref="authenticationManager"/>
4.          <property name="authenticationFailureUrl" value="/login.jsp?error=true"/>
5.          <property name="defaultTargetUrl" value="/"/>
6.          <property name="filterProcessesUrl" value="/j_security_check"/>
7.          <property name="rememberMeServices" ref="rememberMeServices"/>
8.</bean>
```

2. Add the definition of the CrowdLogoutHandler:

```
1.<bean id="crowdLogoutHandler" class=
"com.atlassian.crowd.integration.acegi.CrowdLogoutHandler">
2.          <property name="httpAuthenticator" ref="httpAuthenticator"/>
3.</bean>
```

3. Update the definition of the LogoutFilter to use the CrowdLogoutHandler. You may need to uncomment the logout filter.

```
01.<bean id="logoutFilter" class="org.acegisecurity.ui.logout.LogoutFilter">
02.          <constructor-arg value="/index.jsp"/>
03.          <constructor-arg>
04.                    <list>
05.                              <ref bean="rememberMeServices"/>
06.                              <ref bean="crowdLogoutHandler"/>
07.                              <bean class=
"org.acegisecurity.ui.logout.SecurityContextLogoutHandler"/>
08.                    </list>
09.          </constructor-arg>
10.          <property name="filterProcessesUrl" value="/logout.jsp"/>
11.</bean>
```

4. If the logout filter is not defined in the filter invocation list, you will need to add it:

```
1.<bean id="filterChainProxy" class="org.acegisecurity.util.FilterChainProxy">
2.          <property name="filterInvocationDefinitionSource">
3.                    <value>
4.                              ...
5.
/**=httpSessionContextIntegrationFilter,logoutFilter,authenticationProcessingFilter,securityCo
6.                    </value>
7.          ...
```

5. Now repeat:

```
1.mvn jetty:run-war -Dmaven.test.skip=true
```

SSO will only work for users that are able to **authenticate** with both appplications and are **authorised** to use both applications. Try out the following:

- Log in to Crowd – you should be logged in to AppFuse.
- Log out of AppFuse – you should be logged out of Crowd.

- Log in to AppFuse; log out of Crowd; log in to Crowd as another user; refresh AppFuse – you should be logged in as the new user.

Congratulations, you have **SSO**

Integrating Crowd with Apache

Crowd provides a number of modules that allow you to configure Crowd to authenticate *HTTP Authentication* requests made to an Apache web server.

The following features are provided:

- Use Crowd to password-protect resources on your website.
- Configure website locations to restrict access to specific Crowd groups or users.

Note: These instructions assume some Unix system and Apache configuration knowledge.

SSO not supported
Currently, the Crowd-Apache integration does not support user single sign-on (SSO). Please refer to issue CWD-1440 for more details. If you like, you can add yourself as a watcher of the issue and vote for it to register your interest in this feature request.

Using Subversion under Apache?
Crowd's Subversion connector allows you to password-protect a Subversion repository and provide fine grained access by group or user. Read more.

Prerequisites

- Apache web server version 2.0 or above with the `mod_perl` module installed and configured.
- The following third-party Perl modules:
  - `SOAP::Lite` (v0.69 or greater required)
  - `Digest::SHA1`
  - `Error`
  - `Cache::Cache`

Installation and Configuration

The following instructions are for Unix systems. If you're running Apache on Windows, see the notes below.

Installing the Third-Party Perl Modules

Download the required Perl modules from CPAN using the links above and install them as follows:

```
1.tar xvzf  Cache-Cache-1.05.tar.gz
2.cd Cache-Cache-1.05
3.perl Makefile.PL
4.make
5.make install
```

See http://search.cpan.org/~jhi/perl-5.8.0/pod/perlmodinstall.pod for a detailed description of the various ways of installing Perl modules on your system.

Installing the Crowd Perl Modules

1. Download the three Crowd module files attached to this page:

Attached file	Description
Crowd-Apache-Connector-1.2.3.zip	Crowd authentication, authorisation and perl module for Apache 2

2. Extract the Crowd-Apache-Connection archive file and install the three modules using the same procedure as for the third party modules.

```
unzip Crowd-Apache-Connector-1.2.3.zip
cd Atlassian-Crowd-1.2.3/
perl Makefile.PL
make
make install
cd ../Apache-CrowdAuth-1.2.3/
perl Makefile.PL
make
make install
cd ../Apache-CrowdAuthz-1.2.3/
perl Makefile.PL
make
make install
```

Configuring Apache

Ensure that `mod_perl` is enabled. Your Apache config file should contain a line like the following:

```
1.LoadModule perl_module modules/mod_perl.so
```

Many common distributions of Apache come with `mod_perl` preconfigured.

Configuring Authentication

To tell Apache to use Crowd to authenticate requests for a particular location, edit the Apache config file to add the following commands to a `<Location>` or `<Directory>` section.

```
01.Alias /crowd/ "/var/crowd/"
02.<Directory "/var/crowd/">
03..
04..
05..
06.     AuthName crowd
07.     AuthType Basic
08.
09.     PerlAuthenHandler Apache::CrowdAuth
10.     PerlSetVar CrowdAppName appname
11.     PerlSetVar CrowdAppPassword apppassword
12.     PerlSetVar CrowdSOAPURL http://localhost:8095/crowd/services/SecurityServer
13.     PerlSetVar CrowdCacheEnabled on
14.     PerlSetVar CrowdCacheLocation /tmp/CrowdAuth
15.     PerlSetVar CrowdCacheExpiry 300
16.
17.     require valid-user
18..
19..
20..
21.</Directory>
```

Command	Explanation
`AuthName crowd`	Defines the realm of the authentication. This information is typically provided to the user in the dialogue box popped up by their browser. This *must* be a unique name for each crowd application
`AuthType Basic`	Tells Apache to use *basic* authentication. *Digest* authentication is not currently supported.
`PerlAuthenHandler Apache::CrowdAuth`	Tells Apache to delegate authentication to the CrowdAuth module
`PerlSetVar CrowdAppName`	Set the Application Apache should authenticate as
`PerlSetVar CrowdAppPassword`	Set the password for the Application
`PerlSetVar CrowdSOAPURL`	The URL of the Crowd SOAP service
`PerlSetVar CrowdCacheEnabled`	[optional] Controls whether CrowdAuth caches authentications locally to improve performance. Set to "on" or "off". Caching is "on" by default
`PerlSetVar CrowdCacheLocation`	[optional] The directory in which CrowdAuth's local cache is stored. Defaults to `/tmp/FileCache` if not set.

`PerlSetVar CrowdCacheExpiry`	[optional] The time (in seconds) before cached authentications in CrowdAuth's local cache expire. Defaults to 300 seconds (5 minutes)
`require valid-user`	Tells Apache that clients must provide a valid username/password to access the location

Configuring Authorisation

If you want to restrict access to a certain Directory or Location in your Apache configuration to a subset of Crowd users and/or groups, add the following lines to your configuration:

```
<Location URL_to_restrict>
    .
    .
    .
    PerlAuthzHandler Apache::CrowdAuthz

    PerlSetVar CrowdAllowedUsers johnh,kevinr
    PerlSetVar CrowdAllowedGroups developers,crowd-administrators,customers:r
    .
    .
    .
</Location>
```

Command	Explanation
`PerlAuthzHandler Apache::CrowdAuthz`	Tells Apache to use the `Apache::CrowdAuthz}` for authorisation
`PerlSetVar CrowdAllowedUsers johnh,kevinr`	Allow only the users `johnh` or `kevinr` to access the location
`PerlSetVar CrowdAllowedGroups developers,crowd-administrators,customers:r`	Allow only members of the `developers` or `crowd-administrators` groups to access the location. You can indicate that a group has read-only access to the location by appending ":r" to the group name. This is mainly useful for giving a group read-only access to a Subversion repository. Read-write access is the default privilege if ":r" is not specified.

Note:

- Typically, only one of the `CrowdAllowedUsers` or `CrowdAllowedGroups` would be needed for a particular location. You *can* define both. If you do, then access is granted if *either* is satisfied.
- If the `CrowdCacheEnabled` setting is on, then authorisation checks are cached in order to increase performance. This means that changes to group membership in Crowd may not be reflected immediately in user access.
- If you're using the `CrowdAllowedGroups` setting and your group names contain spaces, then surround the group names with quotes, for example:

  ```
  PerlSetVar CrowdAllowedGroups "Admin Staff,developers"
  ```

- Although the Apache Connector does not support Digest Authentication, the connection with Crowd can still be secured by using **https** to make the SOAP connections.

  ```
  PerlSetVar CrowdSOAPURL https://localhost:8095/crowd/services/SecurityServer
  ```

  For information on how to secure Crowd connections, refer to the documentation on configuring Crowd to work with SSL.

Troubleshooting

The `CrowdAuth` module logs detailed output if the Apache LogLevel parameter is set to `info` or `debug`. This can be useful in diagnosing problems.

Apache Log Error Message	Possible Cause and Next Steps
`CrowdAppName or CrowdAppPassword is not defined`	One or both of the `CrowdAppName` or `CrowdAppPassword` parameters is missing from the Apache config file.
`Failed to authenticate application`	The attempt to authenticate the application with Crowd failed. Check the values of the `CrowdAppName` or `CrowdAppPassword` parameters.
`Failed to authenticate principal`	Failed to authenticate a username/password pair provided by the client. This may just mean that the username or password supplied is incorrect. Note that `CrowdAuth` won't log *successful* authentications unless the `LogLevel` is `info` or above.
`User token not found in SOAP response for user <user>`	Internal SOAP protocol error.
`error 500...at CrowdAuth.pm..`	Indicates that Apache can't connect to the Crowd SOAP service.

`error 404...at CrowdAuth.pm...`	Indicates that the URL used to connect to the Crowd SOAP service is incorrect. Check the value of the `CrowdSOAPURL` parameter.
`failed to resolve handler `Apache::CrowdAuth': Can't locate Apache/CrowdAuth.pm ...`	The `CrowdAuth.pm` file isn't located on the Perl include path (or it has incorrect permissions).
`failed to resolve handler `Apache::CrowdAuth': Can't locate SOAP/Lite.pm...`	The `SOAP:Lite` module hasn't been installed.
`Can't locate object method "call" via package "SOAP::SOM" at ...`	This message indicates a missing or old installation of SOAP::Lite. Try installing (or reinstalling) version 0.69 SOAP:Lite. On Windows, you will get this error if you haven't manually upgraded the SOAP::Lite ppm (see below)
`Can't use string ("1") as an ARRAY ref while "strict refs" in use at (re_eval 116) line 1...`	This message is a result of using Perl 5.10 with SOAP::Lite 0.69. Upgrading SOAP::Lite to version 0.71 or greater should correct the problem.

Installing Perl, `mod_perl` and Perl Modules on Windows

Setting up `CrowdAuth` on an Apache instance running on Windows requires that some things be done differently. The following instructions assume you are using ActivePerl as your Perl environment.

- If you don't already have a Perl interpreter installed, you'll need one. The following instructions assume an install of ActiveState's ActivePerl. We *strongly* recommend running version 5.8.8 (codename 8XX) of ActivePerl rather than any newer version.
- Windows installations of Apache are less likely to come with `mod_perl` pre-installed. A Win32 version of `mod_perl` in PPM format is available here.
- The `.tar.gz` format used to distribute CrowdAuth (and other modules) is supported by most modern Windows archiving utilities (WinZip, for example).
- The `make` utility used to build the Perl modules is not part of a Windows. `nmake`, Microsoft's equivalent, is available as a self-extracting archive here.

Installing Perl Modules on Windows

The required modules (`Digest::SHA1`, `Error`, `Cache::FileCache`, `SOAP:Lite`) are available through the Perl Package Manager utility.

`CrowdAuth` needs a newer version of SOAP::Lite than is supplied with ActivePerl (0.69 vs 0.55). A prebuilt ppm of the correct version can be installed from the University of Winnipeg's repository using the following command:

```
C:\>ppm install http://theoryx5.uwinnipeg.ca/ppms/SOAP-Lite.ppd
```

Installing `Apache::CrowdAuth` on Windows

```
unzip Crowd-Apache-Connector-1.2.3.zip using Winzip or equivalent...
cd Atlassian-Crowd-1.2.3/
perl Makefile.PL
nmake
nmake install
cd ../Apache-CrowdAuth-1.2.3/
perl Makefile.PL
nmake
nmake install
cd ../Apache-CrowdAuthz-1.2.3/
perl Makefile.PL
nmake
nmake install
```

When editing the `httpd.conf` file and adding the `mod_perl.so` module to Apache, you may need to add the following line above the LoadModule line:

```
LoadFile "C:/Perl/bin/perl58.dll"
LoadModule perl_module modules/mod_perl.so
```

This LoadFile line points to the `perl58.dll` in your Perl install directory.

Related Topics

- Using the Application Browser
- Adding an Application

- Integrating Crowd with Atlassian Bamboo
- Integrating Crowd with Atlassian Confluence
  - Configuring Confluence for NTLM SSO
- Integrating Crowd with Atlassian CrowdID
- Integrating Crowd with Atlassian Crucible
- Integrating Crowd with Atlassian FishEye
  - Configuring FishEye 1.3.x to talk to Crowd
- Integrating Crowd with Atlassian JIRA
  - Configuring JIRA for NTLM SSO
- Integrating Crowd with Acegi Security
  - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
- Integrating Crowd with Apache
- Integrating Crowd with Jive Forums
  - Jive SSO
- Integrating Crowd with Spring Security
  - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
- Integrating Crowd with Subversion
- Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with Jive Forums

Jive Forums allows you to specify an implementation that provides authentication and authorisation external to the application. This document outlines how to integrate Crowd's authenticator with Jive Forums.

Support for Jive Forums version 5.5.13 only
Crowd provides centralised authentication and single sign-on (SSO) for Jive Forums version **5.5.13** only. Jive have announced that Jive Forums has evolved into a new product, Jive Social Business Software (SBS). We have no plans to update Crowd to support later versions of Jive Forums.

Prerequisites

1. Download and configure Crowd. Refer to the Crowd installation guide for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.
2. Install/configure Jive Forums. Refer to the relevant Jive Forums documentation for information regarding this installation process. The documentation is usually supplied with the software distribution. Do not attempt to use Crowd as the authentication system during the installation process (use the default authentication system for the installation process).

Step 1. Tell Crowd about Jive Forums

1.1 Prepare Crowd's Directory/Users for Jive Forums

The Jive Forums application will need to locate users from a directory configured in Crowd. You will need to set up a directory in Crowd for Jive. For more information on how to do this, see Adding a Directory. We will assume that the directory is called *Jive Forum Directory* for the rest of this document. It is possible to assign more than one directory for an application, but for the purposes of this example, we will use *Jive Forum Directory* to house Jive Forum users.

If you have an existing Jive Forums deployment and would like to import existing users into Crowd, use the Jive Importer tool by navigating **Users** > **Import Users** > **JIVE**. Select the *Jive Forum Directory* as the directory into which Jive Forum users will be imported. For details please see Importing Users from Jive Forums. If you are going to import users into Crowd, you need to do this now before you proceed any further.

1.2 Define the Jive Forums Application in Crowd

Crowd needs to be aware that the Jive Forums application will be making authentication requests to Crowd. We need to add the Jive Forums application to Crowd and map it to the *Jive Forums Directory:*

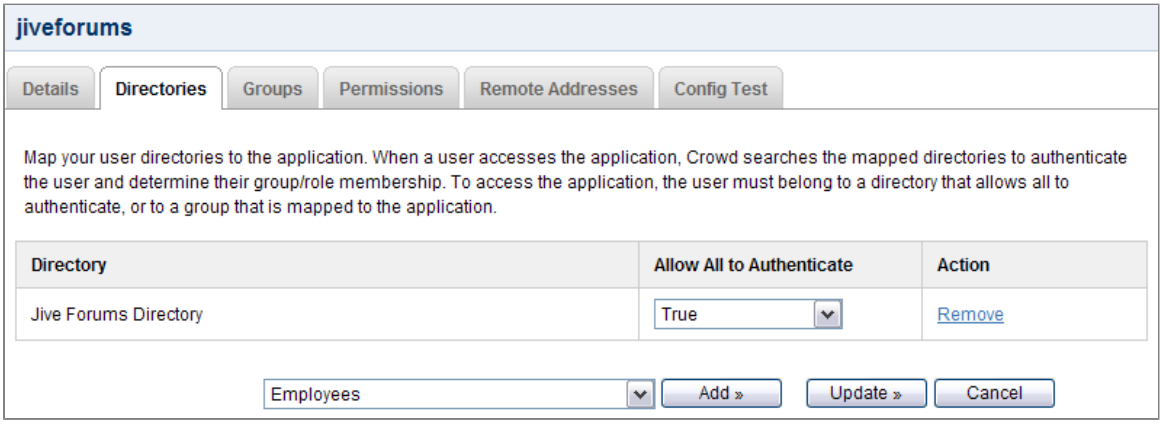
1. Log in to the Crowd Administration Console and navigate to **Applications** > **Add Application**.

2. Complete the **'Add Application' wizard** for the Jive Forums application. See the instructions. The **Name** and **Password** values you specify in the 'Add Application' wizard must match the **application.name** and **application.password** that you will set in the `JIVEFORUMS/WEB-INF/classes/crowd.properties` file. (See Step 2 below.)

1.3 Specify which Users can Log In to Jive Forums

Once Crowd is aware of the Jive Forums application, Crowd needs to know which users can authenticate (log in) to Jive Forums via Crowd. As part of the 'Add Application' wizard, you will set up your directories and group authorisations for the application. If necessary, you can adjust these settings after completing the wizard. Below are some examples.

You can either configure entire directories to authenticate or allow particular groups. In our example, we can simply allow the entire directory to authenticate:



Alternatively, we can use the **Groups** tab to restrict the application to only authenticate particular groups of users. For details please see Specifying which Groups can access an Application.

1.4 Specify the Address from which Jive Forums can Log In to Crowd

As part of the 'Add Application' wizard, you will set up Jive Forums's IP address. This is the address which Jive Forums will use to authenticate to Crowd. If necessary you can add a hostname, in addition to the IP address, after completing the wizard. See Specifying an Application's Address or Hostname.

Step 2. Tell Jive Forums about Crowd

2.1 Install the Crowd Client Libraries into the Jive Forums WebApp

Jive Forums may be deployed on an application server as a single WAR file or a an exploded WAR folder. For the rest of the installation process, we will assume that Jive Forums has been set up as an exploded war file. If you need Jive Forums to be installed as a single WAR file, simply expand the WAR to a directory, make the changes as described below, and zip up the directory to form the WAR file. We will refer to the root folder of the Jive Forums web-app as `JIVEFORUMS`.

1. Copy the Crowd integration libraries and configuration files (this is described in the Client Configuration documentation). This is summarised below:

Copy From	Copy To
CROWD/client/crowd-integration-client-X.X.X.jar	JIVEFORUMS/WEB-INF/lib
CROWD/client/lib/log4j-1.2.8.jar	JIVEFORUMS/WEB-INF/lib/
CROWD/client/lib/ehcache-1.2.3.jar	JIVEFORUMS/WEB-INF/lib/
CROWD/client/conf/crowd.properties	JIVEFORUMS/WEB-INF/classes/
CROWD/client/conf/crowd-ehcache.xml	JIVEFORUMS/WEB-INF/classes/

2. Replace the XFire libraries in your Jive Forums installation with the later version shipped with Crowd:
   - Remove all `xfire*.jar` files from your `JIVEFORUMS/WEB-INF/lib` folder.
   - Copy the XFire libraries from Crowd:

Copy From	Copy To
CROWD/client/xfire*.jar	JIVEFORUMS/WEB-INF/lib/

3. Examine the `JIVEFORUMS/WEB-INF/lib` folder and delete any duplicate JARs. Duplicate JARs represent common libraries used

by both the Crowd client and Jive Forums.

4. Edit `JIVEFORUMS/WEB-INF/classes/crowd.properties`. Change the following properties:

Key	Value
application.name	jiveforums
application.password	*set a password*

The **name** and **password** values must match those set when defining the application in Crowd (see Step 1 above).

You can read more about the crowd.properties file.

2.2 Configure Jive Forums to use Crowd's Authenticator

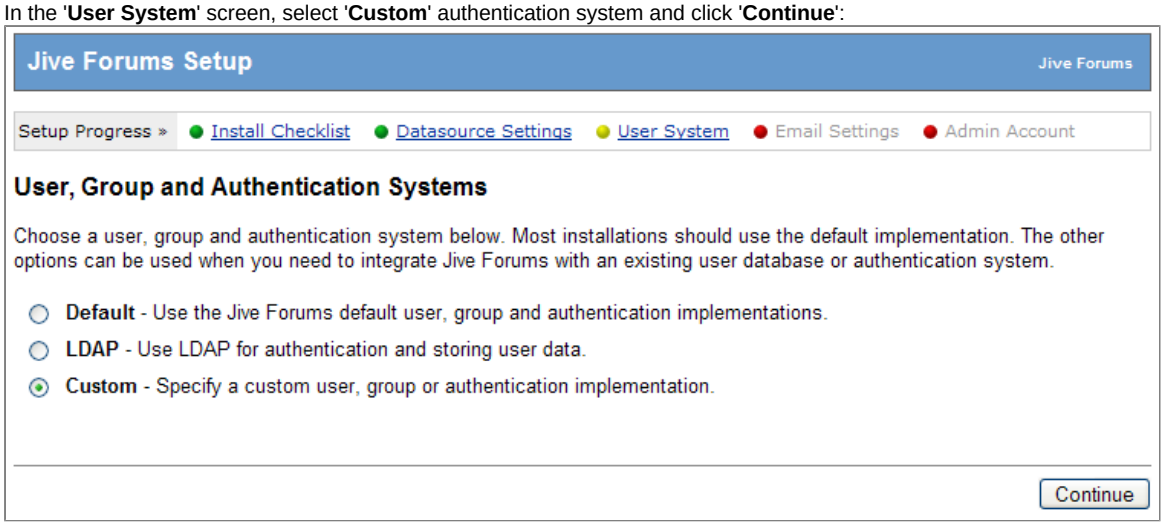
Crowd is now set up to provide authentication services to Jive. Now Jive needs to be set up to use Crowd's authenticator. There are a few ways of doing this. The most user-friendly method is outlined below:

1. In your `jiveHome` directory, edit a file named `jive_startup.xml`. Modify the `<setup>` node to be `false`:

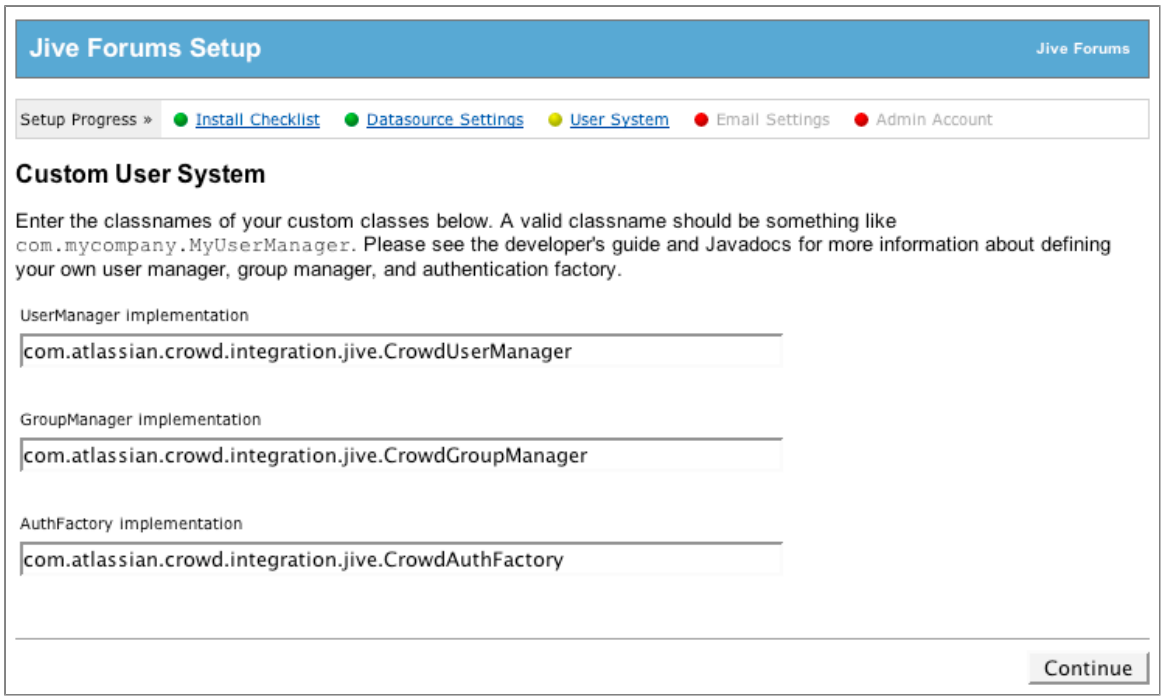
```
01.<jive>
02.     <!-- When setup is false, you can access the setup tool. -->
03.     <setup>false</setup>
04.     ...
05.     <!-- Allow SSO login for admins -->
06.     <admin>
07.          <tryAlternativeLogin>true</tryAlternativeLogin>
08.     </admin>
09.</jive>
```

   As the XML comment states, this lets us re-run Jive's setup.

2. Restart Jive Forums so that it picks up the changes.
3. View the Jive Forums site with a web browser - usually under the `/jiveforums` context-root. Jive will run the "Jive Forums Setup".
4. In the '**Install Checklist**' screen, click '**Continue**' to navigate through the setup process.
5. In the '**Datasource Settings**' screen, re-enter your database configuration details and click '**Continue**'.
6. In the '**User System**' screen, select '**Custom**' authentication system and click '**Continue**':



7. You should be at the '**Custom User System**' screen. Enter the following details which specify Crowd as the custom authenticator:



UserManager implementation:

```
1.com.atlassian.crowd.integration.jive.CrowdUserManager
```

GroupManager implementation:
If you would like Crowd to manage your user groups, add the following group manager:

```
1.com.atlassian.crowd.integration.jive.CrowdGroupManager
```

You can safely leave this field empty if you do not want Crowd to manage your groups.
AuthFactory implementation:

```
1.com.atlassian.crowd.integration.jive.CrowdAuthFactory
```

Click '**Continue**'.
If you have any errors at this stage, it is very likely that there is a classpath issue (eg. the Crowd client libraries aren't being properly loaded by Jive). Please read the documentation regarding Crowd Client Libraries for help identifying the problem.

8. In the '**Email Settings**' screen, re-enter your email configuration details and click '**Continue**'.
9. In the '**Admin Account Setup**' screen, *do not enter any details*. Click '**Skip this step**'.

> **Warning**
> The default administrator for Jive Forums is the user `admin`. This user will need to exist in your mapped directory (i.e. the *Jive Forums Directory*) in Crowd. Without this user, you will not be able to access the administration console of Jive Forums.

10. Bounce the server and test that Crowd is authenticating users for Jive. You can do this by creating users (users) via the Crowd Administration Console and verifying that they are able to log in to Jive Forums.

> **Jive Forums Documentation**
> For further information regarding Jive Forums Authentication Integration, check out the Jive Forums Documentation at
> http://www.jivesoftware.com/builds/docs/latest/documentation/developer-guide.html#userintegration

Check out the Jive SSO page for more details on Jive SSO Integration and corresponding use cases.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye

        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Jive SSO

This page details the nuts and bolts of Jive SSO. If you are having issues with Jive SSO, this page should be able to give you a better idea of what's going on behind the scenes and help you diagnose any common problems.

For Crowd-Jive integration, the incoming request must:

1. be authenticated with Crowd (have a Crowd SSO token in session or as a cookie)
2. be authenticated with Jive (have a CrowdAuthToken stored in HttpSession for Jive)

To authenticate with Crowd: simply log in to Crowd via any Crowd-SSO enabled application. This includes Jive's login page.

To authenticate with Jive: you need to be authenticated with Crowd as a user "allowed to be authenticated" by Jive. This means, the user must belong to a group or directory which Jive is authorised to authenticate. This user also needs to NOT be on any user/IP ban lists within the Jive application. The Crowd integration will honour the ban list. See note below.

Enumeration of Use Cases

User views Jive Forums and:

1. request is not authenticated with Crowd -> appears as guest user in Jive.
2. request is authenticated with Crowd, but user is not in directory/group allowed to authenticate with Jive -> appears as guest user in Jive.
3. request is authenticated with Crowd, user allowed to authenticate with Jive, user not on any ban list -> appears as logged-in user in Jive.
4. authenticated Jive user clicks logout from Jive -> user is logged out of Jive and Crowd.
5. authenticated Jive user logs out of Crowd using another SSO app -> user eventually times out of Jive.
6. request is authenticated with Crowd, user banned from logging into Crowd -> user appears as guest in Jive.
7. admin authenticated with Crowd and attempts to access Jive admin console -> admin appears logged in to Jive admin console.
8. authenticated Jive admin attempts to log out from Jive's admin console -> **admin is still logged in** (support issue filed with Jive Forums).
9. authenticated Jive admin attempts to log out from Jive Forums -> admin is logged out of Jive and Crowd.
10. request is authenticated with Crowd but user is banned from Jive Forums -> user is still authenticated with Crowd, but not allowed to log in to Jive Forums

Special Cases

- It is known that the "remember me" functionality of Jive will cease to function. This has been intentionally disabled. Jive's "remember me" functionality will need to be replaced by a more general "remember me" from within Crowd. Once this is implemented in Crowd, the Jive integration libraries can utilise Crowd's "remember me", so that "remember me" is centralised.

- It is recommended that admins do not use ban lists. Rather, you should manage access control based on Crowd's groups. So it's best to disable Ban Users from within Ban Settings inside the Jive admin console. There is nothing wrong with using ban lists, as they will be honoured by the Crowd-Jive integration libraries. So they will make it hard for a banned user to switch to a non-banned user. The only way a banned Jive user, authenticated with Crowd for Jive, will be able to switch to a different user that Jive will pick up, is when the Jive's Crowd authentication cache clears, so that Jive recognises a new user is signing in. This is because there is no way to log out a banned user from Jive, as they will always appear to be "guest". So basically, if you have users with multiple

identities, if one is banned and attempts to log in, the user will have to wait until the client cache is cleared before he/she can log in with a different identity. Note: it's easy for non-banned users to switch identities as the client authentication cache is cleared when they click "logout" from within Jive.

Related Topics

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with Spring Security

Crowd provides **centralised authentication** and **single sign-on** connectors for the web security framework Spring Security. Spring Security provides a modular and highly configurable approach to authentication and authorisation for J2EE applications.

Spring Security was formerly known as Acegi
There is a separate tutorial for integrating Acegi with Crowd.

If your web application already makes use of the Spring Security framework for authentication and authorisation, you can use the Crowd Spring Security connector to allow your application to easily delegate authentication and authorisation requests to Crowd.

The connector is available with **Crowd 1.6 and later** and has been developed and tested with Spring Security 2.0.4.

Please consult the Spring Security suggested steps or reference guide for a thorough insight into the Spring Security framework. You might also find useful information in our Appfuse integration tutorial.

This guide assumes developer-level knowledge and a Spring Security-based web application
This guide is for developers rather than administrators. This guide assumes you have **Crowd 1.6** or later installed and that you want to integrate your Spring Security-based web application with Crowd's security server. The documentation below describes how to integrate Crowd with your own application that uses the Spring Security framework. It assumes you already use Spring Security in your application. If you need help integrating the Spring Security framework with your web application, have look at some of the Spring Security documentation.

Prerequisites

1. Download and configure Crowd. Refer to the Crowd Installation Guide for detailed information on how to do this. We will refer to the Crowd root folder as `CROWD`.
2. Have your Spring Security-based custom application ready for tweaking. We will refer to your custom application as '**SpringSecApp**'.

Step 1. Configuring Crowd to Talk to your Spring Security Application

Crowd needs to be aware that SpringSecApp will be making authentication requests to Crowd. In brief, you will need to do the following:

1. Add the SpringSecApp application to Crowd.
2. Add and configure the directories visible to SpringSecApp.
3. Add and map the groups which are allowed to authenticate with SpringSecApp.

Please see Adding an Application for a detailed guide.

Step 2. Installing the Crowd Spring Security Connector

2.1 Adding the Crowd Spring Security Connector to your Spring Security Application

You will need to add the Crowd Spring Security connector library and its associated dependencies to your Spring Security application. You can do this manually by copying over the JAR files to your Spring Security application or, if your Spring Security application is a Maven project, you can add the Crowd Spring Security connector as a project dependency. Both methods are described below.

2.1.1 Manually Adding the Crowd Spring Security Connector Libraries

Follow *either* 2.1.1 *or* 2.1.2 (not both).

Copy the Crowd integration libraries and configuration files. This is described in the Client Configuration documentation. You will need to copy at least the following file to your Spring Security application:

Copy From	Copy To
CROWD/client/crowd-integration-client-X.X.X.jar	SpringSecApp/WEB-INF/lib
CROWD/client/lib/*.jar	SpringSecApp/WEB-INF/lib

2.1.2 Adding the Crowd Spring Security Connector as a Maven Dependency

Follow *either* 2.1.1 *or* 2.1.2 (not both).

The page Maven 2 integration does not exist.

See more information on Maven 2 integration.

2.2 Adding the Cache Configuration File

Copy the following file into your application's classpath:

Copy From	Copy To
CROWD/client/conf/crowd-ehcache.xml	SpringSecApp/WEB-INF/classes/crowd-ehcache.xml

This file can be tweaked to change the cache behaviour.

2.3 Configuring the Crowd Spring Security Connector Properties

The Crowd Spring Security connector needs to be configured with the details of the Crowd server.

1. Copy the default `crowd.properties` file to the classpath of your Spring Security application:

Copy From	Copy To
CROWD/client/conf/crowd.properties	SpringSecApp/WEB-INF/classes

2. Edit the `crowd.properties` and populate the following fields appropriately:

Key	Value
application.name	Same as application name defined when adding the application to Crowd in Step 1.
application.password	Same as application password defined when adding the application to Crowd in Step 1.
crowd.server.url	http://localhost:8095/crowd/services/
session.validationinterval	This is the time interval between requests which validate whether the user is logged in or out of the Crowd SSO server. Set to 0, if you want authentication checks to occur on each request. Otherwise set to the number of minutes you wish to wait between requests. Setting this value to 1 or higher will increase the performance of Crowd's integration.

You can read more about the crowd.properties file.

Step 3. Configuring your Spring Security Application to Use the Crowd Spring Security Connector

There are two ways you can integrate your application with Crowd:

- **Centralised user management**: The user repository available to your application will be the user repository allocated to your application via Crowd. This means that your application will use the centralised user repository for retrieving user details as well as performing authentication.
- **Single sign-on**: In addition to centralised authentication, SSO will be available to your application. If any other SSO-enabled applications (such as JIRA, Confluence, or your own custom applications) are integrated with Crowd, then SSO behaviour will be established across these applications. If you sign in to one application, you are signed in to all applications. If you sign out of one application, you are signed out of all applications.

First, you will need to add the Crowd client application context to wire up the Crowd beans that manage the communication to Crowd. You can do this by including the `applicationContext-CrowdClient.xml` Spring configuration file, found in `crowd-integration-client.jar`. For example, if you are configuring Spring using a context listener, you can add the following parameter in your your Spring Security application's `WEB-INF/web.xml`:

```
1.<context-param>
2.          <param-name>contextConfigLocation</param-name>
3.          <param-value>
4.                    ...
5.                   classpath:/applicationContext-CrowdClient.xml
6.                    ...
7.          </param-value>
8.</context-param>
```

3.1 Configuring Centralised User Management

The following sections assume that you have the Spring Security schema mapped to the `security` namespace. Perform the following updates to your Spring Security configuration:

1. Add the definition of the CrowdUserDetailsService:

```
1.<bean id="crowdUserDetailsService" class=
"com.atlassian.crowd.integration.springsecurity.user.CrowdUserDetailsServiceImpl">
2.                    <property name="authenticationManager" ref="crowdAuthenticationManager"
/>
3.                    <property name="groupMembershipManager" ref=
"crowdGroupMembershipManager"/>
4.                    <property name="userManager" ref="crowdUserManager"/>
5.                    <property name="authorityPrefix" value="ROLE_"/>
6.</bean>
```

2. Add the definition of the RemoteCrowdAuthenticationProvider:

```
1.<bean id="crowdAuthenticationProvider" class=
"com.atlassian.crowd.integration.springsecurity.RemoteCrowdAuthenticationProvider">
2.          <security:custom-authentication-provider />
3.          <constructor-arg ref="crowdAuthenticationManager"/>
4.                    <constructor-arg ref="httpAuthenticator"/>
5.                    <constructor-arg ref="crowdUserDetailsService"/>
6.</bean>
```

Further extensions

- If you have an existing user data model, then you can extend or wrap the CrowdDetailsService to cater for user objects within your application domain.
- If you require users within Crowd to be created in your application's persistence model so that you can store application-specific user data, you can extend the CrowdAuthenticationProvider to create records for successfully authenticated Crowd users.

Crowd's remote API
We recommend that applications do not store the Crowd users locally. Rather, applications should query users via Crowd's [remote API].

3.2 Configuring Single Sign-On (SSO)

SSO is optional and requires centralised user management
Single sign-on is optional. If you wish to configure SSO you must first configure centralised user management as described in step 3.1 above.

Perform the following additional updates to your Spring Security configuration:

1. Remove defaults from the `<http/>` element:

a. Remove the `auto-config` attribute and add an `entry-point-ref="crowdAuthenticationProcessingFilterEntryPoint"` attribute to the http element.
b. Remove the `<form-login>` element.
You should end up with a http element similar to this:

```
<http lowercase-comparisons="false" entry-point-ref=
"crowdAuthenticationProcessingFilterEntryPoint"> <!-- note: no auto-config attribute!
-->
        <!--intercept-url pattern="/images/*" filters="none"/>
        <intercept-url pattern="/styles/*" filters="none"/>
        <intercept-url pattern="/scripts/*" filters="none"/-->
        <intercept-url pattern="/admin/*" access="ROLE_ADMIN"/>
        <intercept-url pattern="/passwordHint.html*" access=
"ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
        <intercept-url pattern="/signup.html*" access=
"ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
        <intercept-url pattern="/a4j.res/*.html*" access=
"ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
        <!-- APF-737, OK to remove line below if you're not using JSF -->
        <intercept-url pattern="/**/*.html*" access="ROLE_ADMIN,ROLE_USER"/>
<!--  <form-login login-page="/login.jsp"
authentication-failure-url="/login.jsp?error=true"
login-processing-url="/j_security_check"/> -->
        <remember-me user-service-ref="userDao" key=
"e37f4b31-0c45-11dd-bd0b-0800200c9a66"/>
</http>
```

1. Change the default processing filter to Crowd's SSO filter by adding the following bean definitions:

```
<authentication-manager alias="authenticationManager"/>

<beans:bean id="crowdAuthenticationProcessingFilterEntryPoint" class=
"org.springframework.security.ui.webapp.AuthenticationProcessingFilterEntryPoint">
        <beans:property name="loginFormUrl" value="/login.jsp"/>
</beans:bean>

<beans:bean id="crowdAuthenticationProcessingFilter" class=
"com.atlassian.crowd.integration.springsecurity.CrowdSSOAuthenticationProcessingFilter">
        <custom-filter position="AUTHENTICATION_PROCESSING_FILTER"/>
        <beans:property name="httpAuthenticator" ref="httpAuthenticator"/>
        <beans:property name="authenticationManager" ref="authenticationManager"/>
        <beans:property name="authenticationFailureUrl" value="/login.jsp?error=true"/>
        <beans:property name="defaultTargetUrl" value="/"/>
        <beans:property name="filterProcessesUrl" value="/j_security_check"/>
</beans:bean>
```

2. Add the definition of the CrowdLogoutHandler:

```
<bean id="crowdLogoutHandler" class=
"com.atlassian.crowd.integration.springsecurity.CrowdLogoutHandler">
      <property name="httpAuthenticator" ref="httpAuthenticator"/>
</bean>
```

1. Add the definition of the CrowdLogoutHandler and add in a LogoutFilter that references it:

```
<beans:bean id="crowdLogoutHandler" class=
"com.atlassian.crowd.integration.springsecurity.CrowdLogoutHandler">
        <beans:property name="httpAuthenticator" ref="httpAuthenticator"/>
</beans:bean>

<beans:bean id="logoutFilter" class="org.springframework.security.ui.logout.LogoutFilter">
        <custom-filter position="LOGOUT_FILTER"/>
        <beans:constructor-arg value="/index.jsp"/>
        <beans:constructor-arg>
                <beans:list>
                        <beans:ref bean="crowdLogoutHandler"/>
                        <beans:bean class=
"org.springframework.security.ui.logout.SecurityContextLogoutHandler"/>
                </beans:list>
        </beans:constructor-arg>
        <beans:property name="filterProcessesUrl" value="/logout.jsp"/>
</beans:bean>
```

Step 4. Restarting your Spring Security Application

Bounce your application. You should now have centralised authentication and single sign-on with Crowd.

Authorisation

For the purposes of Crowd integration with Spring Security, you should map Spring Security's roles to Crowd's groups. To put it another way: in order to use Spring Security's authorisation features, users in Crowd will have their Spring Security roles specified by their group names.

For example if user 'admin' is in the 'crowd-admin' group, then the user 'admin' will be authorised to view pages restricted to the 'crowd-admin' role in Spring Security.

```
<!-- authorisation -->
<bean id="filterInvocationInterceptor" class=
"org.springframework.security.intercept.web.FilterSecurityInterceptor">
        <property name="authenticationManager" ref="authenticationManager"/>
        <property name="accessDecisionManager" ref="accessDecisionManager"/>
        <property name="objectDefinitionSource">
                <value>
                        CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
                        PATTERN_TYPE_APACHE_ANT
                        /console/secure/**=ROLE_crowd-admin
                        /console/user/**=IS_AUTHENTICATED_FULLY
                </value>
        </property>
</bean>

<bean id="accessDecisionManager" class="org.springframework.security.vote.AffirmativeBased">
        <property name="allowIfAllAbstainDecisions" value="false"/>
        <property name="decisionVoters">

                <list>
                        <bean class="org.springframework.security.vote.RoleVoter"/>
                        <bean class="org.springframework.security.vote.AuthenticatedVoter"
/>
                </list>
        </property>
</bean>
```

RELATED TOPICS

- Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
- Integrating Crowd with Acegi Security

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating AppFuse - a Crowd-Spring Security Integration Tutorial

AppFuse provides a sweet starting point for developing web applications. You choose the frameworks, AppFuse generates the skeleton application.

At its core, the web security of AppFuse 2.0.2+ applications relies on the modular and extensible Spring Security authentication framework. In this tutorial, we look at a basic integration of Crowd with Spring Security, using an application generated by AppFuse.

Spring Security was formerly known as Acegi

- The Acegi security framework changed its name to Spring Security with its 2.0 release.
- Appfuse 2.0.2 changed from Acegi to Spring Security for authentication. Earlier versions of Appfuse use Acegi.
- If you are working with Acegi in an earlier version of Appfuse, we have a separate tutorial.
- Crowd 1.6 and above provide support for both Spring Security and Acegi. Earlier versions of Crowd only supported Acegi.
- We recommend all new projects use Spring Security as it is being actively maintained.

Prerequisites
This tutorial assumes you have installed Crowd 1.6 or later and are using Appfuse 2.0.2 or later.

Step 1. Get AppFuse

In this tutorial, we will be using the Struts2-basic archetype to create the project, but the other types should be similar. For more information, consult the AppFuse quickstart guide. In particular, it outlines the database requirements for AppFuse.

1. Create the project.

```
1.mvn archetype:create -DarchetypeGroupId=org.appfuse.archetypes \
2.-DarchetypeArtifactId=appfuse-basic-struts \
3.-DremoteRepositories=http://static.appfuse.org/releases -DarchetypeVersion=2.0.2 \
4.-DgroupId=com.mycompany.app -DartifactId=myproject
```

2. Since we will be editing the core Spring Security configuration, we will need the full source code of the application.

```
1.cd myproject
2.mvn appfuse:full-source
```

3. Build it.

```
1.mvn clean install
```

4. Run it.

```
1.mvn jetty:run-war -Dmaven.test.skip
```

5. Play with it.

```
1.http://localhost:8080/
```

6. Shut it down.

```
1.ctrl+c
```

Step 2. Let Crowd Know about AppFuse

Add `appfuse` as an application via the Crowd Console. See Adding an Application for more information.

Step 3. Add the Crowd Spring Security Connector to AppFuse

Open up the `pom.xml` and add the Crowd client libraries as a project dependency:

```
1.<dependencies>
2.     <dependency>
3.          <groupId>com.atlassian.crowd</groupId>
4.          <artifactId>crowd-integration-client</artifactId>
5.          <version>1.6</version>
6.     </dependency>
7....
8.</dependencies>
```

You will also need to create the file `myproject/src/main/resources/crowd.properties`:

```
1.application.name                      appfuse
2.application.password                  password
3.application.login.url                 http://localhost:8095/crowd/
4.crowd.server.url                      http://localhost:8095/crowd/services/
5.session.isauthenticated               session.isauthenticated
6.session.tokenkey                      session.tokenkey
7.session.validationinterval            0
8.session.lastvalidation                session.lastvalidation
```

In particular, the application name and password must match the values defined for the application added in Step 2.

Finally, copy the `STANDALONE/client/conf/crowd-ehcache.xml` to `myproject/src/main/resources/crowd-ehcache.xml`. This file defines the cache properties, such as cache timeouts, used when accessing data from the Crowd server.

Step 4. Hook Up Centralised Authentication

Before modifying the security configuration, you will need to add the Spring configuration file to wire up the Crowd client beans. Add the `applicationContext-CrowdClient.xml` configuration file to the list of `contextConfigLocations` in `myproject/src/main/webapp/WEB-INF/web.xml`:

```
<context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
                classpath:/applicationContext-resources.xml
                classpath:/applicationContext-dao.xml
                classpath:/applicationContext-service.xml
                classpath*:/applicationContext.xml
                classpath:/applicationContext-CrowdClient.xml
                /WEB-INF/applicationContext*.xml
                /WEB-INF/xfire-servlet.xml
                /WEB-INF/security.xml
        </param-value>
</context-param>
```

AppFuse neatly stores all the Spring Security configuration in `myproject/src/main/webapp/WEB-INF/security.xml`. In order to get centralised authentication, we will need to set up Spring Security to use Crowd components for user information. Edit the beans in `security.xml`:

1. Add the definition of the CrowdUserDetailsService:

```
<beans:bean id="crowdUserDetailsService" class=
"com.atlassian.crowd.integration.springsecurity.user.CrowdUserDetailsServiceImpl">
        <beans:property name="authenticationManager" ref=
"crowdAuthenticationManager"/>
        <beans:property name="groupMembershipManager" ref=
"crowdGroupMembershipManager"/>
        <beans:property name="userManager" ref="crowdUserManager"/>
        <beans:property name="authorityPrefix" value="ROLE_"/>
</beans:bean>
```

2. Add the definition of the RemoteCrowdAuthenticationProvider that delegates Spring Security authentication requests to Crowd:

```
<beans:bean id="crowdAuthenticationProvider" class=
"com.atlassian.crowd.integration.springsecurity.RemoteCrowdAuthenticationProvider">
        <custom-authentication-provider />
        <beans:constructor-arg ref="crowdAuthenticationManager"/>
        <beans:constructor-arg ref="httpAuthenticator"/>
        <beans:constructor-arg ref="crowdUserDetailsService"/>
</beans:bean>
```

3. Comment out the default authentication provider, as we've replaced it with Crowd:

```
<!--
        <authentication-provider user-service-ref="userDao">
                <password-encoder ref="passwordEncoder"/>
        </authentication-provider>
-->
```

4. Now do a:

```
mvn clean install
```

   This will pick up the configuration changes and add the Crowd client library into your app. Then run:

```
mvn jetty:run-war -Dmaven.test.skip
```

5. Head over to `http://localhost:8080/`.
   You should now be able to authenticate the users in your Crowd repository that **meet all of the following conditions**:

- They are in a Crowd directory assigned to the AppFuse application in Crowd. See more information.
- They are in Crowd groups named `USER` and `ADMIN`. You will need to add these groups and assign the user as a member of the groups. These Crowd group names map to the Spring Security authorisation roles defined in the AppFuse application.
- They are allowed to authenticate with the AppFuse application because EITHER they are in a group allowed to authenticate with Crowd (click for details) OR their container directory allows all users to authenticate (click for details).

Congratulations. You have **centralised authentication**

Application-level centralised user management
One quirk you may notice is that you can't view the profile details of users who exist in Crowd, but did not exist in AppFuse prior to the Crowd integration. Although it's possible to authenticate a Crowd user 'dude' and still run AppFuse as 'dude', 'dude' will not be in AppFuse's local database. AppFuse makes use of a database-backed user management system. In order to achieve application-level **centralised user management**, AppFuse will need to delegate its calls to create, retrieve, update and delete users to Crowd via [Crowd's remote API]. This will prevent data redundancy and eliminate the hassle of data synchronisation. This is beyond the scope of this short tutorial.

Step 5. Hook Up Single Sign-On

Enabling single sign-on (SSO) requires quite a bit more tweaking of the `security.xml`:

1. Remove defaults from the `<http/>` element:
   a. Remove the `auto-config` attribute and add an `entry-point-ref="crowdAuthenticationProcessingFilterEntryPoint"` attribute to the http element.
   b. Remove the `<form-login>` element.
   You should end up with an http element similar to this:

```
01.<http lowercase-comparisons="false" entry-point-ref=
"crowdAuthenticationProcessingFilterEntryPoint"> <!-- note: no auto-config attribute!
-->
02.          <!--intercept-url pattern="/images/*" filters="none"/>
03.          <intercept-url pattern="/styles/*" filters="none"/>
04.          <intercept-url pattern="/scripts/*" filters="none"/-->
05.          <intercept-url pattern="/admin/*" access="ROLE_ADMIN"/>
06.          <intercept-url pattern="/passwordHint.html*" access=
"ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
07.          <intercept-url pattern="/signup.html*" access=
"ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
08.          <intercept-url pattern="/a4j.res/*.html*" access=
"ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
09.          <!-- APF-737, OK to remove line below if you're not using JSF -->
10.          <intercept-url pattern="/**/*.html*" access="ROLE_ADMIN,ROLE_USER"/>
11.<!--  <form-login login-page="/login.jsp"
authentication-failure-url="/login.jsp?error=true"
login-processing-url="/j_security_check"/> -->
12.          <remember-me user-service-ref="userDao" key=
"e37f4b31-0c45-11dd-bd0b-0800200c9a66"/>
13.</http>
```

2. Change the default processing filter to Crowd's SSO filter by adding the following bean definitions:

```
01.<authentication-manager alias="authenticationManager"/>
02.
03.<beans:bean id="crowdAuthenticationProcessingFilterEntryPoint" class=
"org.springframework.security.ui.webapp.AuthenticationProcessingFilterEntryPoint">
04.          <beans:property name="loginFormUrl" value="/login.jsp"/>
05.</beans:bean>
06.
07.<beans:bean id="crowdAuthenticationProcessingFilter" class=
"com.atlassian.crowd.integration.springsecurity.CrowdSSOAuthenticationProcessingFilter">
08.          <custom-filter position="AUTHENTICATION_PROCESSING_FILTER"/>
09.          <beans:property name="httpAuthenticator" ref="httpAuthenticator"/>
10.          <beans:property name="authenticationManager" ref="authenticationManager"/>
11.          <beans:property name="authenticationFailureUrl" value="/login.jsp?error=true"/>
12.          <beans:property name="defaultTargetUrl" value="/"/>
13.          <beans:property name="filterProcessesUrl" value="/j_security_check"/>
14.</beans:bean>
```

3. Add the definition of the CrowdLogoutHandler and add in a LogoutFilter that references it:

```
01.<beans:bean id="crowdLogoutHandler" class=
"com.atlassian.crowd.integration.springsecurity.CrowdLogoutHandler">
02.          <beans:property name="httpAuthenticator" ref="httpAuthenticator"/>
03.</beans:bean>
04.
05.<beans:bean id="logoutFilter" class="org.springframework.security.ui.logout.LogoutFilter">
06.          <custom-filter position="LOGOUT_FILTER"/>
07.          <beans:constructor-arg value="/index.jsp"/>
08.          <beans:constructor-arg>
09.                    <beans:list>
10.                              <beans:ref bean="crowdLogoutHandler"/>
11.                              <beans:bean class=
"org.springframework.security.ui.logout.SecurityContextLogoutHandler"/>
12.                    </beans:list>
13.          </beans:constructor-arg>
14.          <beans:property name="filterProcessesUrl" value="/logout.jsp"/>
15.</beans:bean>
```

4. Now repeat:

```
1.mvn jetty:run-war -Dmaven.test.skip=true
```

SSO will only work for users that are able to **authenticate** with both appplications and are **authorised** to use both applications. Try out the following:

- Log in to Crowd – you should be logged in to AppFuse.
- Log out of AppFuse – you should be logged out of Crowd.
- Log in to AppFuse; log out of Crowd; log in to Crowd as another user; refresh AppFuse – you should be logged in as the new user.

Congratulations, you have **SSO**

Integrating Crowd with Subversion

Install the Crowd Apache connector first
To use the Subversion connector, you will need to have the Crowd Apache Connector already installed.

Crowd's Subversion connector allows you to password-protect a Subversion repository and provide fine grained access by group or user.

Prerequisites

- Crowd Apache Connector.

Configuring Crowd Authentication for Subversion

If you are using Apache to manage access to a Subversion repository (instructions), and are using Crowd to manage the Apache authentication, then you can use the same configuration method to delegate Subversion's user authentication to Crowd.

Example:

```
01.<Location /svn>
02.
03.   # Uncomment this to enable the repository
04.   DAV svn
05.
06.   # Set this to the path to your repository
07.   SVNPath /var/lib/svn
08.
09.   AuthName crowd
10.   AuthType Basic
11.
12.   PerlAuthenHandler Apache::CrowdAuth
13.   PerlSetVar CrowdAppName subversion
14.   PerlSetVar CrowdAppPassword svn
15.   PerlSetVar CrowdSOAPURL http://localhost:8095/crowd/services/SecurityServer
16.
17.   require valid-user
18.
19.   # The following three lines allow anonymous read, but make
20.   # committers authenticate themselves.
21.   <LimitExcept GET PROPFIND OPTIONS REPORT>
22.   Require valid-user
23.   </LimitExcept>
24.
25.</Location>
```

Note that you will need to restart Apache before any changes to its configuration files will take effect.

Configuring Crowd Authorisation for Subversion

To restrict Subversion repository access to certain groups and/or users, you can add the `Apache::CrowdAuthz` module and the `CrowdAllowedGroups` and `CrowdAllowedUsers` directives (described here).

For more fine-grained access, Crowd provides the `CrowdAuthzSVNAccessFile` directive which allows you to define path-based access rules. Add the following lines (modifying the file location to fit your configuration) to the example above to enable the `CrowdAuthzSVNAccessFile` directive.

```
PerlAuthzHandler Apache::CrowdAuthz
  PerlSetVar CrowdAuthzSVNAccessFile /etc/apache2/dav_svn.authz
```

The `CrowdAuthzSVNAccessFile` setting lets you define a file where you can configure group and user access at directory level.

The format of the file is the same as that used by Subversion's own authorisation module, `mod_authz_svn`. Here is a small example:

```
# Everyone has read access to the repository
# (unless modified below).
[/]
* = r

# Members of the bazdevelopers group can
# read and write to the BazWord project
[/BazWord]
@bazdevelopers = rw

# Members of the foodevelopers group can read and write
# to the FooCalc project
[/FooCalc]
@foodevelopers = rw

# Members of foodevelopers can read the branches
# directory but only user juliag (the release manager)
# can write to this path
[/FooCalc/branches]
juliag = rw
@foodevelopers = r

# peterc is a contractor, so he's denied all access to the statistics
# module (which is full of trade secrets).
[/FooCalc/trunk/statistics]
peterc =
```

Some notes:

- The format is a series of one or more repository paths (minus the leading URL) followed by one or more group or user directives for each path.
- You don't have to include every single path. If an exact path match is not found, the settings for the nearest parent directory are used.
- Access for the user or group can be set to one of:
  - `rw`: read and write access.
  - `r`: read-only access.
  - `<blank>`: no access.
- Group names are indicated by a leading '@' character.
- Lines starting with a '#' are comments.
- Note that the `[groups]` section of the file described in the Subversion documentation is ignored by `Apache::CrowdAuthz`, because group memberships come from Crowd.
- Do not prefix the paths in the file with the repository name (e.g. '[calc:/foo]'). (See note on `SVNParentPath` below.)
- If you specify a `CrowdAuthzSVNAccessFile` as well as one or both of `CrowdAllowedGroups` and `CrowdAllowedUsers`, only the `CrowdAuthzSVNAccessFile` is used for authorisation.

SVNParentPath Not Supported with CrowdAuthzSVNAccessFile
Subversion provides the SVNParentPath directive, which allows multiple repositories in the same directory to use the same URL. The Crowd Apache integration modules **do not** support the use of `SVNParentPath` if you are using the `CrowdAuthzSVNAccessFile` directive to define permissions. (Using `SVNParentPath` without the `CrowdAuthzSVNAccessFile` directive works fine.)

Mixing Authenticated and Anonymous Access

A common requirement for Subversion access is to have a combination of anonymous (where a username and password is not required) and authenticated access. For example, many administrators want to allow anonymous users to read certain repository directories, but want only authenticated users to read (or write) more sensitive areas. To enable anonymous access, add the following lines to the Apache configuration file:

```
PerlAccessHandler Apache::CrowdAuthz->access_handler
PerlAuthzHandler Apache::CrowdAuthz
PerlSetVar CrowdAuthzSVNAccessFile /etc/apache2/dav_svn.authz
Satisfy Any
```

When anonymous access is enabled as shown above, Apache will not require a password for any part of the repository that matches the '*' user in the `CrowdAuthzSVNAccessFile` file. For example, if you wanted to allow anonymous read access to most of a repository but require authentication for a private section, the `CrowdAuthzSVNAccessFile` file would look like this:

```
# login not required to read, only members of the 'developers' group can check in changes
[/]
* = r
@developers = rw

# anonymous access denied to /private directory
[/private]
* =
@developers = rw
```

See also this example in the Subversion documentation.

For a detailed description of the `AuthzSVNAccessFile` file format, see the Subversion documentation.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Integrating Crowd with a Custom Application

Crowd ships with out-of-the-box support for a number of applications. You can also integrate Crowd with other applications as follows:

Step 1. Configuring Crowd to talk to your Application

Please see Adding an Application.

Step 2. Configuring your Application to talk to Crowd

2.1 Developing a Crowd Client

If your application is not listed in Supported Applications and Directories then you will need to create your own Crowd Client for your application, using the Crowd SOAP API.
For assistance, please see Creating a Crowd Client for your Custom Application.

2.2 Configuring your Application

The integration libraries and configuration files are included in the Crowd download, in the `client` folder. You will find the Crowd integration library, and the client libraries on which the framework depends, in the `lib` folder. An example client properties file `crowd.properties` is located in the `conf` folder.

To configure your application, perform the following:

1. Copy the Crowd Client and supporting libraries to your application classpath, typically `WEB-INF/lib`.
    - These files will be in the `client` folder similar to `crowd-integration-client-X.X.X.jar` and all supporting jars in the `client/lib` folder.
2. Copy the client properties file `crowd.properties` to your application's deployment directory, typically `WEB-INF/classes`.
3. Edit the `crowd.properties` file to reflect the values of your deployment parameters. Refer to the description of the attributes in the `crowd.properties` file.

Passing crowd.properties as an environment variable

You can pass the location of a client application's `crowd.properties` file to the client application as an environment variable when starting the client application. This means that you can choose a suitable location for the `crowd.properties` file, instead of putting it in the client application's `WEB-INF/classes` directory.

This applies to the Crowd Administration Console's `crowd.properties` file too. You may find this particularly useful when integrating with a WAR deployment of an integrated application.

Example:

```
-Dcrowd.properties={FILE-PATH}/crowd.properties
```

RELATED TOPICS

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

[Crowd Documentation

Configuring the Google Apps Connector

The Google Apps connector is shipped with your Crowd installation. This is a Crowd application connector which allows single sign-on (SSO) to Google Apps. If you wish to activate SSO between Crowd-connected applications and Google Apps, you will need to configure the Google Apps connector as described below.

On this page:

- Prerequisites
- Step 1. Configuring the Crowd Application, Directory and Group Details

- Step 2. Generating your SSO Keys
- Step 3. Configuring Google Apps to Recognise Crowd
- Step 4. Verifying that a User can Log in to Google Apps
- More Information about the Google Apps Connector
  - Deleting the Keys
  - The Ins and Outs of SSO with Google Apps
  - Usernames must be the Same in Google Apps and Crowd
  - Other Authentication Frameworks and SAML Support
- An Example of Google Apps SSO in Action

Prerequisites

Please note the following before you start:

- **Google Apps support for SSO:** To enable single sign-on in Google Apps, you will need the Premier, Education, or Partners edition of Google Apps. The free Standard Edition of Google Apps does not support SSO. See the Google Apps documentation.
- **Using the Google Apps Connector with Java 6:** If you want to integrate Crowd with Google Apps in a **JDK 1.6** environment, you will need to download two extra files. Please refer to CWD-1388.

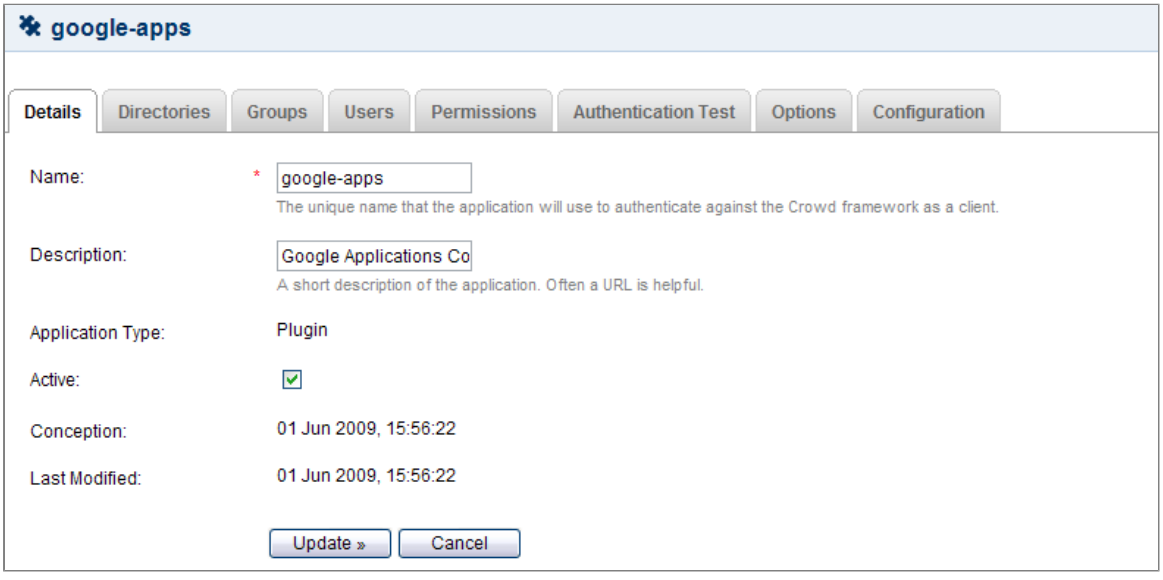
Step 1. Configuring the Crowd Application, Directory and Group Details

In this step, you will enter the application details for the Google Apps application connector in Crowd. You will manage access to Google Apps by associating Crowd directories and/or groups with the Google Apps application.

To define the Google Apps application details in Crowd,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications** tab in the top navigation bar.
3. The Application Browser will appear. Click the link on the '**google-apps**' application name.
4. The application '**Details**' screen will appear, as shown below. If you wish, you can change the '**Description**'. Please ensure that the '**Active**' checkbox remains ticked.
5. Click the '**Directories**' tab and select one or more user directories which contain the users who should have access to Google Apps.
6. To choose which users within the directory may authenticate against the application, either:
   - On the '**Directories**' tab, change '**Allow all to authenticate**' to '**True**'. This will allow all users in that directory to log in to Google Apps. (The default is '**False**'.)
     OR
   - Click the '**Groups**' tab and select one or more groups of users, clicking the '**Add**' button to add each group you need.
7. Click the '**Permissions**' tab and set the directory permissions for the application.
8. If you wish, you can change the application options on the '**Options**' tab:
   - **Lower Case Output** — See Enforcing Lower-Case Usernames, Groups and Roles for an Application.
   - **Enable Aliasing** — See Specifying a User's Aliases.
9. Click the '**Configuration**' tab and generate your SSO keys as described in Step 2 below.

Screenshot: Google Apps application details in Crowd



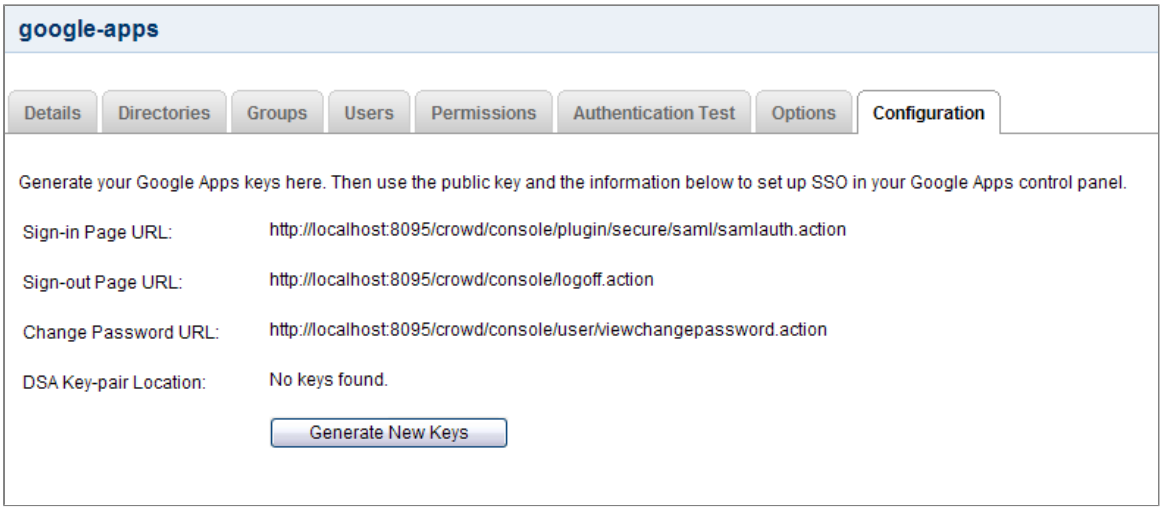
Step 2. Generating your SSO Keys

Now you will ask Crowd to generate a public and a private key for use in authenticating Crowd to Google Apps. (Google Apps calls the public key a 'verification certificate'.)

To generate your SSO keys,

1. Still in the Crowd Application Browser as described in Step 1 above, click the '**Configuration**' tab for the Google Apps application.
2. The 'Configuration' screen will appear, as shown below. Click the '**Generate New Keys**' button.
3. Crowd will generate a public key and a private key, placing them in the `plugin-data\crowd-saml-plugin` directory of your Crowd Home. (For more information about Crowd Home, see Important Directories and Files.) When the keys have been generated, you will see a message '*DSA keys successfully generated and stored to disk.*'

Screenshot: Configuring the Google Apps connector in Crowd



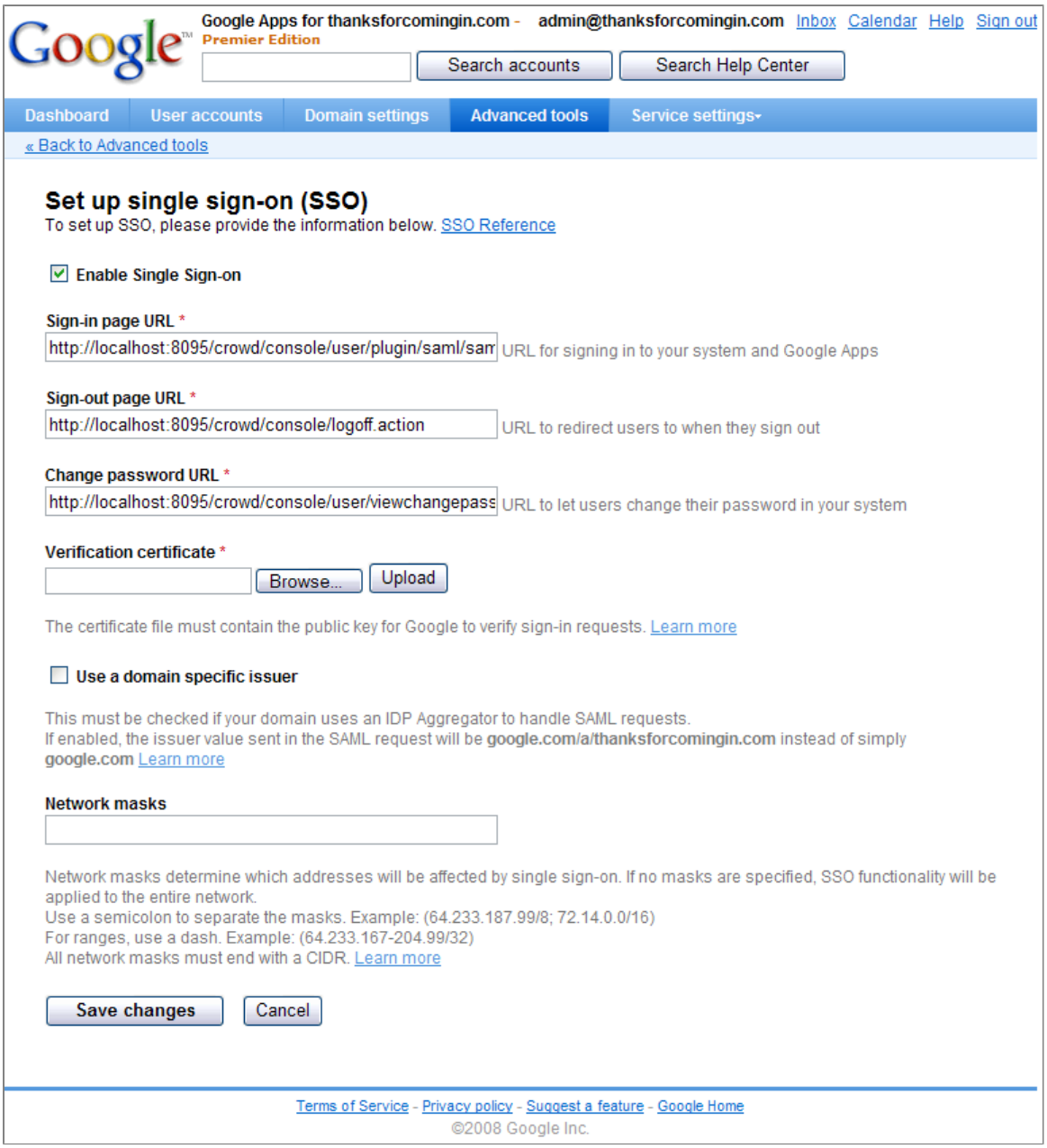
Step 3. Configuring Google Apps to Recognise Crowd

In this step, you will log in to Google Apps as an administrator and enter the information required for Crowd to authenticate to Google Apps. This information consists of some Crowd URLs and the public key which you generated from Crowd.

To configure Google Apps to recognise Crowd,

1. Log in to your **Google Apps Dashboard** as a **Google Apps administrator**.
2. In Google Apps, go to the '**Advanced tools**' tab.
3. Click the '**Set up single sign-on (SSO)**' link.
4. The 'Set up single sign-on (SSO)' screen will appear, as shown below.
5. Copy the URLs from the Crowd configuration screen (see above) and paste them into the Google Apps screen.
6. Now you will upload the public key which Crowd generated for you in Step 2 above:
   - Still in Google Apps, click the '**Browse**' button under the heading '**Verification certificate**'.
   - Navigate to the the `plugin-data\crowd-saml-plugin` directory of your Crowd Home.
   - Select the public key certificate (file name `DSAPublic.key`) and upload it to Google Apps.
7. If necessary for your network configuration, set the 'Use a domain specific issuer' checkbox and the 'Network masks' in Google Apps. Please refer to the Google Apps documentation for guidance on these settings.
8. Save your changes in Google Apps.

Screenshot: Setting up SSO in Google Apps



Step 4. Verifying that a User can Log in to Google Apps

It is a good idea now to check your users can log in to Google Apps.

To test a user's authentication to Google Apps,

1. Still in the Crowd Application Browser as described in Step 2 above, click the '**Authentication Test**' tab for the Google Apps application.
2. Enter a user's login details and verify the login. For more details, you can refer to Testing a User's Login to an Application.

Congratulations! You have now configured Crowd for SSO with Google Apps.

More Information about the Google Apps Connector

Deleting the Keys

Once you have generated the keys, a '**Delete Keys**' button will appear on Crowd's configuration screen. Click this button to remove the keys from the Crowd Home directory. This will disable SSO with Google Apps.

The Ins and Outs of SSO with Google Apps

- Single sign-on (SSO) applies only to the applications within Google Apps. The Google Apps administration section (control panel) does not support SSO.

- When you sign out of Google Apps, you will also be signed out of Crowd and all Crowd-connected applications. This is the usual SSO behaviour.
- But when you sign out of Crowd, you will remain logged in to Google Apps even though you will be logged out of other Crowd-connected applications. (Reason: Google does not rely on a cookie, so there is no easy way for Crowd to tell Google you have signed out.)

  It would take some additional development to support single sign-out from Google Apps. If you would like to see this work undertaken, please vote for issue CWD-1238.
- If you go directly to a Google Apps application without logging in to Crowd, Google Apps direct you to a Crowd login screen.
- The Crowd login screen for Google Apps will not offer a 'Forgotten your password' link. You cannot change your Crowd password via Google Apps. Instead, if you need to change your password please log in to Crowd directly, by going to this URL: http://YOUR-CROWD-LOCATION:8095/crowd/

Usernames must be the Same in Google Apps and Crowd

Usernames must exist in Google Apps as well as Crowd and a person's username must be the same in both Google Apps and Crowd. The Crowd Google Apps connector does not support the automatic adding of users. If a user exists in Crowd but not in Google Apps, then the user will not be able to log in to Google Apps.

Other Authentication Frameworks and SAML Support

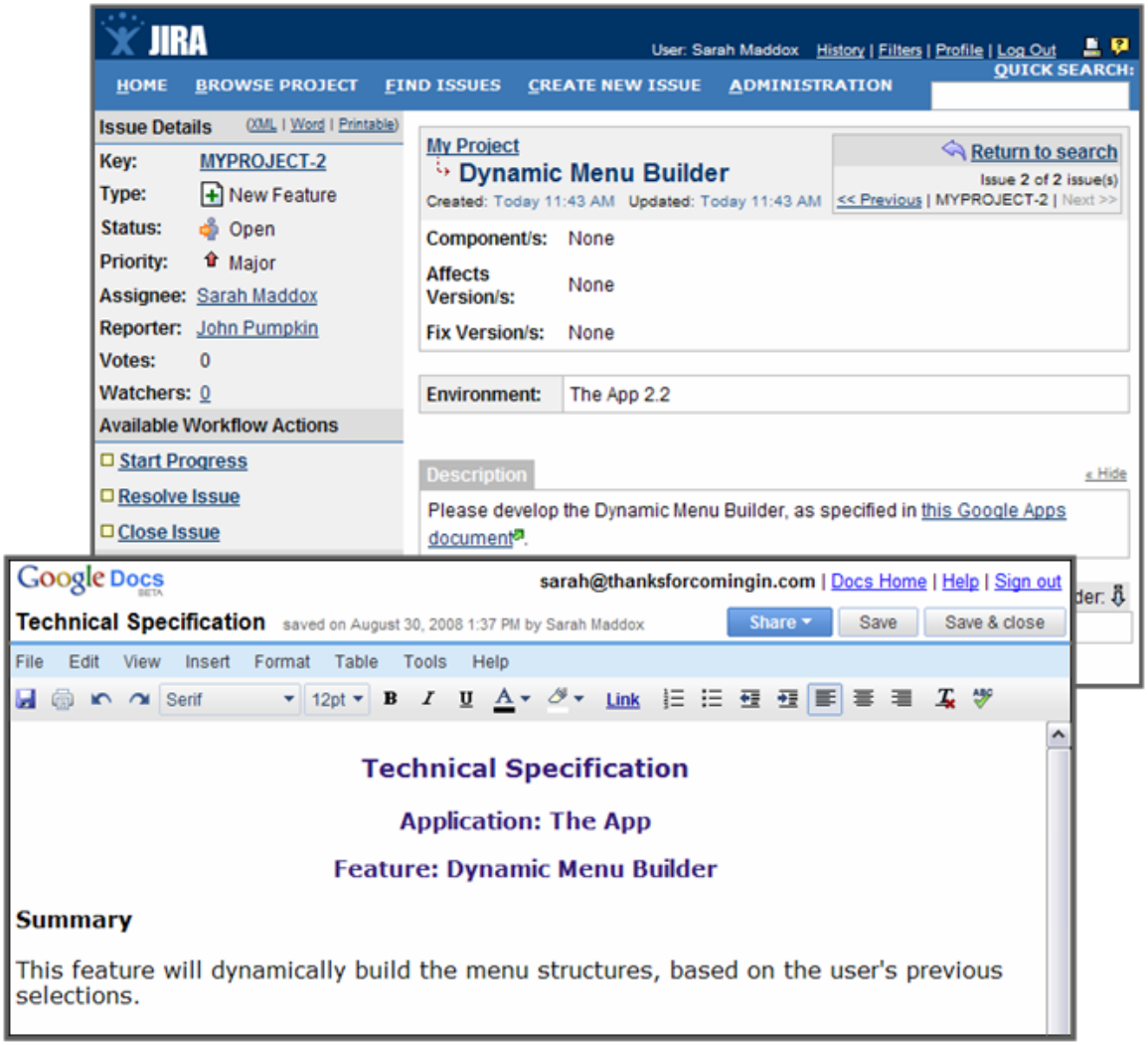
Crowd currently supports SSO via SAML with Google Apps only. The following information is relevant to developers who may want to use Crowd's classes to develop a plugin that supports SAML authentication with other frameworks.

Crowd's SAML implementation meets the requirements for Google Apps SSO. As Google Apps supports a subset of the SAML 2.0 spec, any authentication framework that relies on the same subset should also be compatible. The Crowd implementation is capable of servicing SAML 2.0 authentication requests using the HTTP-Redirect binding. For more information on the Google Apps authentication protocol, check out their SSO documentation.

An Example of Google Apps SSO in Action

Here's one example of how it might work:

- John raises an issue in JIRA. In the issue description, he adds a link to a Google Apps document containing more details.
- He assigns the issue to Sarah.
- Sarah clicks the link and opens the document directly in Google Apps. No need to log in again, no need to remember a different password.



RELATED TOPICS

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Mapping a Directory to an Application

Mapping a directory to an application defines the user-base for an application. Sometimes known as 'application provisioning', directory mappings determine which user stores will be used when authenticating and authorising a user's access request. Read more about users, groups and roles.

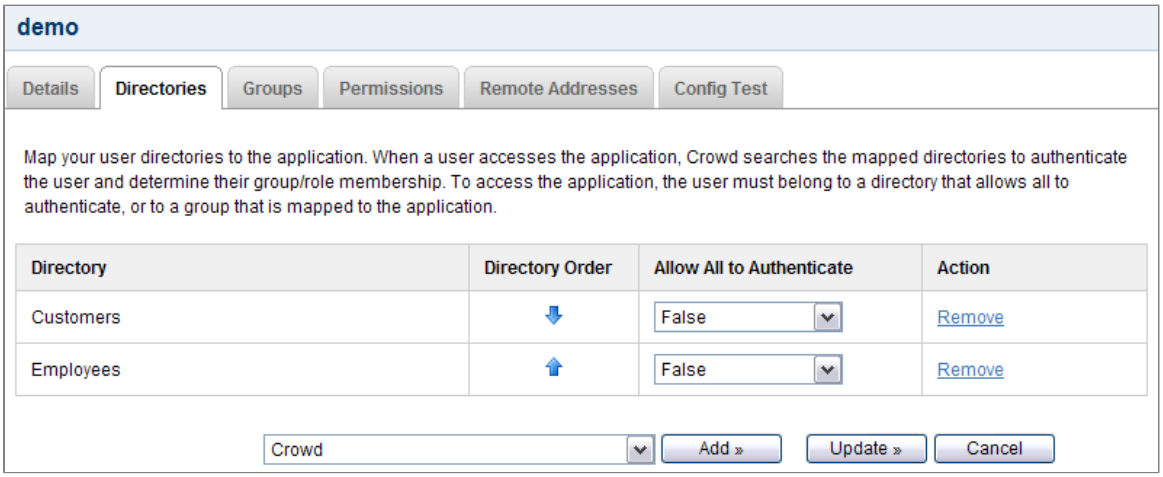
When you defined an application, you chose a default directory for that application to use. Crowd also allows you to map multiple directories to each application. This allows each of your applications to view multiple user directories as a single repository.

To map a directory to an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications** tab in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link that corresponds to the application you wish to map.
4. This will display the '**View Application**' screen. Click the '**Directories**' tab.
5. This will display a list of directories that are currently mapped to the application. Select the new directory from the drop-down list and click the '**Add**' button.
6. The new directory will be added to the bottom of the list of mapped directories. You can use the blue up-arrow or down-arrow to move a directory higher or lower in the order:

   Why directory order is important
7. You now need to choose which users within the directory may authenticate against the application. You have two choices:
   - To allow *all users* within the directory to authenticate against the application, change '**Allow all to Authenticate**' to '**True**', then click the '**Update**' button.
     OR:
   - To allow *only specific groups of users* within the directory to authenticate against the application, see Specifying which Groups can access an Application.
8. Next, you should define the application's ability to add/update users in the directory. Click the '**Permissions**' tab and set the directory permissions for the application.

Screenshot: 'Application — Map Directories'



RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application

- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Specifying the Directory Order for an Application

When you map multiple directories to an application, you also need to define the directory order.

The directory order is important during the authentication of the user, in cases where the same user exists in multiple directories. When a user attempts to log in to an application, Crowd will search the directories in the order you specified, and will use the credentials (password, etc) of the *first occurrence of the user* to validate the login attempt. See diagram below.

The directory order is *not* significant when granting the user access to an application based on group membership. In the case of multiple directories, Crowd amalgamates the group memberships in the directories. See below.

On this page:

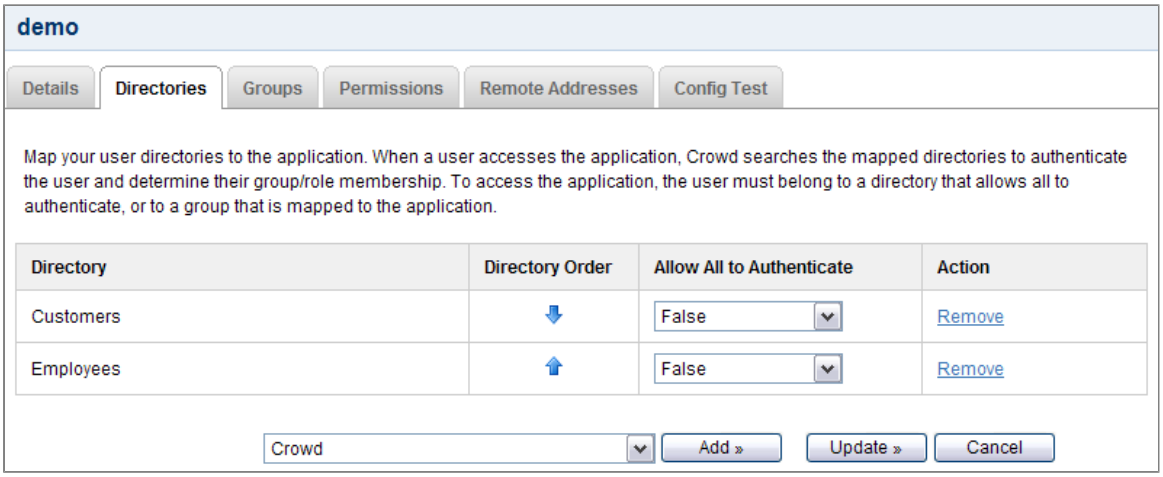
- Specifying the Directory Order
- How Authentication Works
- How Authorisation via Group Membership Works

Specifying the Directory Order

To specify the directory order,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link that corresponds to the application you wish to map.
4. This will display the '**View Application**' screen. Click the '**Directories**' tab.
5. This will display a list of directories that are currently mapped to the application. Use the blue up-arrow or down-arrow to move a directory higher or lower in the order:

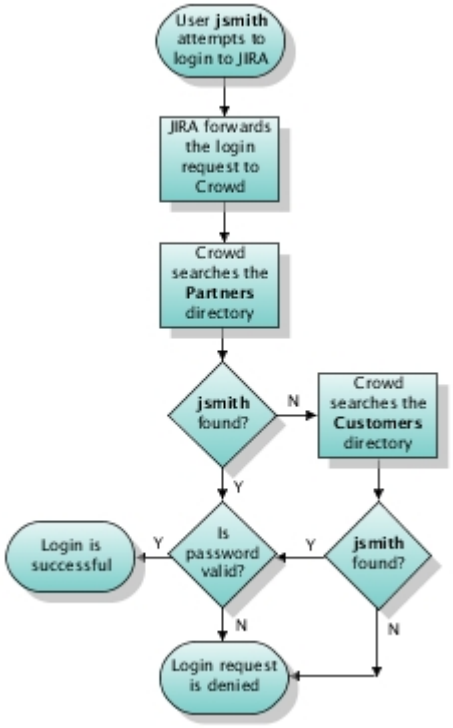
Screenshot: 'Application---Mapped Directories'



How Authentication Works

The directory order is important during the authentication of the user.

Let's assume that JIRA has been set up as a Crowd application, and has been mapped to two directories, '**Partners**' and '**Customers**', in that order.
Here is what happens when a user attempts to log in to JIRA:



How Authorisation via Group Membership Works

The directory order is *not* significant when granting the user access to an application based on group membership.

If the same username exists in more than one directory assigned to an application, Crowd treats these usernames as the same user. Crowd searches all the assigned directories for the user, and amalgamates the group and role memberships.

For example, let's assume you have a user 'jsmith' who exists in both directories 'Customers' and 'Partners', and is a member of group 'G1' in 'Customers' and 'G2' in 'Partners'. Crowd will grant access to the user based on membership of both 'G1' and 'G2'.
RELATED TOPICS

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Specifying an Application's Directory Permissions

When you map a directory to an application, you can also define the application's ability to add/update/delete users, groups and roles in the directory. To do this, use the '**Permissions**' tab in the 'View Application' screen.

Directory permissions are defined at two levels:

1. **Directory-level permissions** are defined on the 'Permissions' tab of the 'View Directory' screen. These permissions apply to each application mapped to the directory, unless the application has its own application-level permissions.
2. **Application-level directory permissions** are defined on the 'Permissions' tab of the 'View Application' screen. If a permission is enabled at directory level, you can enable it for a specific application. For example, you could enable the 'Add User' permission on the 'Customers' directory in JIRA but disable the permission for Confluence.

Take a look at an example.

Disabling a directory-level permission will override any permissions enabled at application level. If a permission is enabled at application level and then subsequently disabled at directory level, the directory-level permission will apply. (The application-level permissions will be 'remembered' and will apply again if re-enabled at directory level.)

How do directory permissions affect the Crowd application (Crowd Administration Console)?

- If a particular permission is turned off at directory level, then **no** application can perform the related function - not even the Crowd application. So, for example, if you disable the 'Remove User' permission for a directory, then the Crowd Administration Console will not allow you to delete a user from that directory.
- The Crowd application is not bound by application-level permissions.

For details on directory-level permissions, refer to the instructions on specifying directory permissions. Below are instructions on setting the application-level directory permissions.

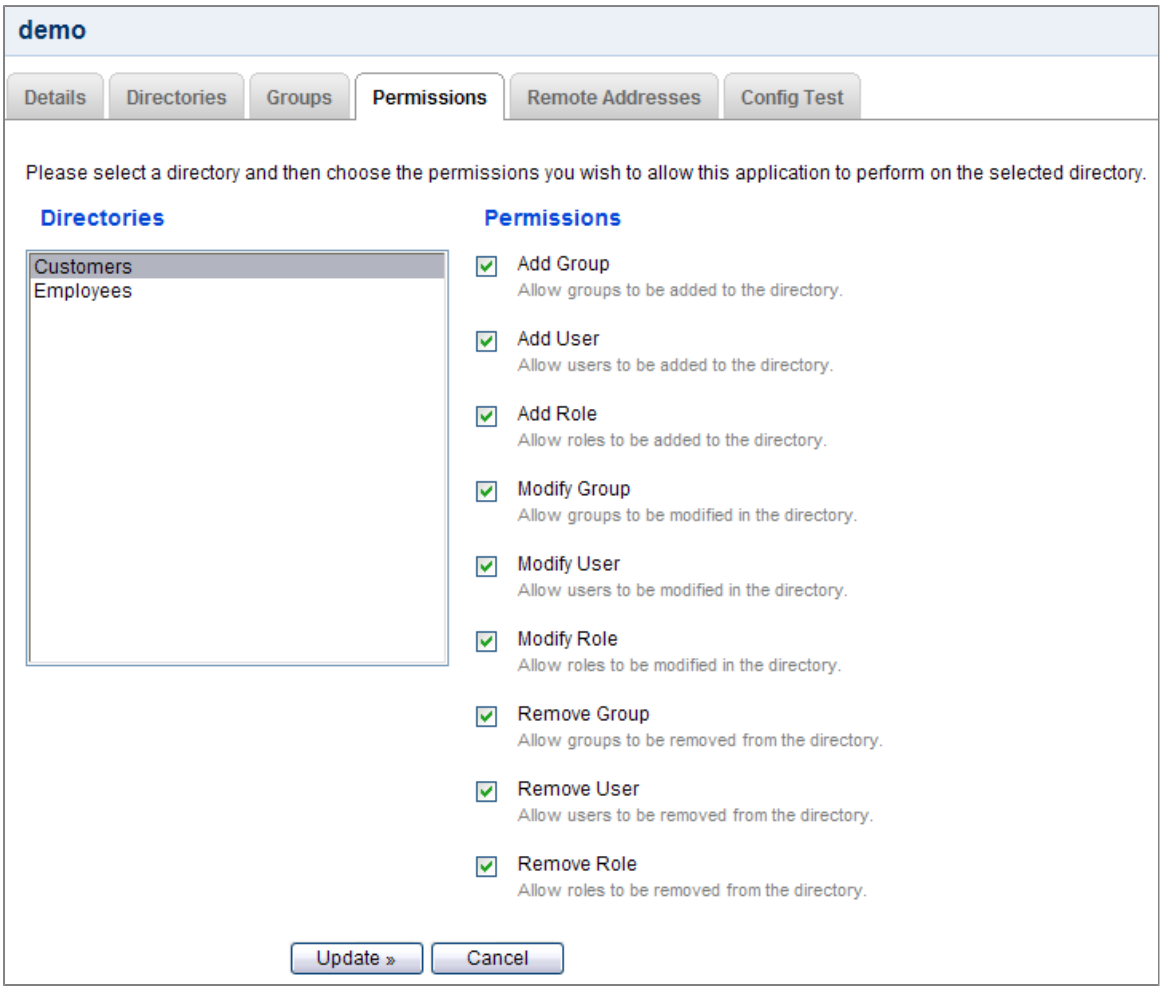
Permission	Description
Add Group	Allows the application to add groups to the selected directory.
Add User	Allows the application to add users to the selected directory.
Add Role	Allows the application to add roles to the selected directory.
Modify Group	Allows the application to modify groups in the selected directory.
Modify User	Allows the application to modify users in the selected directory.
Modify Role	Allows the application to modify roles in the selected directory.
Remove Group	Allows the application to delete groups from the selected directory.
Remove User	Allows the application to delete users from the selected directory. Consider carefully whether you allow the deletion of users, as some applications contain historical data, e.g. documents that the user has created. Read more.
Remove Role	Allows the application to delete roles from the selected directory.

When you initially map a directory to an application, all of the application's permissions are enabled by default. But note that disabling a directory-level permission will override any permissions enabled at application level.

To set the directory permissions for an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link next to the application you wish to update.
4. This will display the '**View Application**' screen. Click the '**Permissions**' tab.
5. This will display a list of directories that are currently mapped to the application, and a set of permission check-boxes. Select a **directory** from the list on the left.
6. The '**Permissions**' check-boxes will change to show the application's existing permissions for that directory.
   - **To enable a directory permission**, select the corresponding check-box.
   - **To disable a directory permission**, deselect the corresponding check-box.

Screenshot: Setting directory permissions for an application



On the application permissions screen, the words '**(disabled globally)**' will appear next to any permission that is disabled at directory level.

RELATED TOPICS

- Specifying Directory Permissions

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application

- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

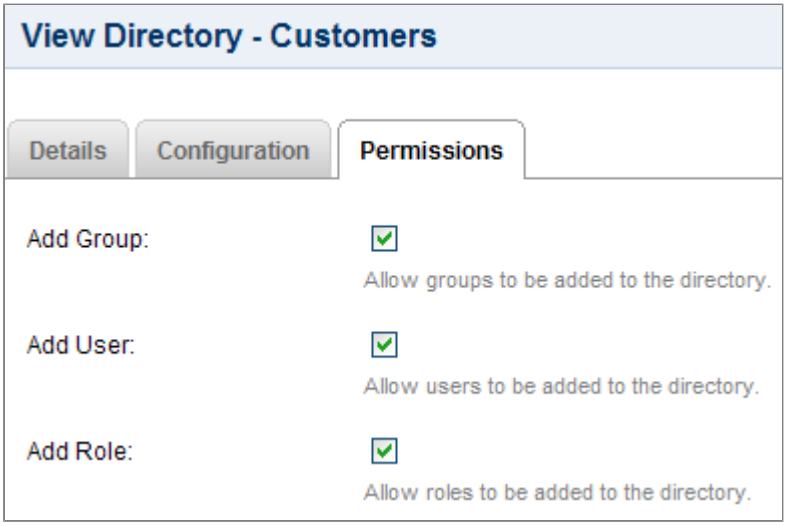
Example of Directory Permissions

Let's assume that you want to:

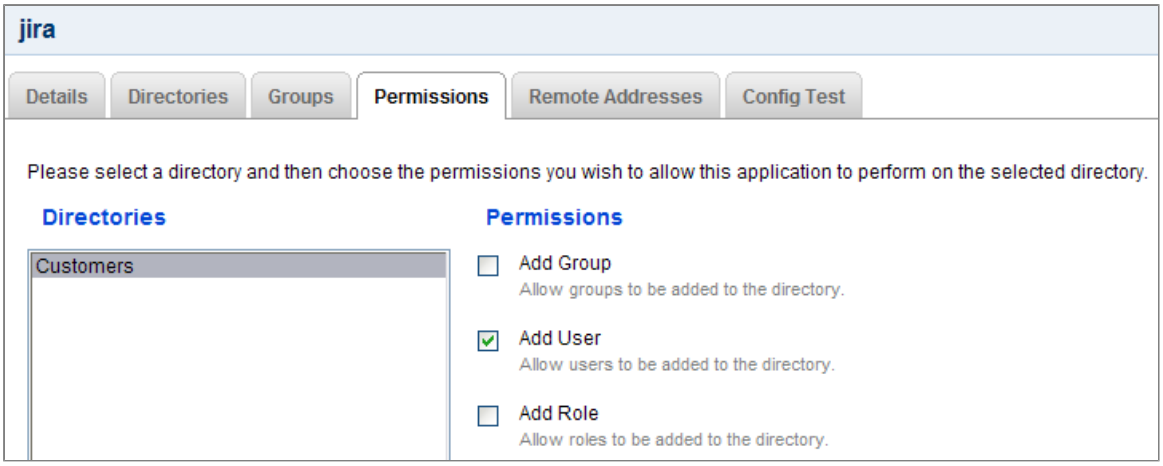
- Allow self-registration (automatic signup) of new users in your 'Customers' directory via JIRA, and
- Disable self-registration via Confluence.

Here's how you would set the directory-level and application-level permissions in Crowd.

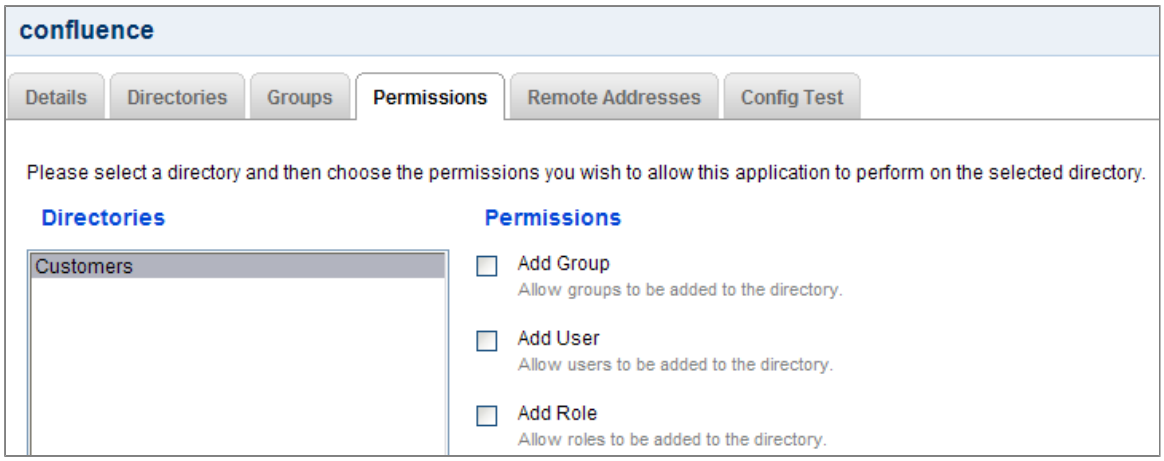
1. At directory level, enable the 'Add User' permission (and any other permissions you want):
    a. In the Crowd Administration Console, click the '**Directories**' tab in the top navigation bar.
    b. Select the 'Customers' directory.
    c. Click the '**Permissions**' tab.
    d. Select the '**Add User**' check-box.



2. At application level, make sure the 'Add User' permission is enabled for the JIRA application:
    a. Click the '**Applications**' tab in the top navigation bar.
    b. Click the '**View**' link next to the JIRA application.
    c. In the '**View Application**' screen, click the '**Permissions**' tab.
    d. Select the 'Customers' directory.
    e. Select the '**Add User**' check-box.



3. At application level, disable the 'Add User' permission the Confluence application:
    a. Click the '**Applications**' tab in the top navigation bar.
    b. Click the '**View**' link next to the Confluence application.
    c. Click the '**Permissions**' tab.
    d. Select the 'Customers' directory.
    e. Deselect the '**Add User**' check-box.



In summary:
With the above application permissions, a person will be able to sign up for a user account via JIRA and this user will be created in the 'Customers' directory, but they will not be able to sign up for an account via Confluence.

RELATED TOPICS

- Specifying Directory Permissions
- Specifying an Application's Directory Permissions

Crowd Documentation

Specifying which Groups can access an Application

You can specify which users are allowed to authenticate against each application. For each mapped directory, you can either allow *all* users within the directory to authenticate with the application, or just particular groups within the directory. You can then assign group membership to each user.

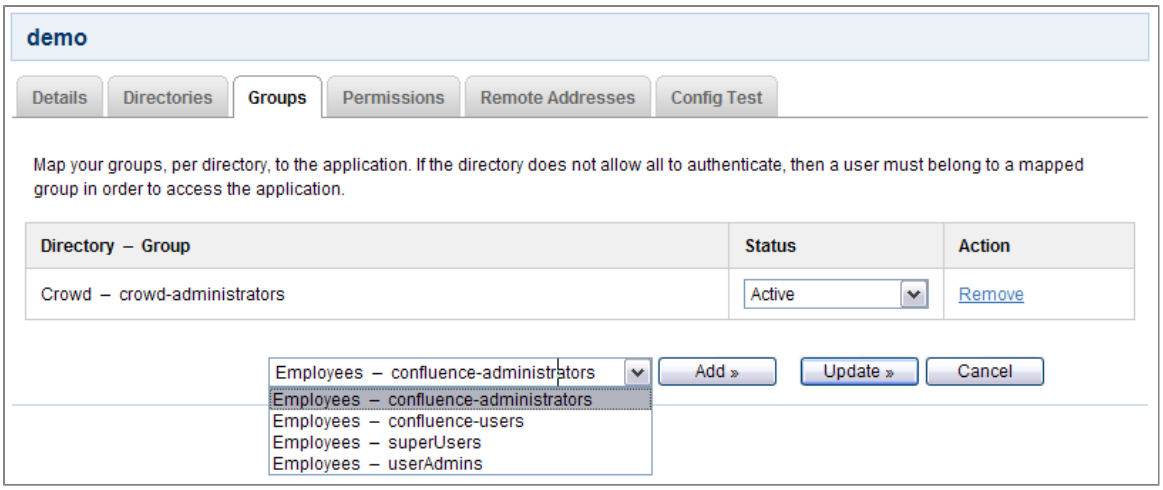
For example, the default group `crowd-administrators`, which is automatically created in the default directory that you specified during setup, is allowed to access the Crowd Administration Console. This means that users who belong to the group `crowd-administrators` are allowed to log in to the Crowd Administration Console (assuming they supply a valid password).

To allow a group to access an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link that corresponds to the application you wish to map.
4. This will display the '**View Application**' screen. Click the '**Groups**' tab.
5. This will display a list of groups that currently have access to the application. Click the drop-down arrow next to the '**Add**' button.
6. This will display a list of all the groups that exist within each directory. Select the new group from the drop-down list and click the '**Add**' button.

Alternatively, you can allow *all* users from a particular directory to authenticate against the application. See Mapping a Directory to an Application.

Screenshot: 'Application — Specify Groups'



RELATED TOPICS

- Managing Users, Groups and Roles

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Viewing Users in Directories Mapped to an Application

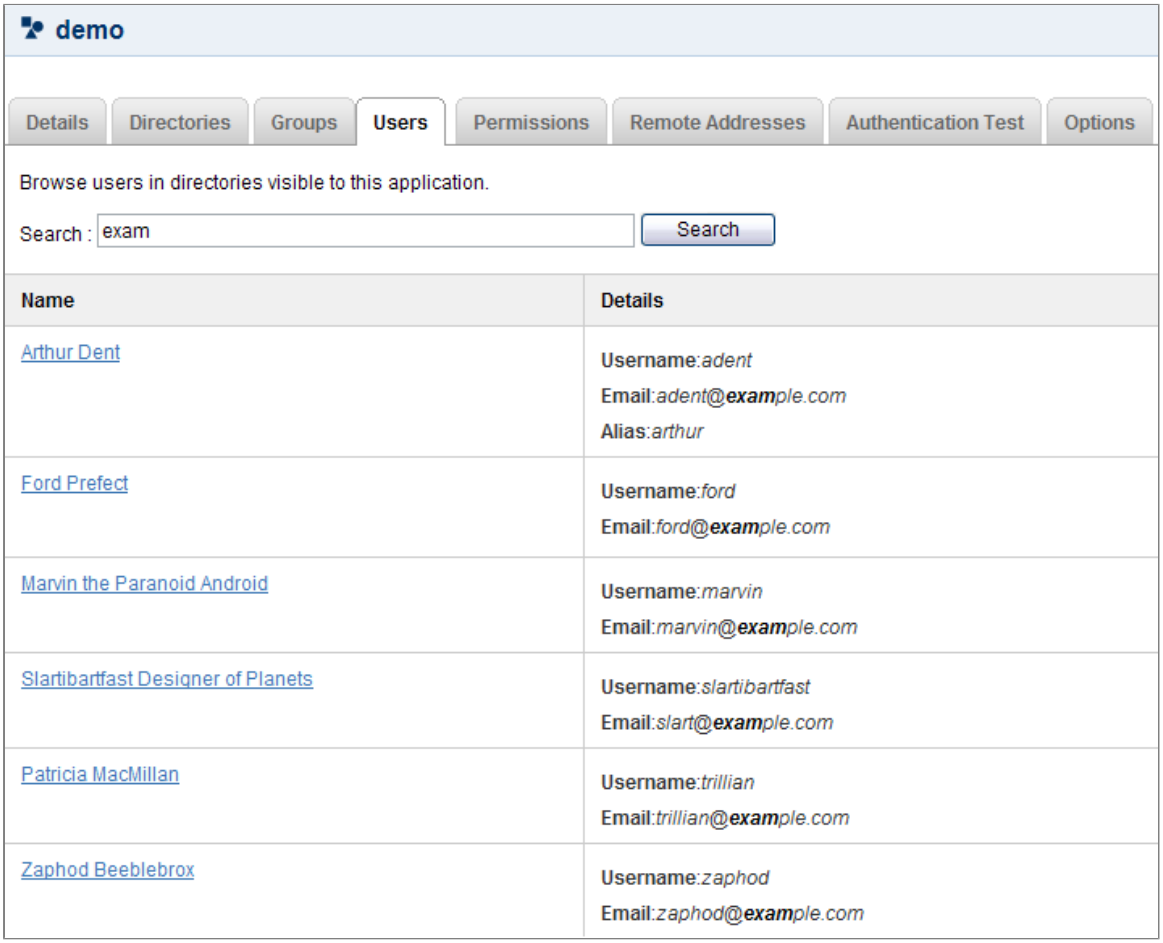
The application '**Users**' tab shows all the users in all the directories mapped to the selected application. You will also see basic information for each user, including the user's full name, username and email address. If the user has an alias for the selected application, the alias will appear too.

> **Group authorisation is not taken into account**
> Note the application 'Users' tab displays all users in the directory/directories mapped to the application, even if the application only allows specific groups within the directory/directories. There is an open feature request to limit the user search to only the users allowed to authenticate with the application: CWD-1348.

To see the users visible to an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. This will display the Application Browser. Click the link on the name of the application you wish to view.
4. The '**View Application**' screen will appear. Click the '**Users**' tab.
5. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the user's name, email address or username. Leave the search box empty to match all users.
6. Click the '**Search**' button.

Screenshot: Viewing users for an application



RELATED TOPICS

Specifying a User's Aliases
Managing Applications
Crowd Documentation

Specifying an Application's Address or Hostname

To ensure that your Crowd server can be used by legitimate applications only, Crowd will allow applications to log in only from known addresses. This means that you need to specify the IP address(es) and/or hostname(s) of each application.

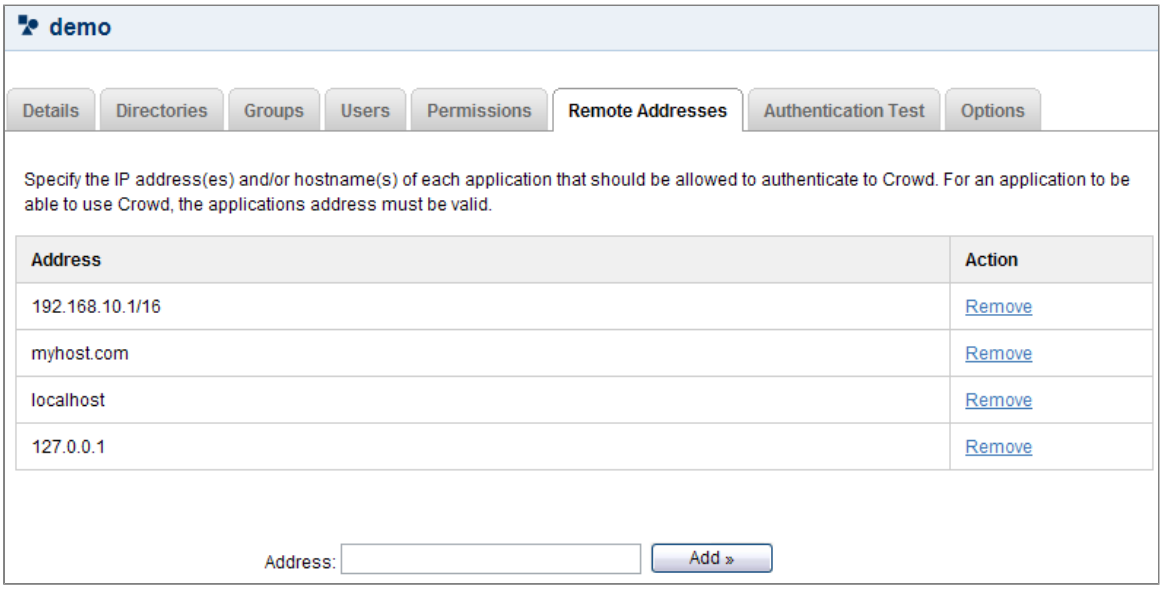
When you add a new application, you will specify the application's IP address. After adding the application, you can update the IP address if necessary, as described below. In some cases, you may need to add the applicable host name as well as the IP address.

IP address and/or host name?
You should always specify the application's IP address. In addition, you may need to give a host name as well as the IP address. Some application servers may pass the host name to Crowd, instead of the IP address. If this happens, Crowd will not grant the application's authorisation request unless Crowd recognises the host name.

To specify an application's IP address or hostname,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. The Application Browser will appear. Click the link on the name of the application you wish to update.
4. The '**View Application**' screen will appear. Click the '**Remote Addresses**' tab.
5. You will see a list of IP addresses and hostnames that are currently mapped to the application. Type the new IP address or hostname into the '**Address**' field and click the '**Add**' button. Possible values are:
    - A full IP address, e.g. `192.168.10.12`.
    - A wildcard IP range, using CIDR notation, e.g. `192.168.10.1/16`. For more information, see the introduction to CIDR notation on Wikipedia and RFC 4632.
    - A host name, e.g. `myhost.com`.
6. The new address will be added to the bottom of the list.

Screenshot: Application addresses



A common problem: Application not connecting with Crowd
For an application to be able to use Crowd, the application's address must be valid and active. Ensure the 'Status' field is set to '**True**'.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
    - Integrating Crowd with Atlassian Bamboo
    - Integrating Crowd with Atlassian Confluence
        - Configuring Confluence for NTLM SSO
    - Integrating Crowd with Atlassian CrowdID
    - Integrating Crowd with Atlassian Crucible
    - Integrating Crowd with Atlassian FishEye
        - Configuring FishEye 1.3.x to talk to Crowd
    - Integrating Crowd with Atlassian JIRA
        - Configuring JIRA for NTLM SSO
    - Integrating Crowd with Acegi Security
        - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    - Integrating Crowd with Apache
    - Integrating Crowd with Jive Forums
        - Jive SSO
    - Integrating Crowd with Spring Security
        - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
    - Integrating Crowd with Subversion
    - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
    - Specifying the Directory Order for an Application
    - Specifying an Application's Directory Permissions
        - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application

- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Testing a User's Login to an Application

You can use an application's '**Authentication Test**' tab to verify that a user will be able to log in to a given application, based on the user, directory and group associations in Crowd.

Performing the Test

The test works like this:

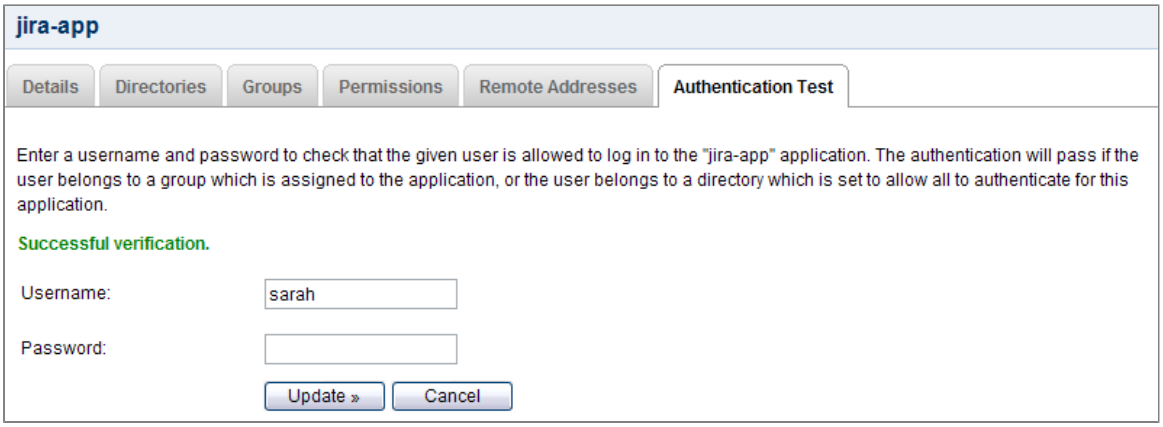
1. You enter the username and password of the user you wish to verify has access to a given application.
2. Crowd searches for the user with that username in the application's mapped directories, and verifies the password.
3. If the user is not found or the password is invalid, the authentication fails the test.
4. Crowd checks whether the directory is set to allow all to authenticate.
5. If all can authenticate, the test passes.
6. Else, Crowd checks the group(s) to which the user belongs and verifies whether those groups have access to the application.
7. If the user belongs to an allowed group, the test passes, otherwise it fails.

To test a user's login to an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications** link in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link that corresponds to the application you wish to verify.
4. This will display the '**View Application**' screen. Click the '**Authentication Test**' tab.
5. Enter the '**Username**' and '**Password**' that you wish to verify.
6. Click the '**Update**' button.
7. A message appears above the 'Username', displaying one of the following:
   - '**Successful verification**' – The authentication has passed the test.
   - '**Invalid verification**' – The authentication has failed the test.

Below are some suggestions for the next steps you can take in each case.

Screenshot: Authentication test showing successful verification



Successful Verification

If this test is successful, but the user is having trouble authenticating to an application, then the problem is caused by the connection between the application and Crowd rather than by user authentication.

Next step: Check the '**Application Sessions**' tab in the Session Browser to see if the application is connected to Crowd.

Failed Verification

If the test declares the login to be invalid, this means that the configuration is incorrect within Crowd.

Next steps:

Check the following - all must be true to allow successful verification.

- The user must belong to a directory which is mapped to this application.
- The password you used must be valid. In particular, check that the password is the one specified in the **first** directory in which the user appears. (If the user belongs to more than one directory, Crowd uses the first directory in which the user appears, as determined by the directory order.)
- Either:
  - The directory must be set to allow all to authenticate.
    OR:
  - The user must belong to a group which has access to the application.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Enforcing Lower-Case Usernames, Groups and Roles for an Application

In some cases you may wish to convert usernames, group names and role names to lower case when passing them to an application. You can set an option for each application, as described below. When the option is set, Crowd will convert upper-case and mixed-case information obtained from your user directory to lower case before passing the information to the application. The conversion is applied to the following information:

- Usernames
- Group names
- Role names
- Group and role memberships

If you set this option for an application, the conversion will apply to **all** directories mapped to the application.

This option is useful in the following situations:

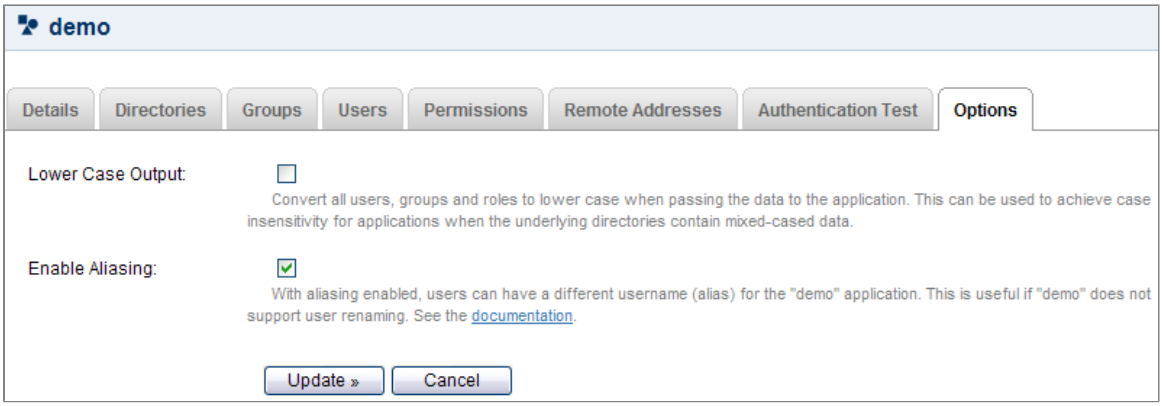
1. First situation: Existing application-to-directory integration:
   - You have previously integrated an application that enforces lower-case usernames (e.g. `jsmith`) with a corporate directory which allows mixed-case usernames (e.g. `JSmith`). Examples of such applications are JIRA and Confluence.
   - You have existing usernames in the application, which are therefore all lower case.
   - Now you want to integrate the application with Crowd.
2. Second situation: You have a custom application which demands lower-case usernames and cannot do the conversion itself.

Check your options carefully
You should only enforce lower-case conversion if you are in a situation as described above. There is no need to enforce lower-case conversion if you are starting out afresh with a Crowd-to-JIRA or Crowd-to-Confluence integration. When lower-case conversion is not enforced, Crowd's behaviour is **case-insensitive but case-preserving** — it will ignore case when comparing usernames etc ('JSmith' = 'jsmith') and it will preserve case when passing information between applications and directories ('JSmith' remains 'JSmith'). This results in the expected behaviour in the Crowd-integrated directories as well as the Crowd-integrated applications such as JIRA and Confluence.

To enforce lower-case conversion for an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. The Application Browser will appear. Click the link on the name of the application you wish to configure.
4. The '**View Application**' screen will appear. Click the '**Options**' tab.
5. Put a tick in the checkbox labelled '**Lower Case Output**'.
6. Click the '**Update**' button.

Screenshot: Application Options



RELATED TOPICS

- Case Sensitivity of Usernames, Groups and Roles
  - Using the Application Browser
  - Adding an Application
  - Configuring the Google Apps Connector
  - Mapping a Directory to an Application
  - Specifying which Groups can access an Application
  - Viewing Users in Directories Mapped to an Application
  - Specifying an Application's Address or Hostname
  - Testing a User's Login to an Application
  - Enforcing Lower-Case Usernames, Groups and Roles for an Application
  - Managing an Application's Session
  - Deleting or Deactivating an Application
  - Configuring Caching for an Application
  - Overview of SSO
  - Configuring Options for an Application

Crowd Documentation

Managing an Application's Session

Crowd allows you to see a list of all applications currently logged in to the Crowd framework. This is effectively a list of the applications which currently have users logged in to them, since an application will only ever log in to the Crowd framework when it needs to authenticate a user.

You can also force any session to expire, that is, you can log the application out of Crowd.

To see which applications are currently logged in to Crowd,

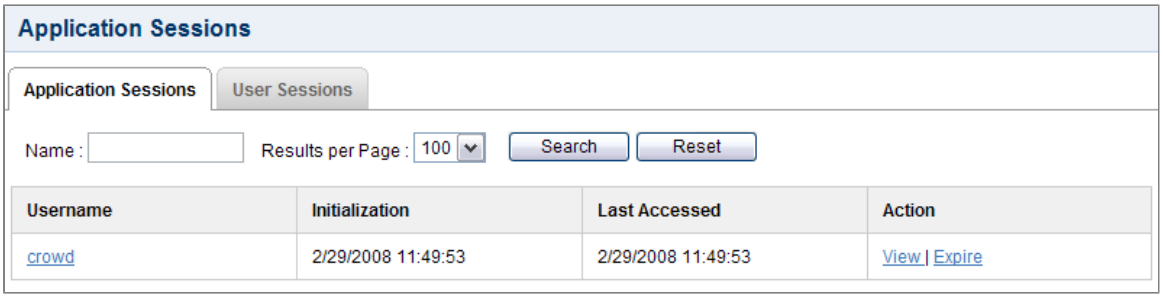
1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Current Sessions**' in the left-hand menu.
4. This will display the '**Application Sessions**' screen, showing a list of all applications which are currently logged in to the Crowd framework. For example, the screenshot below shows that the **crowd** application (i.e. the Crowd Administration Console) is currently logged in to the Crowd framework.
   You can refine your search by specifying an application's '**Name**'. (Note that this is case sensitive.)

To force an application to log out of Crowd,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Current Sessions**' in the left-hand menu.
4. This will display the '**Application Sessions**' screen, showing a list of all applications which are currently logged in to the Crowd framework. Click the application's '**Expire**' link.

If you want to *permanently* prevent an application from logging in to Crowd, please see Deleting or Deactivating an Application.

Screenshot: 'Sessions — Applications'



RELATED TOPICS

- Managing a User's Session
- Session Configuration

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application

- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Deleting or Deactivating an Application

Deactivating an application prevents users from logging in to the application. You might do this if you are making changes to an application and need to temporarily keep users out of it.

Deleting an application removes the application's details and its directory mappings. You would typically only do this if the application is no longer required.

To deactivate an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications** tab in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link that corresponds to the application you wish to deactivate.
4. This will display the '**Application Details**' screen. Deselect the '**Active**' check-box, then click the '**Update**' button. No users will now be able to log in to the application.

To reactivate the application, follow the same steps but *select* the '**Active**' check-box.

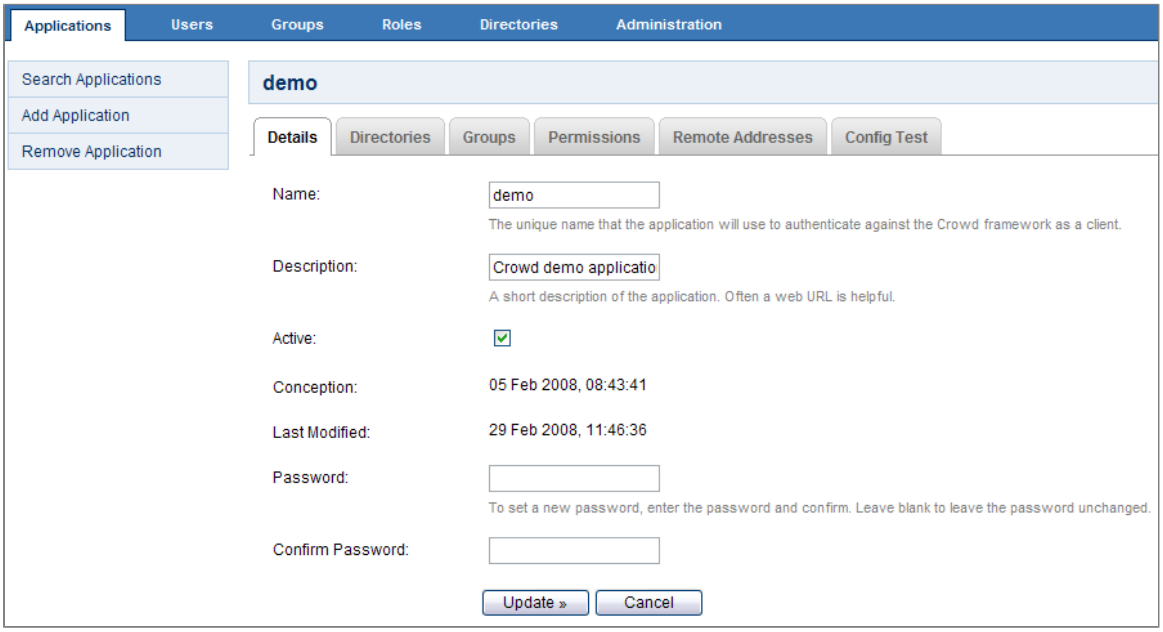
To delete an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications** tab in the top navigation bar.
3. This will display the Application Browser. Click the '**View**' link that corresponds to the application you wish to deactivate.
4. This will display the '**Application Details**' screen. Click '**Remove Application**' in the left-hand menu.

The application will be removed from Crowd and will no longer appear in the Application Browser.

You cannot delete or deactivate the '**crowd**' application (i.e. the Crowd Administration Console).

Screenshot: 'Deleting or Deactivating an Application'



RELATED TOPICS

- Using the Application Browser
- Adding an Application
  - Integrating Crowd with Atlassian Bamboo
  - Integrating Crowd with Atlassian Confluence
    - Configuring Confluence for NTLM SSO
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye 1.3.x to talk to Crowd
  - Integrating Crowd with Atlassian JIRA
    - Configuring JIRA for NTLM SSO
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
  - Integrating Crowd with Jive Forums
    - Jive SSO
  - Integrating Crowd with Spring Security
    - Integrating AppFuse - a Crowd-Spring Security Integration Tutorial
  - Integrating Crowd with Subversion
  - Integrating Crowd with a Custom Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
  - Specifying the Directory Order for an Application
  - Specifying an Application's Directory Permissions
    - Example of Directory Permissions
- Specifying which Groups can access an Application
- Viewing Users in Directories Mapped to an Application
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames, Groups and Roles for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
- Configuring Options for an Application

Crowd Documentation

Configuring Caching for an Application

Caching is used to store run-time authentication and authorisation rules, which can be expensive to calculate.

This page describes the cache that can be configured in each of the Crowd-connected applications, such as JIRA, Confluence and Bamboo. For an overview of the other types of caching offered by Crowd, please refer to Overview of Caching.

Crowd application caching is also referred to as 'client caching'.

On this page:

- Explanation of Crowd Application Caching
- Enabling Application Caching
- Extract from the ehcache.xml file
- Basic Cache Attributes
- Important Client Caches

Explanation of Crowd Application Caching

Crowd-integrated applications can store user, group and role data in a local cache. This helps improve the performance of Crowd since these applications do not have to repeatedly request information from Crowd. Generally, it is not necessary to configure application caching, although this depends on the size of your application deployments.

Enabling Application Caching

To enable application caching,

- Edit the `crowd-ehcache.xml` file, which is located in the `WEB-INF/classes` directory of your application's Crowd client. The two main properties are:
  - **diskStore**: If you have enabled disk persistence (`diskPersistent="true"`) this is the location on the file system where Ehcache will store its caching information. By default it uses `java.io.tmpdir` which is Java's default temporary file location.
  - **defaultCache**: This property has *many* configurable options. Please read the documentation provided by Ehcache to fully understand the implications and possibilities with this property. Some basic features are described below.

Some applications may enable/disable caching based on the Crowd server setting
The Crowd API allows an application to query whether caching is enabled on the Crowd server (`isCacheEnabled`). The Crowd Java client does not make use of this API feature, because it makes more sense to have application caching configured entirely on the application side. If you have a Crowd-integrated custom application which does make use of this API call, then the setting on the Crowd server will affect your application-side caching as well.

Extract from the ehcache.xml file

Below is a small snippet of the `crowd-ehcache.xml` file.

```
01.<ehcache>
02.
03.          <diskStore path="java.io.tmpdir"/>
04.
05.          <defaultCache
06.                    maxElementsInMemory="50000"
07.                    eternal="false"
08.                    overflowToDisk="false"
09.                    timeToIdleSeconds="300"
10.                    timeToLiveSeconds="300"
11.                    diskPersistent="false"
12.                    diskExpiryThreadIntervalSeconds="120"/>
13.
14.</ehcache>
```

Basic Cache Attributes

- **eternal**: This indicates that all elements in the cache will live for ever and that any time-outs will be ignored. It is strongly recommended that you set this to false.
- **timeToIdleSeconds**: This sets the maximum amount of time between an element being accessed and its expiry. If you set this value to *0*, the element will idle indefinitely.
- **timeToLiveSeconds**: This sets the maximum time between creation time of an element and its expiry. If you set this value to *0* it will live indefinitely.
- **maxElementsInMemory**: Sets the maximum number of elements that can be stored in the cache's memory. If this limit is reached, the default caching strategy **LRU** *(Least Recently Used)* will be invoked and those elements will be removed.
  An element is anything stored in Crowd's cache: a user, a group, a list of users, a list of groups, a list of user memberships, a list of group memberships.
  Hint: If you want to store everything in memory, try this value to start with:
  `(Number of users x 2) + (number of groups x 2)`

Important Client Caches

The default **maxElementsInMemory** value of 50000 should be sufficient for most Crowd-integrated applications. However, for larger installations please ensure that the **maxElementsInMemory** matches the recommended size calculation listed below:

Name of Cache:	Size Calculation:
`com.atlassian.crowd.integration-user`	The number of users in your system.
`com.atlassian.crowd.integration-group`	The number of groups in your system.
`com.atlassian.crowd.integration-parentgroup`	The number of groups in your system.
`com.atlassian.crowd.integration-group-membership`	The number of users multiplied by the number of groups *(users * groups)*. This total could be quite large, so you can optimise it by setting it to the number of users that are likely to be active at any one time. The algorithm will fall back to using the `com.atlassian.crowd.integration-all-group-members` cache (see below) before hitting the server to check.
`com.atlassian.crowd.integration-all-memberships`	The number of users in your system.
`com.atlassian.crowd.integration-all-group-members`	The number of groups in your system.

RELATED TOPICS

- Overview of Caching
- Configuring Caching for an LDAP Directory
- Authorisation Caching

- Backing Up and Restoring Data
- Configuring Server Settings
  - Authorisation Caching
  - Compression of Server Output

- Deployment Title
- Domain
- Licensing
- Session Configuration
- SSO Cookie
- Token Seed

- Configuring Trusted Proxy Servers
- Configuring your Mail Server
- Creating an Email Notification Template
- Logging and Profiling
  - Performance Profiling
- Overview of Caching
- Viewing Crowd's System Information

Crowd Documentation

Overview of SSO

Crowd provides single sign-on (SSO) across a number of applications. This means that users can log in just once, then access the applications without having to log in to each one individually. The SSO functionality is available for applications within a single domain, such as JIRA, Confluence and others. You can also extend SSO to beyond-the-firewall applications using CrowdID for OpenID and Crowd's Google Apps connector.

This page gives an overview of Crowd's SSO capabilities, plus links to detailed information on configuring Crowd and the applications concerned.

On this page:

- SSO within a Single Domain
  - How It Works
  - Configuring Crowd for SSO
  - Configuring the Applications for SSO
  - Troubleshooting SSO
- SSO Beyond the Firewall
  - Using CrowdID as an OpenID Provider
  - Using SSO with Google Apps

SSO within a Single Domain

The core Crowd functionality supports SSO across applications within a single domain, such as `*.mydomain.com`. Crowd uses a browser cookie to manage SSO. Because your browser limits cookie access to hosts in the same domain, this means that all applications participating in SSO must be in the same domain.

Example 1: If you wish to have single sign-on (SSO) support for ***.mydomain.com**, you will need to configure the SSO domain in Crowd as `.mydomain.com` — including the full stop ('.') at the beginning. All your Crowd-connected applications must be in the same domain. For example:

Crowd	`crowd.mydomain.com`	✓
JIRA	`jira.mydomain.com`	✓
Confluence	`confluence.mydomain.com`	✓
FishEye	`fisheye.mydomain.com`	✓
FishEye in different domain	`fisheye.example.com`	✗

Example 2: If you wish to have single sign-on (SSO) support for **mydomain.com/***, you will need to configure the SSO domain in Crowd as `mydomain.com`. All your Crowd-connected applications must be in the same domain. For example:

Crowd	`mydomain.com/crowd`	✓
JIRA	`mydomain.com/jira`	✓
Confluence	`mydomain.com/confluence`	✓
FishEye	`mydomain.com/fisheye`	✓
FishEye in different domain	`example.com/fisheye`	✗

You can find information the comparison of host name strings in RFC 2965 (pages 2 and 3).

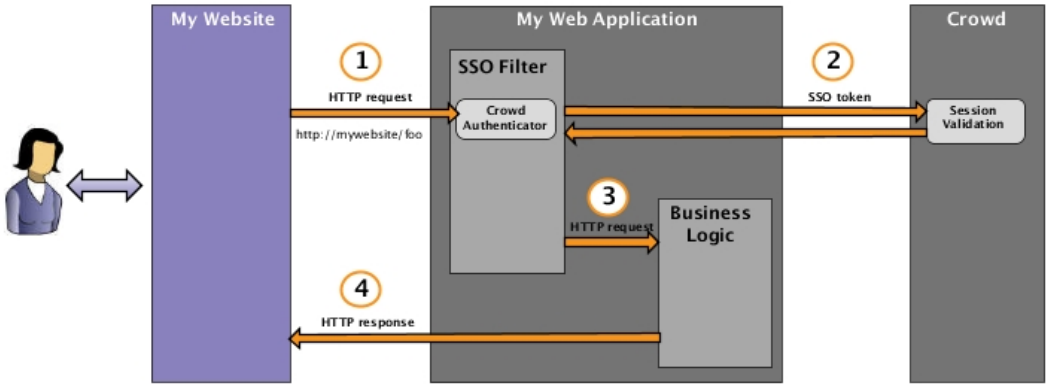
You can configure the SSO domain via the Crowd Administration Console, as described in the documentation.

How It Works

The diagram below gives a conceptual overview of an HTTP request passing through an SSO filter and moving directly through the application business logic to create the response. (Click the link below the diagram to see a larger version.)

The diagram shows the 'happy path' only, assuming that:

- The user has already logged in to an application that is configured to participate in SSO. If the user has already logged in to one application, they will not need to log in again when accessing another application in the same domain.
- The request passes all authentication and authorisation checks.



The diagram illustrates the following steps:

- **Step 1:** The HTTP request with an SSO cookie.
  - The user has already logged in to an application that is part of the SSO environment.
  - The user accesses a new application within the SSO environment, or performs some other action on the website.
  - The browser creates an HTTP request, bundles all the cookies for the domain and sends the request to the web application. This includes the SSO cookie, since the user has already logged in.
  - The request is trapped by the SSO filter in the web application's security framework. This filter may be provided by Atlassian Seraph, by Spring Security, by another framework or via custom code.
  - (If the user has not logged in, the filter re-directs the user to the login screen at this point. But we're assuming the user has logged in.)
  - The Crowd authenticator finds the SSO cookie, extracts the SSO token and passes the token to Crowd. The Crowd authenticator is a plugin to the security framework (Atlassian Seraph, Spring Security, or others).
- **Step 2:** Validation of the SSO token.
  - Crowd validates the session token. If another application in the same domain has already authenticated the user, Crowd will validate the existing authentication.
  - If the session has expired, Crowd re-directs the user to the login screen and re-authenticates the user.
  - Crowd checks that the user is authorised to access the application.
  - If the user does not have the required permissions, Crowd re-directs the user to the login screen.
  - Once validation is successful, Crowd passes the validated token back to the application's SSO filter.

    If the session is still valid, the user will not need to log in again even if accessing a different application. The authentication and authorisation will be transparent to the user.
- **Step 3:** Processing of the HTTP request.
  - The application's SSO filter passes the request to the business logic handler. (In a Java application, this is the servlet.)
  - The business logic handler processes the request and builds the response.
- **Step 4:** The HTTP response.
  - The application sends the response back to the browser.

Here is a an overview of servlet filters from Sun and a useful tutorial from O'Reilly.

The SSO filter may be provided by a security framework or by custom code as follows:

Security Framework or Custom Code	Comments
Framework: Atlassian Seraph	Most of the Atlassian applications use Seraph. The Crowd documentation tells you how to integrate SSO into Confluence, JIRA, Bamboo, etc. If you are integrating a custom application with Crowd, you may also decide to use Seraph as your security framework.

Framework: Spring Security	You may have a web application that uses the Spring Security framework and that you are now integrating with Crowd. The Crowd documentation tells you how to integrate SSO into a Spring Security-based application. A point of interest: Crowd uses the Spring Security framework, and so does the Crowd 'demo' application.
Framework: Acegi Security (old)	You may have a web application that uses the Acegi Security framework and that you are now integrating with Crowd. The Crowd documentation tells you how to integrate SSO into an Acegi-based application. Note that Acegi Security is an earlier version of Spring Security.
Custom authentication for Atlassian FishEye and Crucible	Crowd provides a custom integration with FishEye and/or Crucible, including SSO. See the Crowd documentation.
Crowd API for your custom application	When integrating your own web application with Crowd, you can use the Crowd API to implement SSO. • We recommend that you use the SOAP API for long-term compatibility. • If you have a Java application, you can use the Java client libraries shipped with Crowd, but please be aware that they may change between releases. You may need to re-compile your source and possibly change a package name. • There are a number of third-party language bindings and application connectors developed by Crowd users. You can see them in the Atlassian Plugin Exchange. (Please check the '**Plugin Details**' for each plugin to see if it is supported by Atlassian.)

Configuring Crowd for SSO

Below are the configuration settings which affect SSO:

Short Description	More Information
Set your SSO domain	Set the domain via the Crowd Administration Console, as described in the documentation.
Optional: Configure Trusted Proxy Servers	Configure Crowd to trust a proxy's IP address, if you are running applications behind one or more proxy servers. See the documentation.
Optional: Enforce a secure connection, such as SSL, for all SSO requests	You can specify that the 'secure' flag is set on the SSO cookie, as described in the documentation. Unsecured connections will be rejected, including the Crowd Administration Console if not accessed via SSL.

Configuring the Applications for SSO

When integrating an application with Crowd, you will configure the application to use Crowd as a centralised authentication repository. For most applications, **but not all**, you can also choose to configure SSO. This is described in detail for each application:

- Integrating Crowd with Atlassian Bamboo
- Integrating Crowd with Atlassian Confluence
- Integrating Crowd with Atlassian CrowdID
- Integrating Crowd with Atlassian Crucible
- Integrating Crowd with Atlassian FishEye
- Integrating Crowd with Atlassian JIRA
- Integrating Crowd with Acegi Security
- Integrating Crowd with Apache
- Integrating Crowd with Jive Forums
- Integrating Crowd with Spring Security
- Integrating Crowd with Subversion
- Integrating Crowd with a Custom Application

Troubleshooting SSO

See Troubleshooting SSO with Crowd.

SSO Beyond the Firewall

Crowd allows you to extend SSO to beyond-the-firewall applications using CrowdID and Crowd's Google Apps connector.

Using CrowdID as an OpenID Provider

Crowd allows you to host an OpenID provider, called CrowdID, so that your users have a single point of authentication for all OpenID-enabled websites. Refer to the CrowdID Administration Guide and CrowdID User Guide.

OpenID is an open, free protocol which allows a user to have a single identifier for logging in to any OpenID-enabled website. The website will communicate with a specific OpenID provider (in this case, your CrowdID server) when attempting to verify the user's login. For example, if your team uses 37signals' CRM tool Highrise, using Crowd's OpenID provider means you can get SSO between Highrise and your behind-the-firewall applications for all your team.

Using SSO with Google Apps

Crowd offers SSO with Google Apps via the Google Apps connector shipped with your Crowd installation. This means that your users can log in just once and then move between Google Apps and other applications like JIRA, Confluence, etc.

RELATED TOPICS

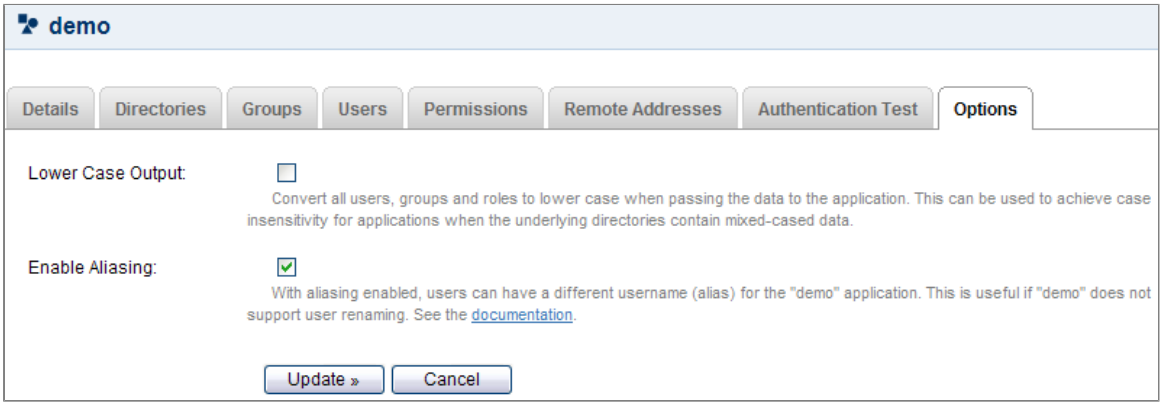
Managing Applications
System Administration
Crowd Documentation

Configuring Options for an Application

Once you have added an application to Crowd, you can configure various options for that applicationn on the '**Options**' tab. Click the links below for information about each option:

- Lower Case Output
- Enable Aliasing

Screenshot: Application Options



RELATED TOPICS

Managing Applications
Crowd Documentation

Managing Users, Groups and Roles

In Crowd, users are referred to as *user entity objects* or just *users*.

Groups and roles are known as *permission container objects*. Groups are particularly important in Crowd, as they are often used to control access to applications. Note also that the crowd-administrators group confers Crowd administration rights to its members.

Roles are used less frequently, depending on the requirements of individual applications.

Crowd's role-based access control could be enhanced
At present, the implementation of roles in Crowd is identical to the implementation of groups. Additional development work would be needed to differentiate the functionality of roles from groups. If you would like to help us to design better role-based access control, please add a comment to the improvement request CWD-931, letting us know how you would like to see it work.

This section describes how to add/edit users, groups and roles via the Crowd Administration Console. Note that the ability to do this depends on the permissions of the directory which contains the users, groups and roles.

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User

- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd
  - Adding a Sub-Group
  - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Using the User Browser

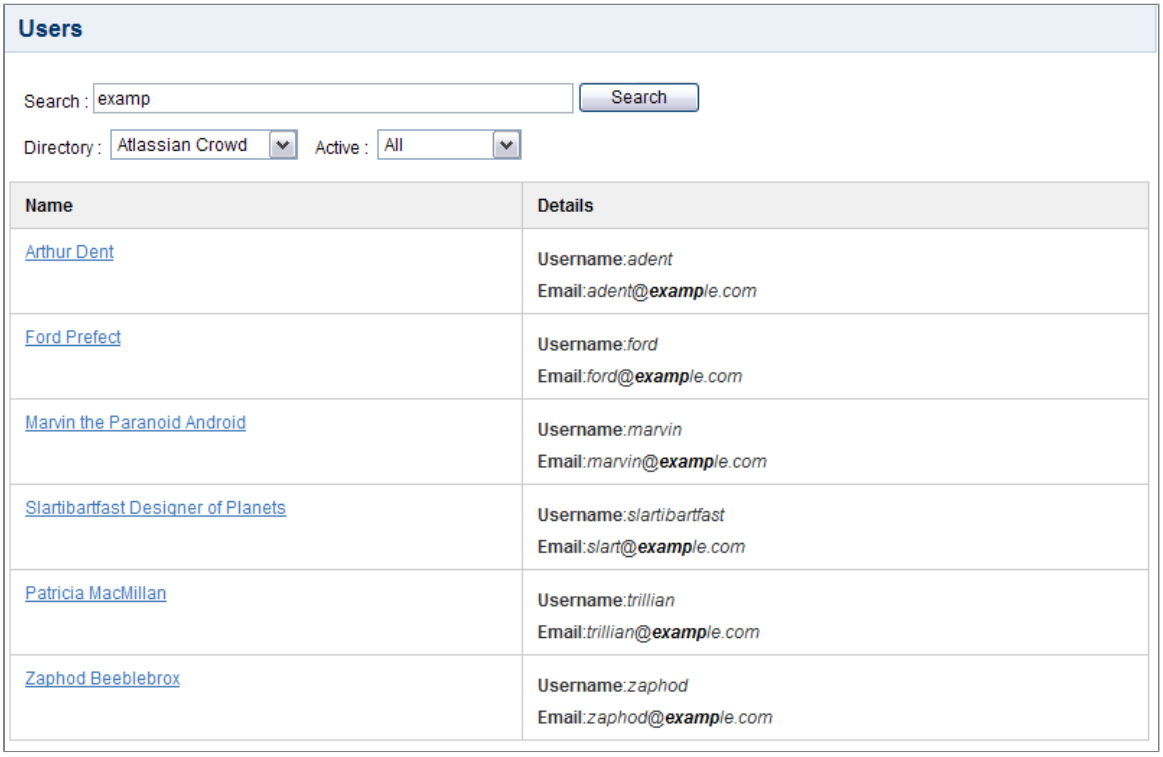
In Crowd, users are referred to as *user entity objects* or just *users*.

The User Browser allows you to search, view, add and edit users within a specified directory.

To use the User Browser,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' tab in the top navigation bar.
3. The User Browser will appear. Select the directory in which you are interested.
4. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the user's name, email address or username. Leave the search box empty to retrieve all users.
5. You can refine your search by choosing '**Active**' or '**Inactive**' users. (An 'Inactive' user is typically someone who has left your organisation.)
6. Click the '**Search**' button. Crowd will list all the users in the selected directory who match your search criteria.
   - A maximum of 100 users will appear on a page.
   - If there are more than 100 users that match the search, the '**Next**' and '**Previous**' links will appear at the bottom of the page, so that you can move from one page to the next.
7. If you want to display fewer users, you can change the search criteria and click 'Search' again.
8. To view or edit a user's details, click the link on the user's name.

Screenshot: 'User Browser'



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session

- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd
  - Adding a Sub-Group
  - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Adding a User

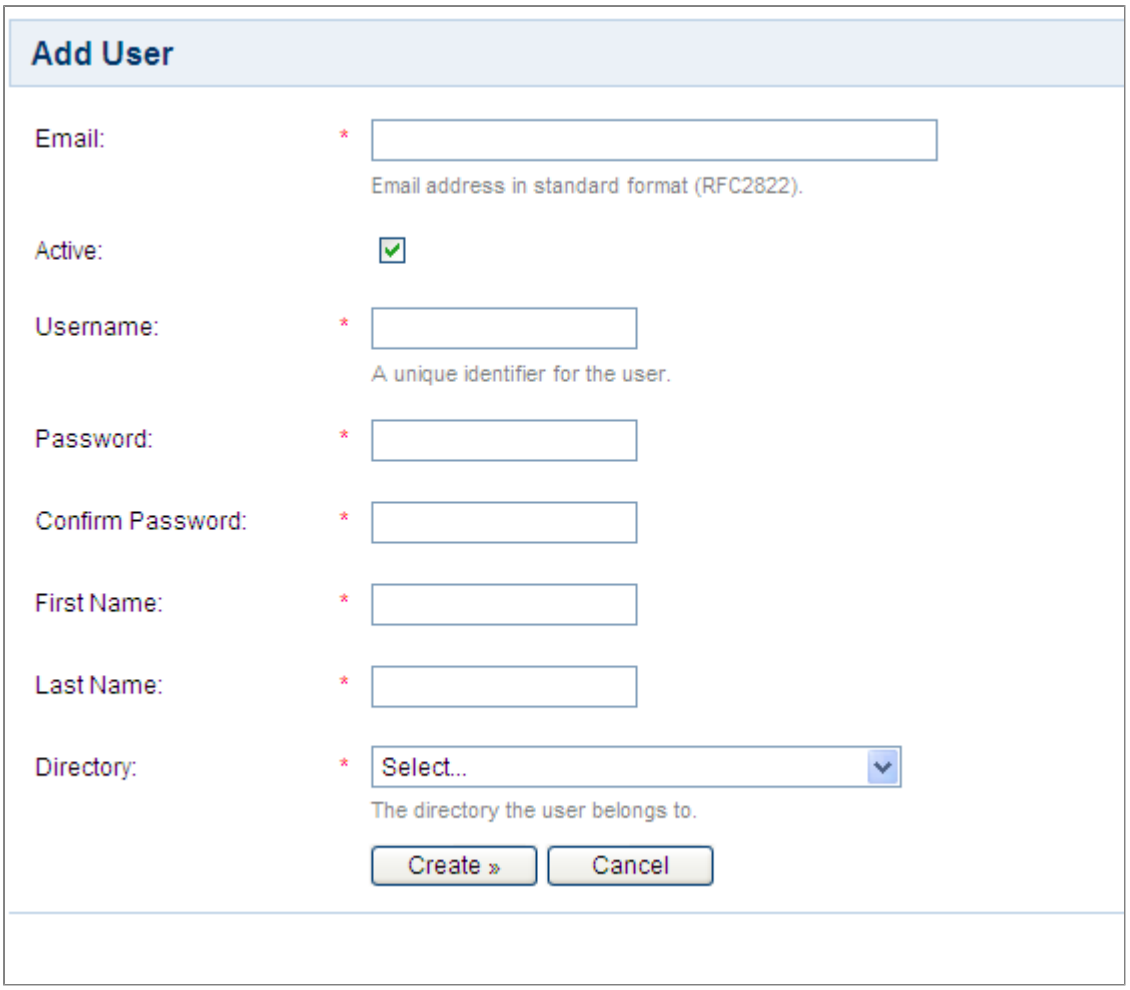
In Crowd, users are referred to as *user entity objects* or just *users*.
You can either import users into Crowd in bulk (see Importing Users and Groups into a Directory), or add them individually as described below.

To add a user,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' tab in the top navigation bar.
3. This will display the User Browser. Click '**Add User**' in the left-hand menu.
4. Complete the following fields:
   - **Email** — The email address of the user. Email addresses must follow the RFC2822 format.
   - **Active** — Only deselect this if you wish to deny the user access to the Crowd-integrated applications.
   - **Username** — The user's login name. Within a given directory, the username must be unique. Note that you cannot change the username once the user has been created.
   - **Password** — The user's password.

     If you have configured an email server and a notification template, Crowd will send the user an email notification about their new password.
   - **Confirm Password** — Enter the same password again, to ensure that you have typed it correctly.
   - **First Name** — The user's first name.
   - **Last Name** — The user's last name.
   - **Directory** — The directory to which the user will be added. Note that the user cannot be moved to a different directory once the user has been created.
5. Click the '**Create**' button to add the user.
6. After creating the user, you will be able to specify the user's attributes and group/role membership. If you wish, you can also verify that the user can log in to appropriate applications.

Screenshot: 'Add User'



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
    - Adding a Group or Role
    - Deleting or Deactivating a Group
- Managing Group Members
    - Automatically Assigning New Users to Groups
    - Adding Users to a Group
    - Removing Users from a Group
    - Nested Groups in Crowd
    - Adding a Sub-Group
    - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Deleting or Deactivating a User

Deactivating a user prevents the user from logging in to any applications that use the Crowd framework and also excludes the user from the license count. You would typically do this when a user leaves your organisation.

Deleting a user removes the user completely from the relevant directory.

Deactivating instead of Deleting

We recommend that you deactivate a user rather than delete them, in case some applications contain historical data, such as documents

that the user has created. Read more.

> **Deactivating a user that resides in LDAP**
> For applications that need users to exist for historical data (such as JIRA), you should recreate the user and mark it inactive in a Crowd Internal Directory before deleting from your LDAP directory.

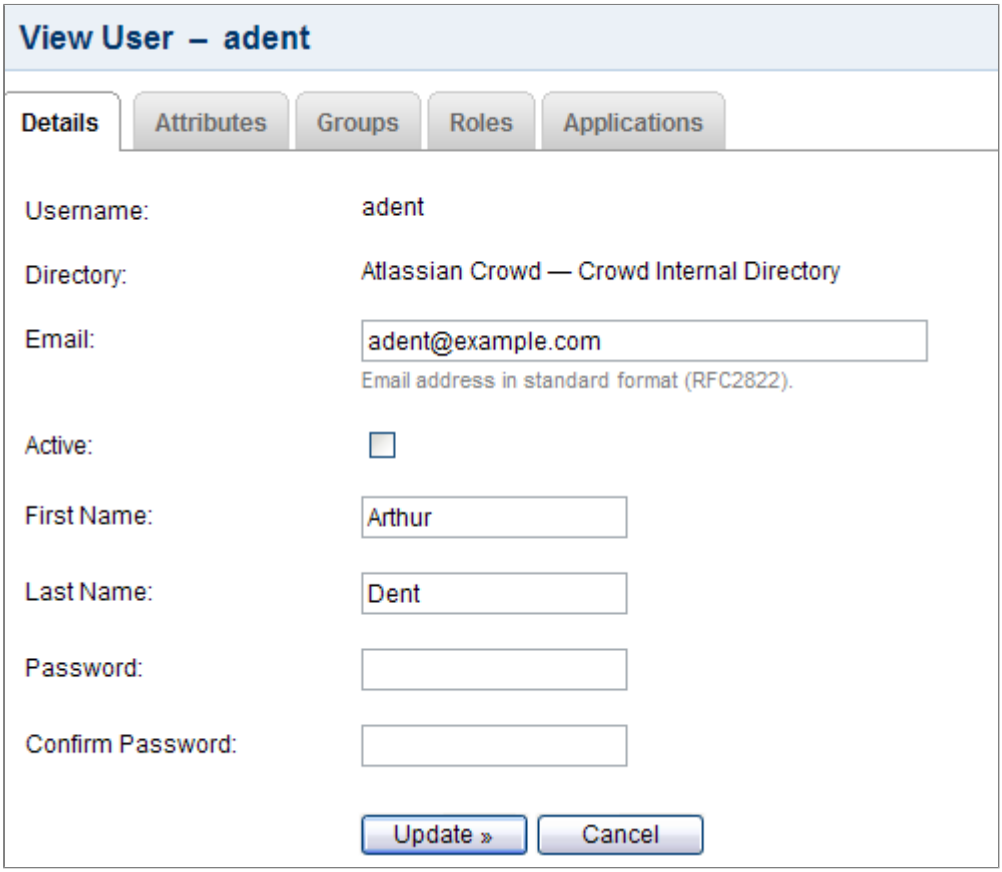
Deactivating a User

To deactivate a user,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, search for the user you wish to deactivate, and click the link on the user's name.
4. This will display the '**User Details**' screen. Deselect the '**Active**' checkbox, then click the '**Update**' button.

The user will now be unable to log in to any applications that use the Crowd framework.

Screenshot: Deactivating a user



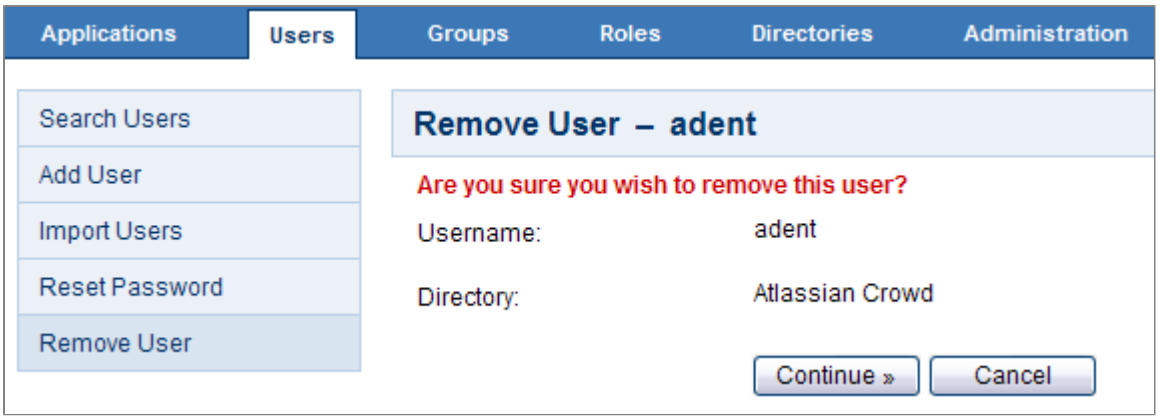
Deleting a User

To delete a user,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, search for the user you wish to delete, and click the link on the user's name.
4. This will display the '**User Details**' screen. Click '**Remove User**' in the left-hand menu. Confirm the deletion when prompted.

The user will be removed from the relevant directory and will no longer appear in the User Browser.

Screenshot: Deleting a user



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd
  - Adding a Sub-Group
  - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Managing a User's Session

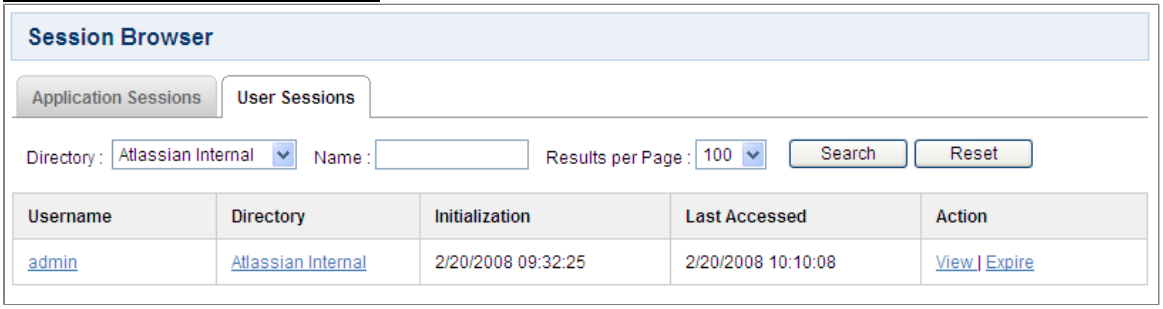
For any given directory, Crowd allows you to see which users are currently logged in to one or more applications that use the Crowd framework.

You can also force any session to expire, that is, you can log the user out of Crowd.

To see which users are currently logged in to Crowd,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Current Sessions**' in the left-hand menu.
4. This will display the '**Session Browser**'. Click the '**User Sessions**' tab.
5. Select the directory containing the users in which you are interested, and click the '**Search**' button.
6. This will display a list of all users, within your chosen directory, who are currently logged in to the Crowd framework.
   You can refine your search by specifying a user's '**Name**' (note that this is case-sensitive).

Screenshot: 'Session Browser — Users'



To log a user out of Crowd,

1. Login to the Crowd Administration Console.
2. Click the '**Administration**' link in the top navigation bar.
3. Click '**Current Sessions**' in the left-hand menu.
4. Click the '**User Sessions**' tab.
5. This will display a list of all users which are currently logged in to the Crowd framework. Click the user's '**Expire**' link.

If you want to *permanently* prevent a user from logging in to Crowd, please see Deleting or Deactivating a User.

RELATED TOPICS

Managing an Application's Session
Session Configuration

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
- Managing Group Members
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Editing a User's Details and Password

In Crowd, users are referred to as *user entity objects* or just *users*.
To edit a user's details,

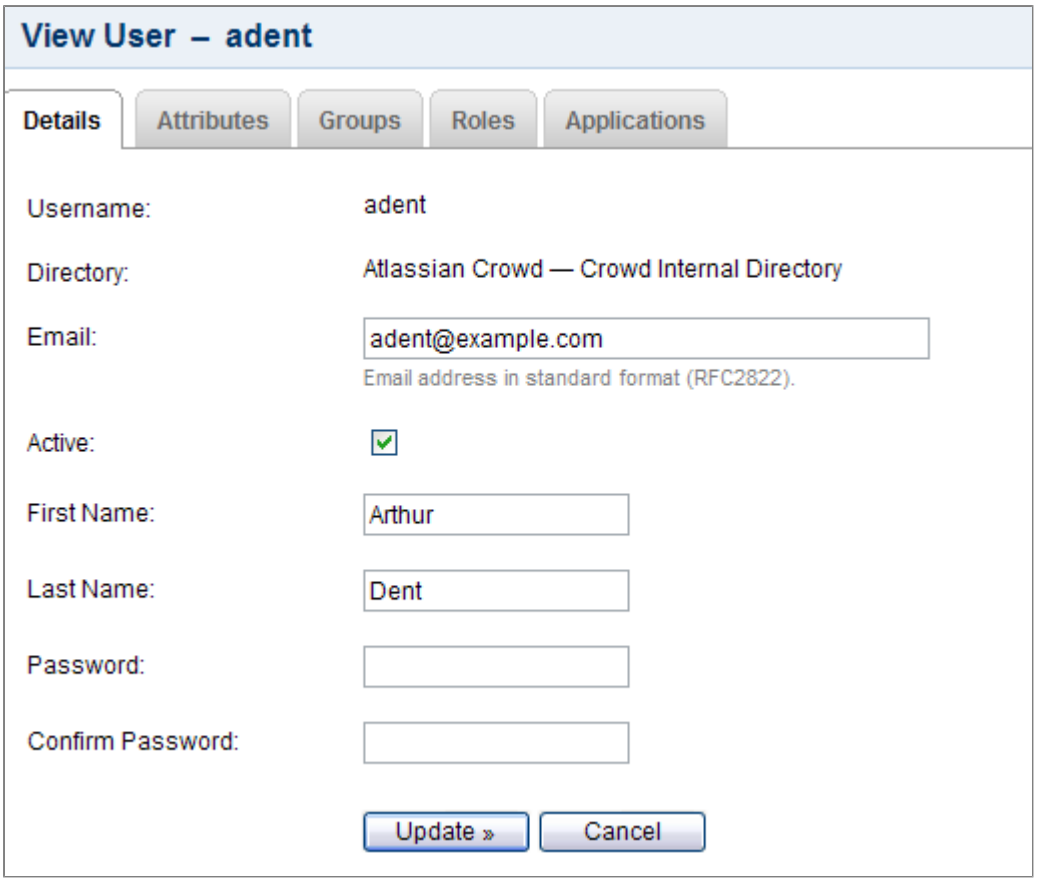
1. Log in to the Crowd Administration Console.
2. Click the '**Users**' tab in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, search for the user you want to update, and click the link on the user's name.
4. This will display the '**User Details**' screen.
5. Edit the details as required, then click the '**Update**' button.

To change a user's password,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' tab in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, search for the user you want to update, and click the link on the user's name.
4. This will display the '**User Details**' screen. You can *either*:
    - Enter the new password, then click the '**Update**' button.
      OR
    - Click '**Reset Password**' in the left-hand menu. This will generate a new password (i.e. one which you do not know) and email it to the user.

If you have configured an Email Server and a Notification Template, Crowd will send the user an email notification about their new password.

Screenshot: 'User Details'



Users can update their own profiles
Authorised Crowd users can log in to the Self Service Console and update their own user profiles, as described in the Crowd User Guide.

RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
- Managing Group Members
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Specifying a User's Attributes

In Crowd, users are referred to as *user entity objects* or just *users*.
A user's default *attributes* are specific to the directory to which the user belongs. You can add other attributes (e.g. address, phone number, date of birth) manually as required.

Cannot add attributes to LDAP directories
You cannot add new attributes to directories connected via Crowd's LDAP connector, although you can update the existing supported attributes as described in our LDAP connector documentation. Any new attributes added via the Crowd Administration Console will simply not appear in the directory.

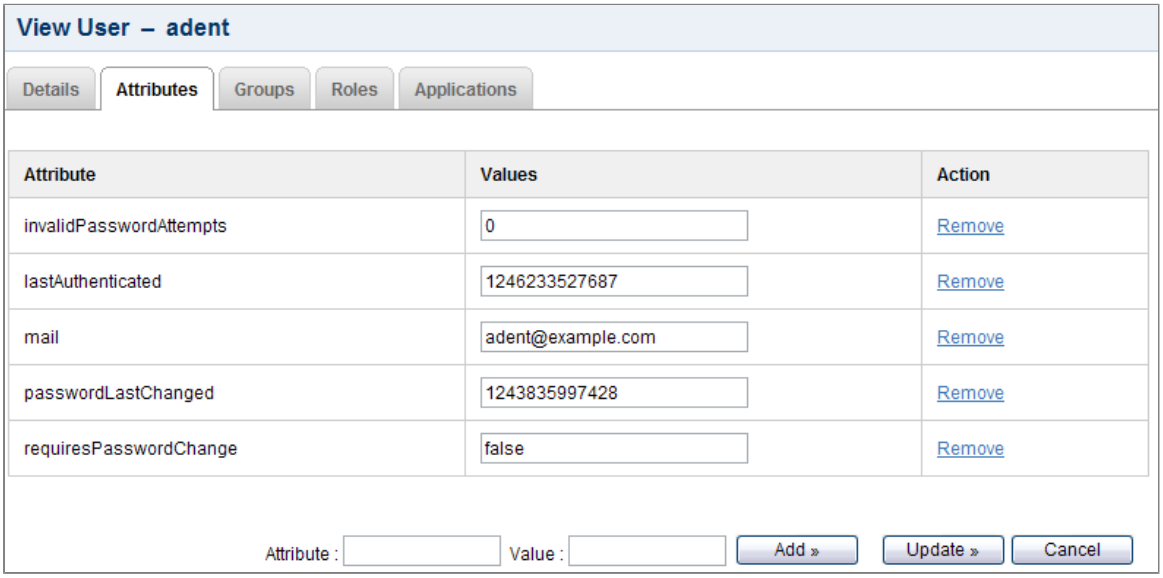
To edit a user's attributes,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. The User Browser will appear. Select the relevant directory, search for the user you want to update, and click the link on the user's name.
4. The '**User Details**' screen will appear. Click the '**Attributes**' tab.

- **To add a new attribute,**
  ⚠ You cannot add an attribute to an LDAP directory — see note above.
  1. Enter the name of the new attribute (e.g. `phone`) in the '**Attribute**' field at the bottom of the screen.
  2. Enter the value of the new attribute (e.g. `0123456789`) in the '**Value**' field at the bottom of the screen.
  3. Click the '**Add**' button.
- **To edit an existing attribute,** edit the corresponding field in the '**Values**' column, then click the '**Update**' button.
- **To delete an attribute,** click the corresponding '**Remove**' link in the '**Action**' column.

ℹ Note that some attributes may correspond to particular fields on the User Details screen. However, attributes are optional whereas the 'Details' fields are all required.

Screenshot: 'User Attributes'



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
- Managing Group Members
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Specifying a User's Aliases

A single user can have different usernames in different applications. These different usernames are called 'aliases'. As a Crowd administrator, you can manage each user's aliases for the applications the user is authorised to access.

On this page:

- Enabling User Aliasing for an Application
- Specifying a User's Aliases
- Examples and Use Cases
- Illustration

Enabling User Aliasing for an Application

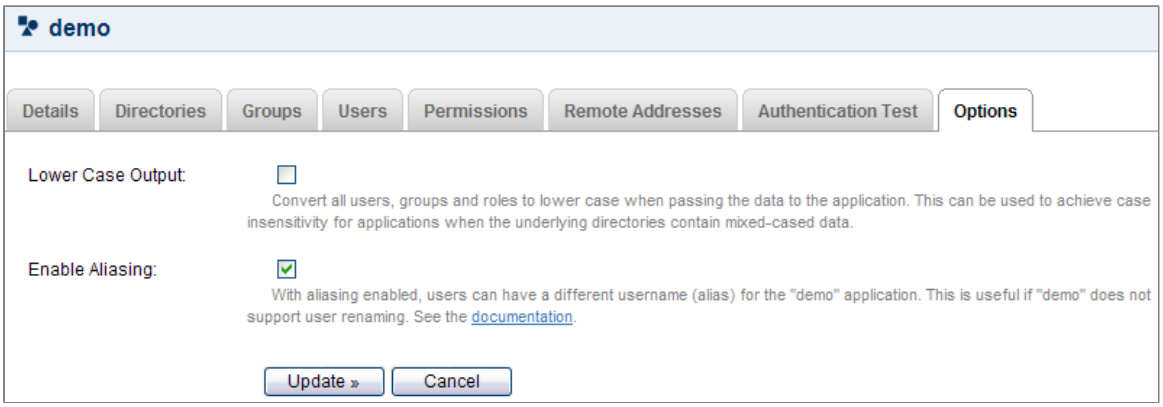
You can choose to enable or disable aliasing for each application. By default, user aliasing is disabled.

User aliasing can reduce the performance of your user directory, especially on user searches.

To enable user aliasing for an application,

1. Log in to the Crowd Administration Console.
2. Click the '**Applications**' tab in the top navigation bar.
3. The Application Browser will appear. Click the link on the name of the application you wish to configure.
4. The '**View Application**' screen will appear. Click the '**Options**' tab.
5. Put a tick in the checkbox labelled '**Enable Aliasing**'.
6. Click the '**Update**' button.

Screenshot: Application options



Specifying a User's Aliases

You can add and remove aliases via the user management screens in the Crowd Administration Console.

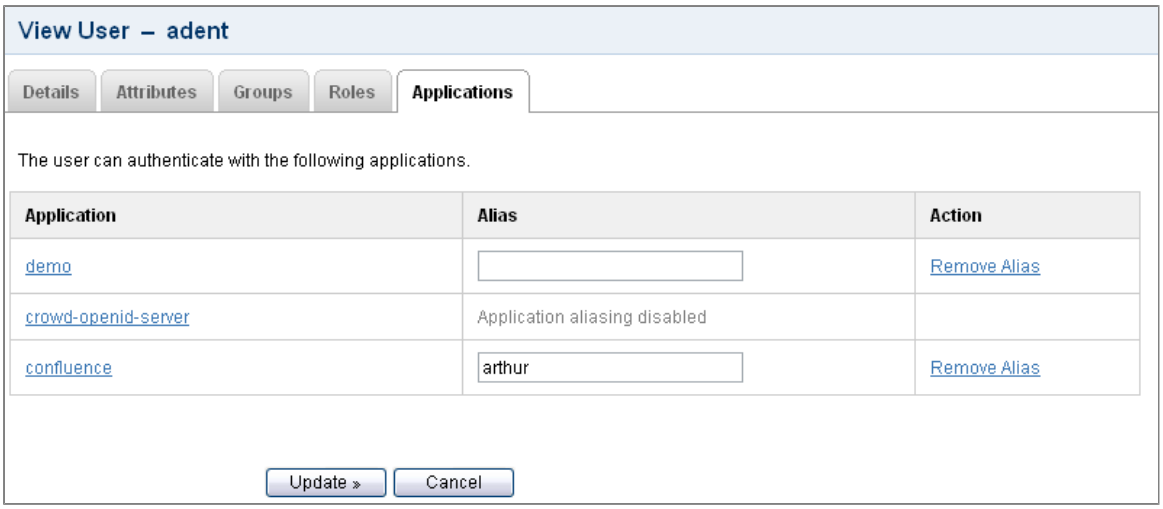
To edit a user's aliases,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, find the user in that you want to update, then click the link on the user's name.
4. The '**User Details**' screen will appear. Click the '**Applications**' tab.

- **To add an alias for the user,**
    1. Scroll down until you find the application to which the alias applies.

       For example, if the user's primary username is 'adent' but he has a username of 'arthur' in Confluence, then you need to find your Confluence application.
    2. Type the value of the new alias (e.g. 'arthur') into the '**Alias**' field next to the application.
    3. Click the '**Update**' button.
- **To edit an existing alias,** update the corresponding field in the '**Alias**' column, then click the '**Update**' button.
- **To remove an alias,** click the corresponding '**Remove Alias**' link in the '**Action**' column.

Screenshot: Specifying a user's aliases



Examples and Use Cases

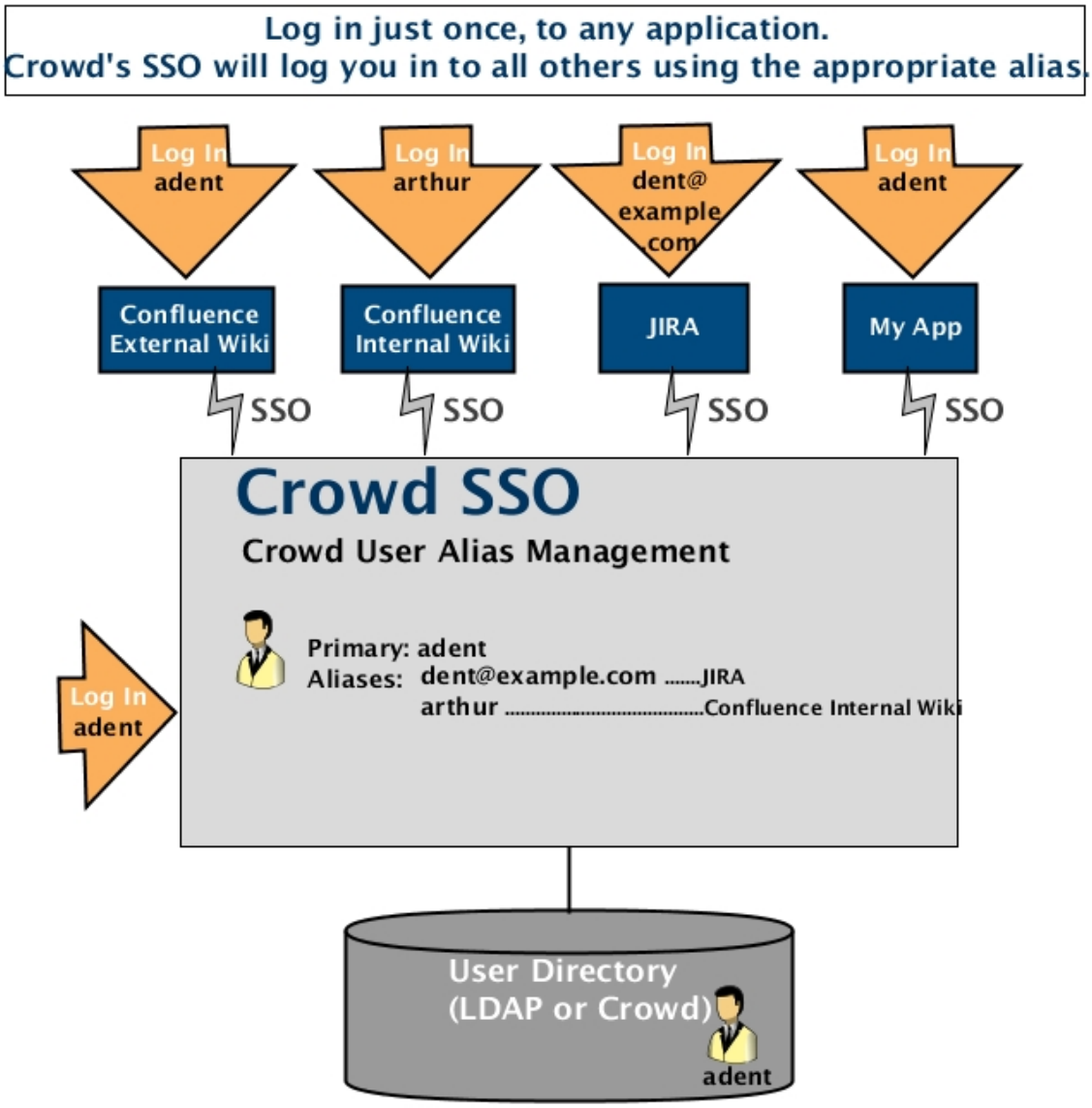
An example: Arthur Dent might have username 'dent@example.com' in your JIRA issue tracker, 'arthur' in your internal Confluence wiki and 'adent' in your public-facing Confluence wiki.

- Using Crowd, you can link a number of usernames as aliases of Arthur's primary login ID.
- Arthur can log in just once, to any Crowd-connected application. He will be automatically logged into the other applications via single sign-on (SSO).
- When logging in to a specific application (e.g. Confluence), Arthur must use the specific username (alias) for that application, e.g. 'arthur'.
- When logging in to Crowd, Arthur must use his primary login i.e. the one in the directory, e.g. 'adent'.

Here are some cases where Crowd's user aliasing may be useful:

- Aliasing allows you to work around the problem that occurs when you want to implement a single user base for a number of existing systems, where users may have different usernames in each system.
- When someone gets married or changes their name, you may wish to rename a user in your LDAP directory, such as Microsoft Active Directory. To avoid problems in applications which do not allow user renaming, you can now link the new LDAP username to an alias in Crowd.
- Some systems may use email addresses as usernames, while in others this may expose users to email spambots. Using Crowd aliasing, you can use different username formats to suit your application requirements.

Illustration



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
- Managing Group Members
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Granting Crowd Administration Rights to a User

Members of the '**crowd-administrators**' group have administration privileges — that is, they can:

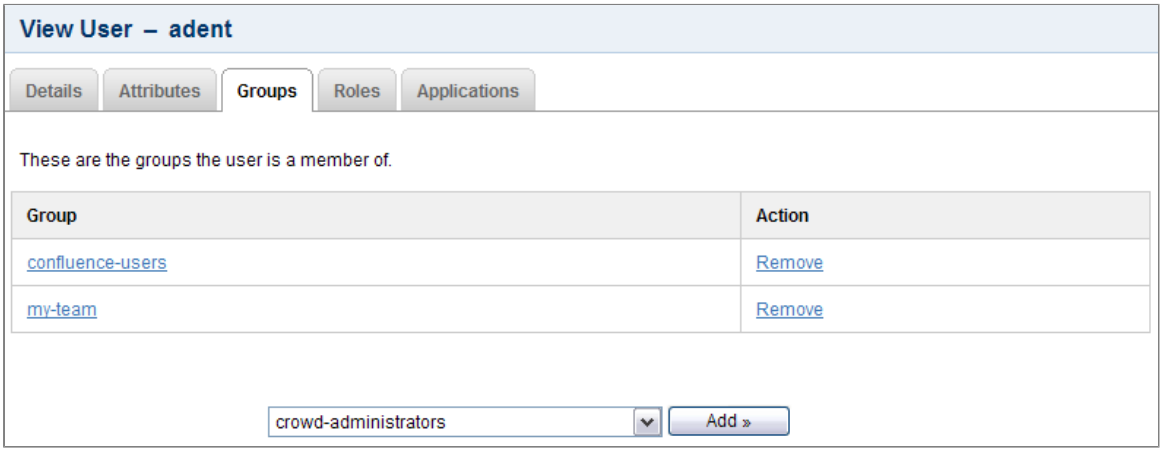
- Access the Crowd Administration Console and perform the functions described in the Crowd Administration Guide.
- Access the CrowdID '**Administration**' menu and perform the functions described in the CrowdID Administration Guide.

The '**crowd-administrators**' group is automatically created in your default directory when you install Crowd. (See Running the Setup Wizard .) If you need to grant Crowd administration rights to users in other directories, you can create a '**crowd-administrators**' group in any or all of your other directories and map the directories to the '**crowd**' application.

To grant administration privileges to a user,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' tab in the top navigation bar.
3. The User Browser will appear. Select the relevant directory, search for the user you want to update, and click the link on the user's name.
4. The '**User Details**' screen will appear. Click the '**Groups**' tab.
5. A list of the user's current groups (if any) will appear. Select the '**crowd-administrators**' group from the dropdown box below the list, then click the '**Add**' button.

Screenshot: Granting Crowd administrator rights to a user



If you wish, you can use a different or additional group to contain your Crowd administrators. To do this, map your chosen group(s) to the '**crowd**' application as described in Specifying which Groups can access an Application. Note that CrowdID administrators, however, must always belong to the 'crowd-administrators' groups.

RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
- Managing Group Members
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Granting Crowd User Rights to a User

This page tells you how to authorise users to access Crowd, without giving them Crowd administration rights. Only Crowd administrators can authorise other users to access Crowd.

Administrators and Non-Administrators

The Crowd Administration Console presents the full range of Crowd administration functionality to authorised Crowd administrators.

Authorised Crowd users who are **not** administrators can also access the Crowd Console. They will see a subset of functionality, which we call the 'Self-Service Console'. Refer to the Crowd User Guide for details of this functionality.

Non-administrators cannot affect other users or the Crowd installation
Granting Crowd user rights will give your users the power to update their own profiles and passwords and view their authorisation details. But they will not be able to view or update other user profiles, nor perform any Crowd administration functions.

Authorising Non-Administrators to Use the Crowd Self-Service Console

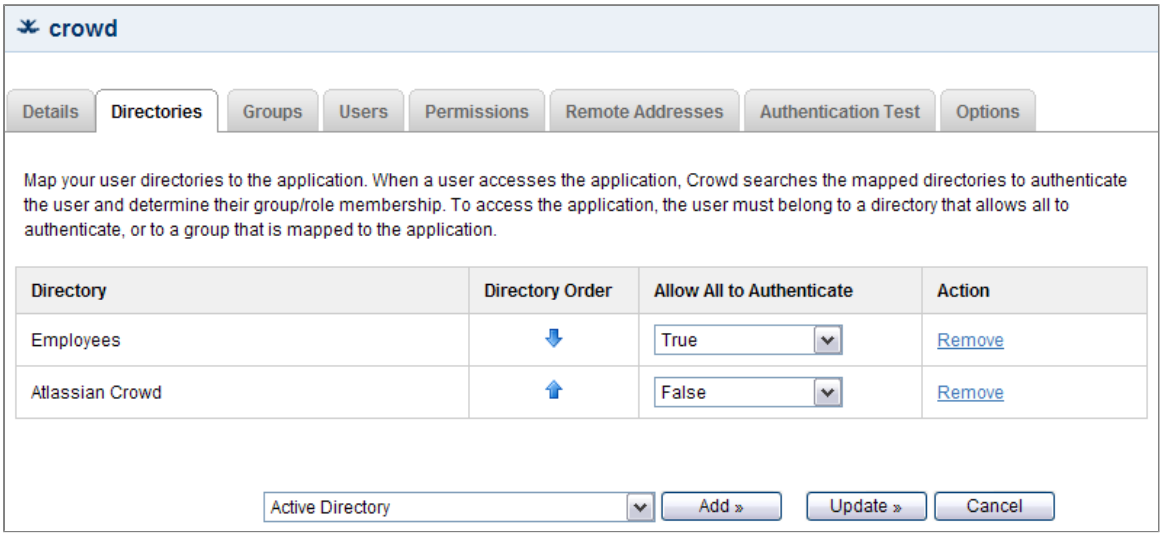
To authorise a non-administrator to use Crowd, you should ensure that both of the following are true:

- The person's username is in a user directory where all users are authorised to use Crowd. See the instructions below.
- The person is **not** a member of a group mapped to the 'crowd' application. (Group members will have Crowd administration rights.)

To grant an entire directory access to Crowd,

1. Log in to the Crowd Administration Console.
2. Map your chosen user directory to the '**crowd**' application.
3. On the '**Directories**' tab, set the '**Allow All to Authenticate**' option to '**True**'.
4. Add the user(s) to the directory, if not already added.

Screenshot: Granting an entire directory access to the 'crowd' application



RELATED TOPICS

Granting Crowd Administration Rights to a User
Crowd User Guide
Crowd Documentation

Managing Groups and Roles

This page introduces you to groups and roles in Crowd.

About Groups and Roles

Groups and roles are known as *permission container objects*. Groups are particularly important in Crowd, as they are often used to control access to applications. Note also that the crowd-administrators group confers Crowd administration rights to its members.

Roles are used less frequently, depending on the requirements of individual applications.

Crowd's role-based access control could be enhanced
At present, the implementation of roles in Crowd is identical to the implementation of groups. Additional development work would be needed to differentiate the functionality of roles from groups. If you would like to help us to design better role-based access control, please add a comment to the improvement request CWD-931, letting us know how you would like to see it work.

About nested groups

Some user directories allow you to define a group as a member of another group. Groups in such a structure are called '**nested groups**'. In Crowd, you can map any group to an application, including a group which contains other groups. Crowd supports nested groups for LDAP directory connectors, Crowd internal directories, Delegated Authentication directories and custom directories. You can **enable or disable** support for nested groups on each directory individually. For more information, refer to the documentation on configuring a directory.

For more details about nested groups, refer to Nested Groups in Crowd.

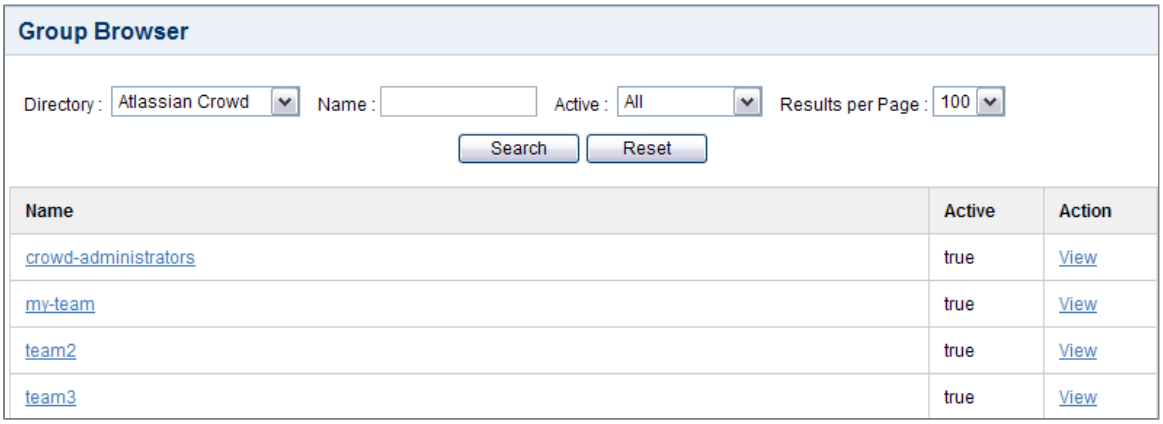
About the Group Browser and the Role Browser

The Group Browser and the Role Browser are very similar. They allow you to search, view, add and edit the various groups and roles stored within a specified directory.

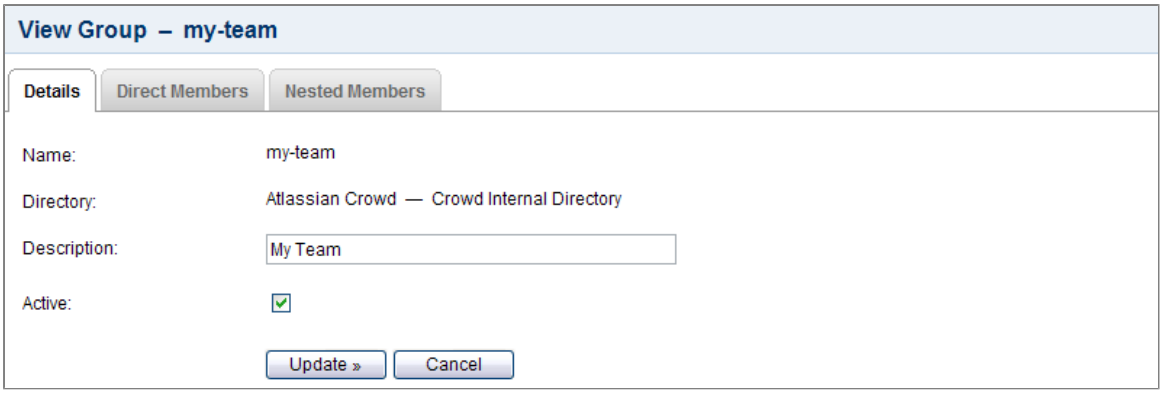
To use the Group Browser,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' tab in the top navigation bar.
3. The Group Browser will appear. Select the directory in which you are interested, then click the '**Search**' button to list all the groups that exist in that directory.
   You can refine your search by specifying a '**Name**' or by choosing '**Active**' or '**Inactive**' groups.
4. To view or edit a group's details, click the link on the group name.
5. Click the '**Direct Members**' tab to view the immediate members of the group, including users and other groups.
6. Click the '**Nested Members**' tab to view all users who are included in the group and in its sub-groups
7. You can read more about group members in Managing Group Members.

Screenshot 1: Group Browser



Screenshot 2: Viewing and updating group details



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd

    - Adding a Sub-Group
    - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Adding a Group or Role

Groups and roles are known as *permission container objects*. Groups are particularly important in Crowd, as they are often used to control access to applications. Note also that the crowd-administrators group confers Crowd administration rights to its members.

Roles are used less frequently, depending on the requirements of individual applications.

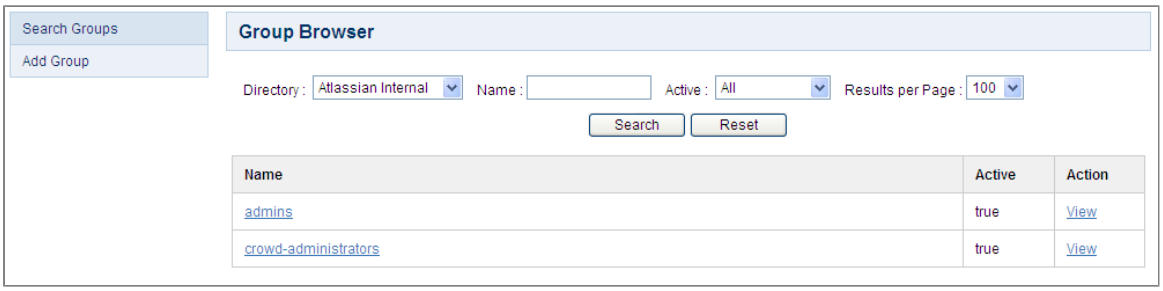
To add a group or role,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' or '**Roles**' link in the top navigation bar.
3. This will display the Group Browser (or Role Browser). Click '**Add Group**' or '**Add Role**' in the left-hand menu.
4. Complete the fields as described in the table below, then click the '**Create**' button.

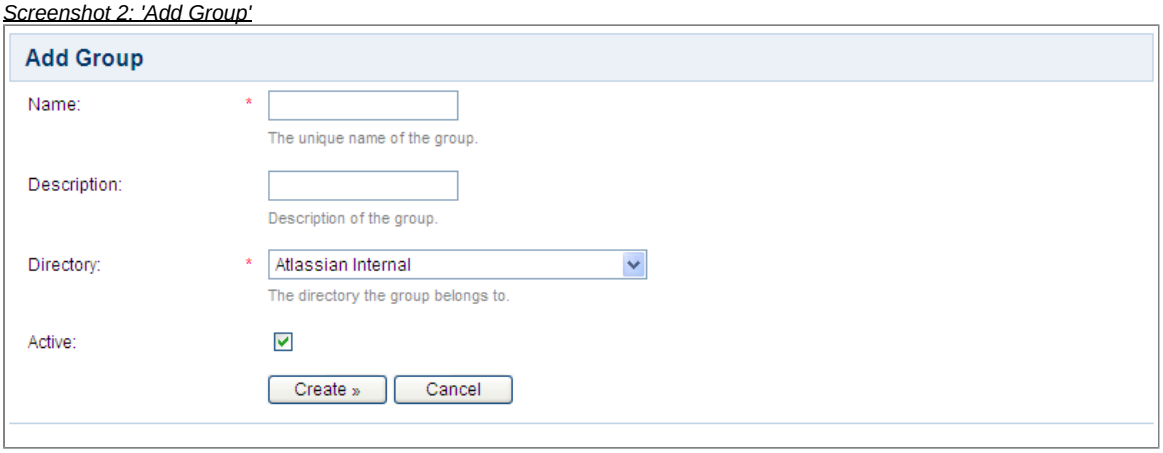
   You can now add users to the new group or role. If your directory supports nested groups, you can now add sub-groups.

Field	Description
Name	The unique name of the group or role. Within a given directory, the Name must be unique. Note that the Name cannot be changed once the group or role is created.
Description	A short description of the group or role.
Directory	The directory to which the group or role will be added. Note that the group or role cannot be moved to a different directory after it is created.
Active	Only deselect this if you wish to deny access to all members of the group or role.

Screenshot 1: 'Group Browser'



Screenshot 2: 'Add Group'



⚠ Groups (not roles) can also be added via Crowd's migration tools — see Importing Users and Groups into a Directory.

See Also

Specifying which Groups can access an Application

RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd
  - Adding a Sub-Group
  - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Deleting or Deactivating a Group

Deactivating a group prevents its members from logging in to any applications that use the Crowd framework. Deleting a group removes it completely from the relevant directory.

To deactivate a group,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' tab in the top navigation bar.
3. This will display the Group Browser. Select the relevant directory, locate the group you wish to deactivate, and click the '**View**' link that corresponds to the group.
4. This will display the '**Group Details**' screen. Deselect the '**Active**' check-box, then click the '**Update**' button.

To delete a group,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' tab in the top navigation bar.
3. This will display the Group Browser. Select the relevant directory, locate the group you wish to deactivate, and click the '**View**' link that corresponds to the group.
4. This will display the '**Group Details**' screen. Click '**Remove Group**' in the left-hand menu.

RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd

    - Adding a Sub-Group
    - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Managing Group Members

Groups are known as *permission container objects*. Groups are particularly important in Crowd, as they are often used to control access to applications. Note also that the 'crowd-administrators' group confers Crowd administration rights to its members.

This page tells you how to view the members of a group in Crowd. The list of group members may take a while to load, depending upon the size of your user base.

Other things you can do from the group browser:

- Add users to a group
- Remove users from a group
- Add sub-groups (nested groups)
- Remove sub-groups (nested groups)

About nested groups

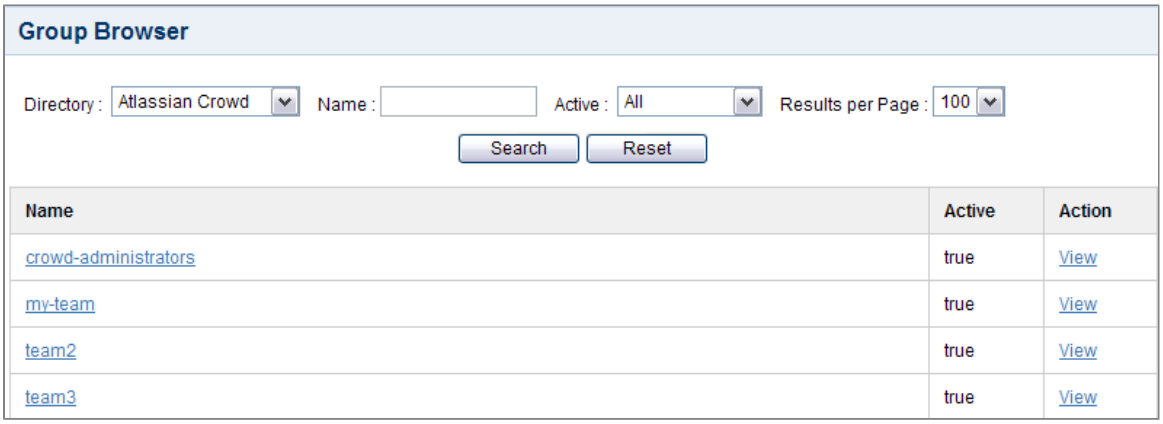
Some user directories allow you to define a group as a member of another group. Groups in such a structure are called '**nested groups**'. In Crowd, you can map any group to an application, including a group which contains other groups. Crowd supports nested groups for LDAP directory connectors, Crowd internal directories, Delegated Authentication directories and custom directories. You can **enable or disable** support for nested groups on each directory individually. For more information, refer to the documentation on configuring a directory.

For more details about nested groups, refer to Nested Groups in Crowd.

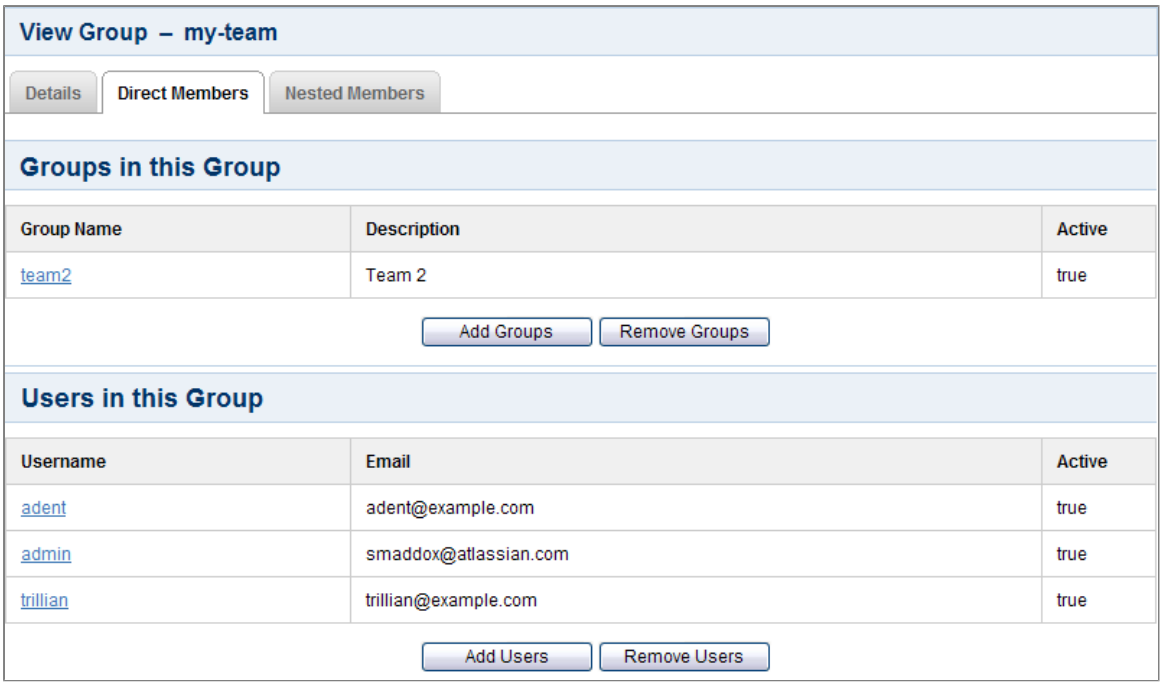
To view the members of a group,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' tab in the top navigation bar.
3. The Group Browser will appear, as shown in *screenshot 1* below. Select the directory in which you are interested, then click the '**Search**' button to list all the groups that exist in that directory. You can refine your search by specifying a '**Name**' or by choosing '**Active**' or '**Inactive**' groups.
4. Click the link on a specific group name to view the group's details.
5. The '**View Group — Details**' screen will appear. Click the '**Direct Members**' tab to view the immediate members of the group, as shown in *screenshot 2* below.
    - If your user directory allows nested groups, users and other groups may be members of the selected group. The 'Direct Members' tab shows all the immediate members of the group, including users and other groups.
    - If the group you are viewing does not contain other groups as members, the 'Direct Members' tab will show only users.
6. Click the '**Nested Members**' tab (if present) to view all users who are included in the group and in its sub-groups, as shown in *screenshot 3* below.

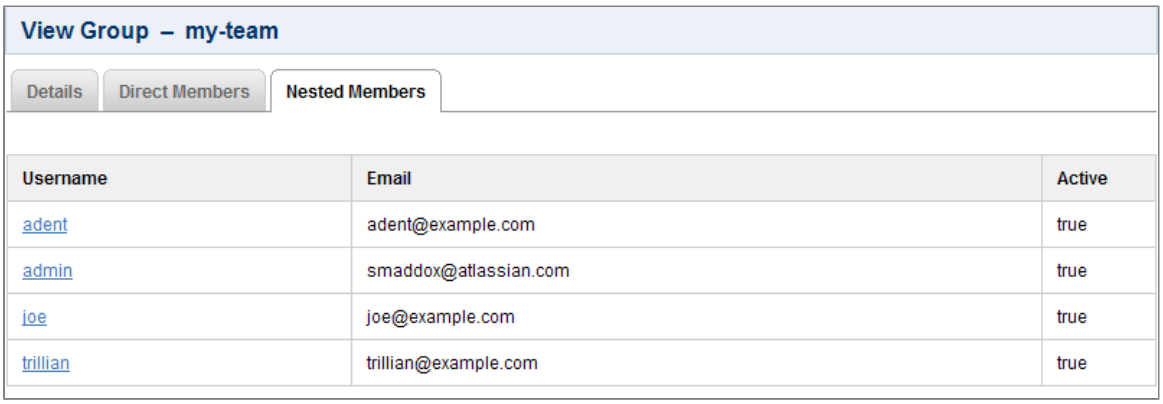
Screenshot 1: Group Browser



Screenshot 2: Viewing the direct members of a group



Screenshot 3: Viewing the nested users in a group



Adding users to groups and sub-groups
The 'Nested Members' tab does not allow you to add or remove members. To edit the membership of the group, please click the '**Direct Members**' tab. To edit the membership of a sub-group, click the '**Direct Members**' tab and then click the name of the sub-group to open the group maintenance screens for that group.

RELATED TOPICS

- Automatically Assigning New Users to Groups
- Adding Users to a Group
- Removing Users from a Group
- Nested Groups in Crowd
- Adding a Sub-Group
- Removing a Sub-Group

Managing Groups and Roles
Crowd Documentation

Automatically Assigning New Users to Groups

You can configure Crowd to assign new users to specific groups automatically. In summary:

- You can define default groups for each directory, as shown below.
- Every new user automatically becomes a member of these groups, whether the user is added via the Crowd Administration Console or via a Crowd-connected application.
- Note that the automatic group membership does not work when importing users and groups via Crowd's external user importer.

To add new default groups for a directory,

1. Log in to the Crowd Administration Console.
2. Click the '**Directories**' link in the top navigation bar.
3. The Directory Browser will appear. Search for the directory you wish to update, and click the link on the directory name.
4. The directory '**Details**' screen will appear. Click the '**Options**' tab.
5. The '**Options**' screen will appear, as shown below. Click the '**Add Groups**' button.
6. The '**Add Groups**' popup screen will appear, as shown below. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the group name. Leave the search box empty to match all group names.
7. You can refine your search by choosing '**Active**' or '**Inactive**' groups.
8. You can also set the '**Maximum Results**', i.e. the number of groups to be retrieved.
9. Click the '**Search**' button. Crowd will list the groups in the selected directory that match your search criteria, but excluding groups that are already defined as default groups for the selected directory.

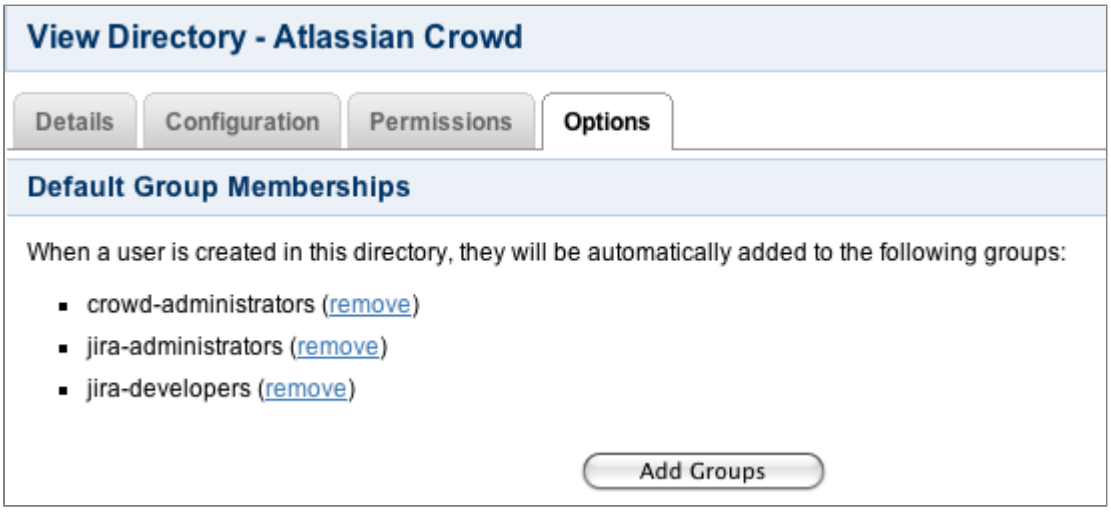
   Crowd will display a maximum number of groups as specified in the '**Maximum Results**' field. If too many groups match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching groups.)
10. Select the groups by putting a tick in the checkbox next to one or more group names. To select all groups, you can put a tick in the checkbox at the top of the table.
11. Click the '**Add Selected groups**' button to add the selected groups to the list of default groups for the directory.

To remove a group from the list of default groups for a directory,

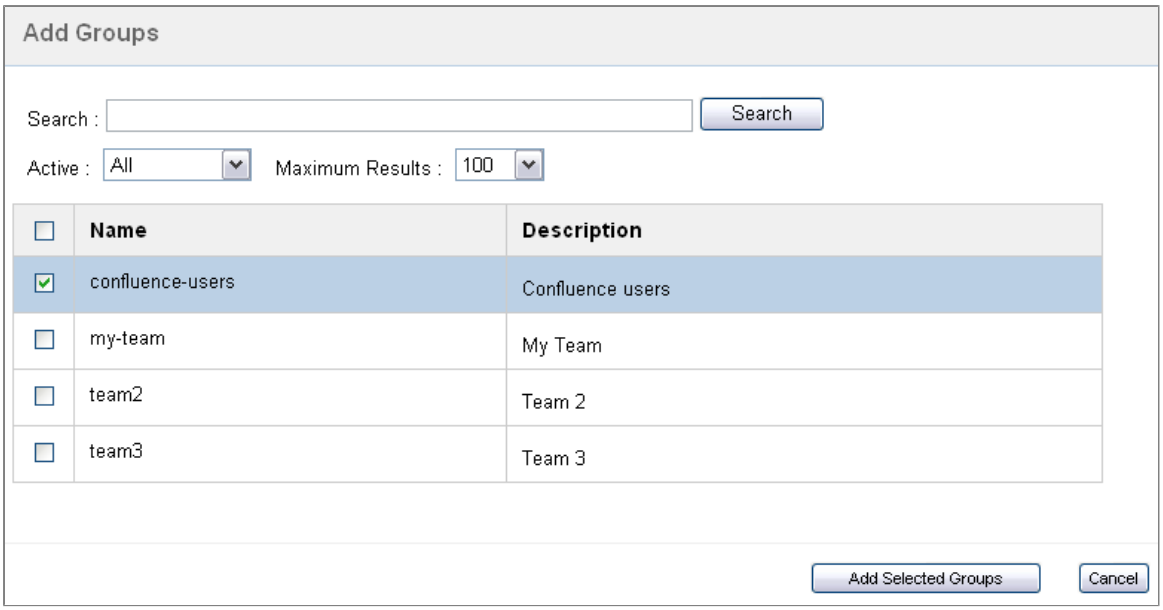
1. Find the group in the list on the '**Options**' tab.
2. Click the '**remove**' link next to the group name.

After you have removed the group from the list, new users will not be added automatically into the group. Existing users will remain members of the group.

Screenshot: Default groups for a directory



Screenshot: Popup for adding default groups



RELATED TOPICS

Managing Groups and Roles
Managing Group Members
Managing Directories
Crowd Documentation

Adding Users to a Group

When you add a user to a group, that user will be authorised to use any applications that use this group to control access.

You can add users to a group in two places:

- The group management screen for a specific group — Here you can add **many users at once** to the selected group.
- The user management screen for a specific user — Here you can add the selected user to **one or more groups** at a time.

Both methods are described below.

On this page:

- Adding Users via Group Management
- Adding Users via User Management
- Same Username in Multiple Directories

Adding Users via Group Management

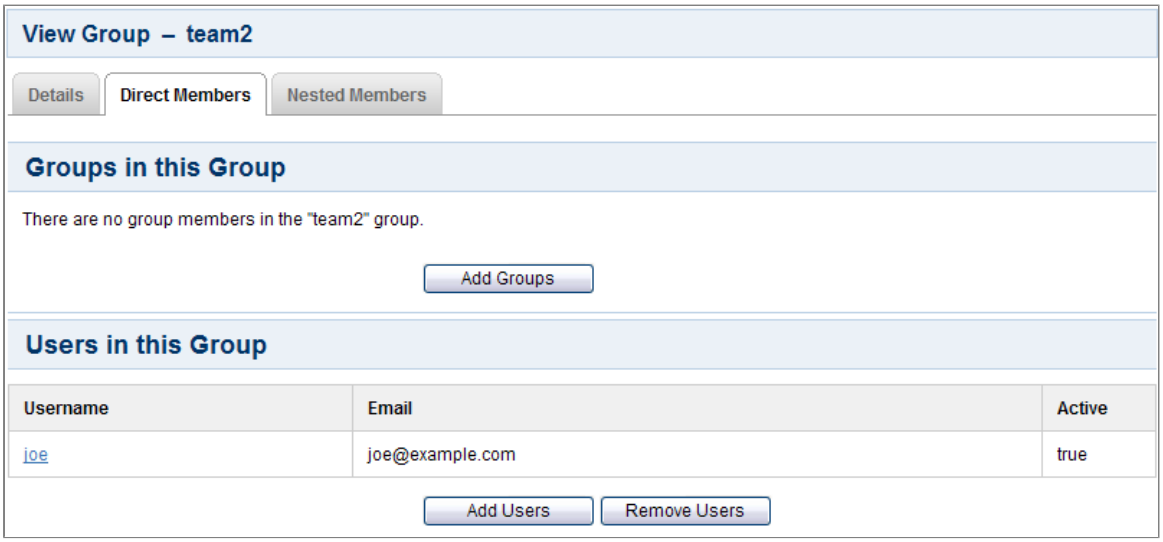
Using the group management screen for a specific group, you can add many users at once to the selected group.

To add one or more users to a group via the group management screen,

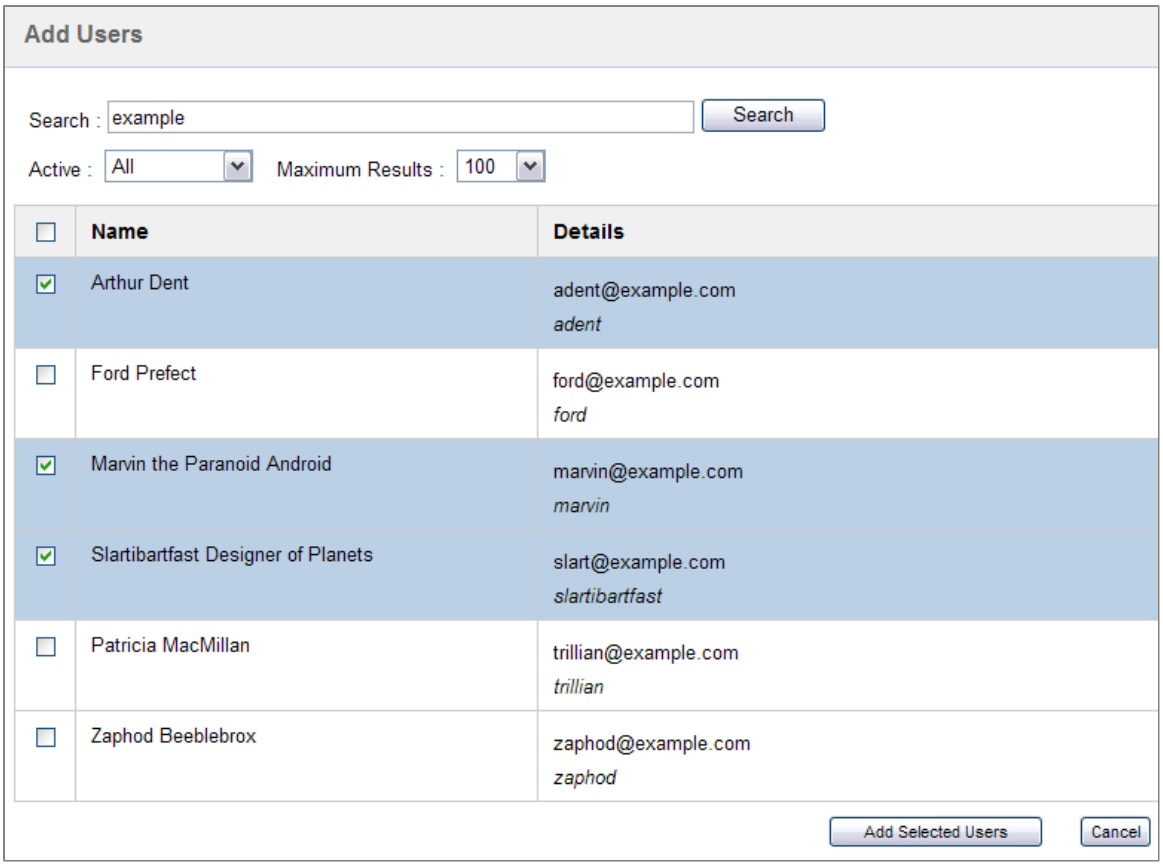
1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' link in the top navigation bar.
3. The Group Browser screen will appear. Select the relevant directory, locate the group you are interested in, and click the link on the group name.
4. The '**Group Details**' screen will appear. Click the '**Direct Members**' tab.
5. This will display a list of the selected group's members, both the groups and the users that are direct members of the group. See the screenshot below. Click the '**Add Users**' button.
6. The '**Add Users**' popup screen will appear, as shown below. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the user's email address or username. Leave the search box empty to match all usernames and email addresses.
7. You can refine your search by choosing '**Active**' or '**Inactive**' users. (An 'Inactive' user is typically someone who has left your organisation.)
8. You can also set the '**Maximum Results**', i.e. the number of users to be retrieved.
9. Click the '**Search**' button. Crowd will list the users in the selected directory who match your search criteria, but excluding users who are already members of the selected group.

   Crowd will display a maximum number of users as specified in the '**Maximum Results**' field. If too many users match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching users.)
10. Select the users by putting a tick in the checkbox next to one or more users. To select all users, you can put a tick in the checkbox at the top of the table.
11. Click the '**Add Selected Users**' button to add the selected users to the group.

Screenshot: Direct members of a group



Screenshot: Popup for adding users to a group



Adding Users via User Management

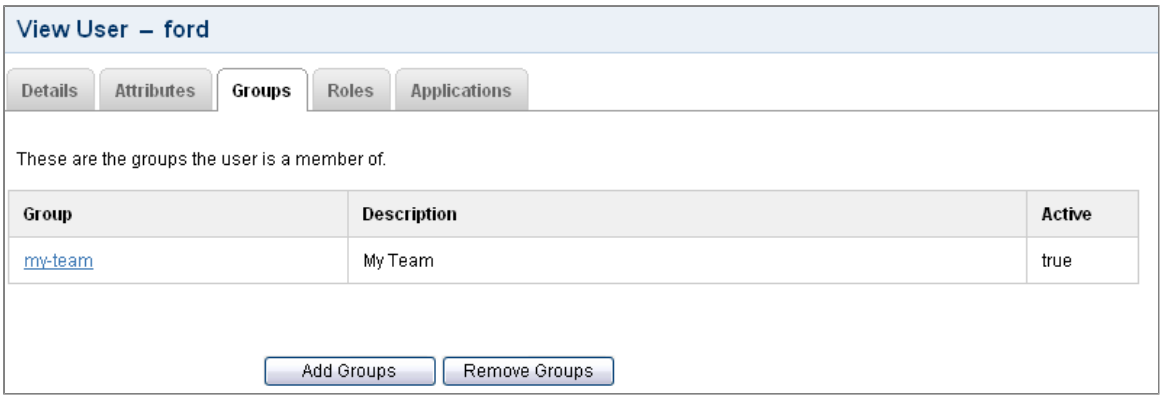
Using the user management screen for a specific user, you can add the selected user to one or more groups at a time.

To add a user to a one or more groups,

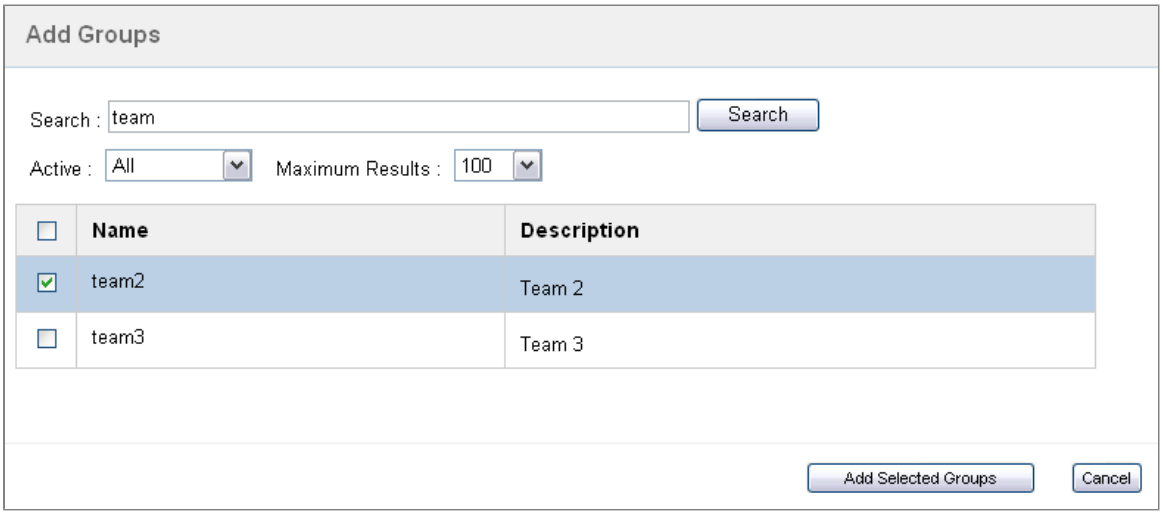
1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. The User Browser will appear. Select the relevant directory, locate the user you wish to add, and click the link on the user's name.
4. The '**User Details**' screen will appear. Click the '**Groups**' tab.
5. A list of the user's current groups (if any) will appear, , as shown below. Click the '**Add Groups**' button.
6. The '**Add Groups**' popup screen will appear, as shown below. Enter all or part of the group name in the '**Search**' textbox. Leave the search box empty to match all groups.
7. You can refine your search by choosing '**Active**' or '**Inactive**' groups.
8. You can also set the '**Maximum Results**', i.e. the number of groups to be retrieved.
9. Click the '**Search**' button. Crowd will list the groups in the selected directory that match your search criteria, but excluding groups that the user already belongs to.

   Crowd will display a maximum number of groups as specified in the '**Maximum Results**' field. If too many groups match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching groups.)
10. Select the groups by putting a tick in the checkbox next to one or more groups. To select all groups, you can put a tick in the checkbox at the top of the table.
11. Click the '**Add Selected groups**' button to add the user to the selected groups.

Screenshot: The groups that a user belongs to



Screenshot: Popup for adding a user to one or more groups



Same Username in Multiple Directories

If the same username exists in more than one directory assigned to an application, Crowd treats these usernames as the same user. Crowd searches all the assigned directories for the user, and amalgamates the group and role memberships.

For example, let's assume you have a user 'jsmith' who exists in both directories 'Customers' and 'Partners', and is a member of group 'G1' in 'Customers' and 'G2' in 'Partners'. Crowd will grant access to the user based on membership of both 'G1' and 'G2'.
RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
    - Adding a Group or Role
    - Deleting or Deactivating a Group
- Managing Group Members
    - Automatically Assigning New Users to Groups
    - Adding Users to a Group
    - Removing Users from a Group
    - Nested Groups in Crowd
    - Adding a Sub-Group
    - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Removing Users from a Group

If you remove a user from a group, the user will no longer be able to log in to any applications that use this group to control access.

Removing a user from a group does not delete the user from the directory. See Deleting or Deactivating a User.

You can remove users from a group in two places:

- The group management screen for a specific group — Here you can remove **many users at once** from the selected group.
- The user management screen for a specific user — Here you can remove the selected user from one or more groups at a time.

Both methods are described below.

On this page:

- Removing Users via Group Management
- Removing Users via User Management

Removing Users via Group Management

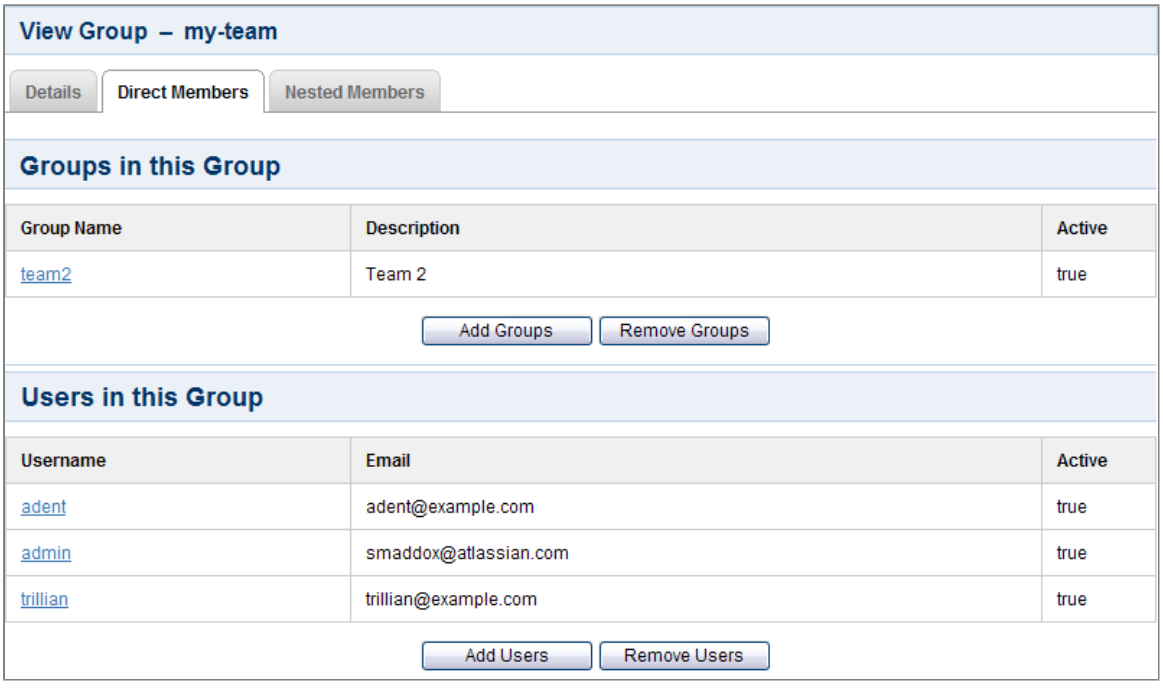
Using the group management screen for a specific group, you can remove **many users at once** from the selected group.

To remove one or more users from a group via the group management screen,

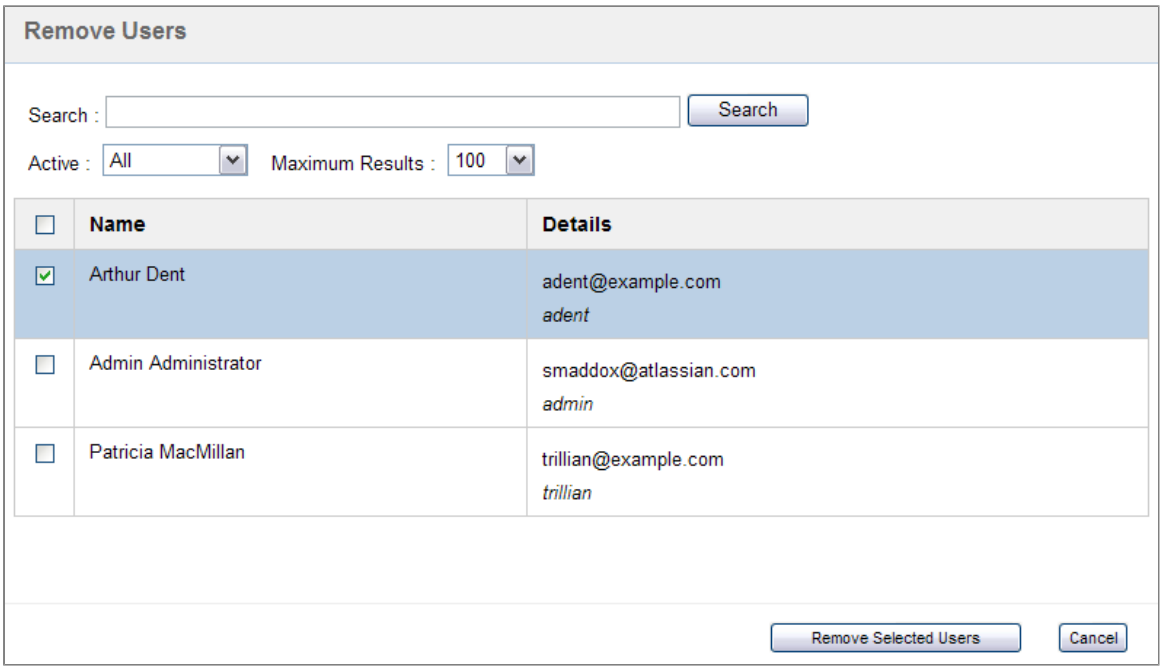
1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' link in the top navigation bar.
3. The Group Browser screen will appear. Select the relevant directory, locate the group you are interested in, and click the link on the group name.
4. The '**Group Details**' screen will appear. Click the '**Direct Members**' tab.
5. This will display a list of the selected group's members, both the groups and the users that are direct members of the group. See the screenshot below. Click the '**Remove Users**' button.
6. The '**Remove Users**' popup screen will appear, as shown below. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the user's email address or username. Leave the search box empty to match all usernames and email addresses.
7. You can refine your search by choosing '**Active**' or '**Inactive**' users. (An 'Inactive' user is typically someone who has left your organisation.)
8. You can also set the '**Maximum Results**', i.e. the number of users to be retrieved.
9. Click the '**Search**' button. Crowd will list the users in the selected directory who match your search criteria and are members of the selected group.

   Crowd will display a maximum number of users as specified in the '**Maximum Results**' field. If too many users match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching users.)
10. Select the users by putting a tick in the checkbox next to one or more names. To select all users, you can put a tick in the checkbox at the top of the table.
11. Click the '**Remove Selected Users**' button to remove the selected users from the group.

Screenshot: Direct members of a group



Screenshot: Popup for removing users from a group



Removing Users via User Management

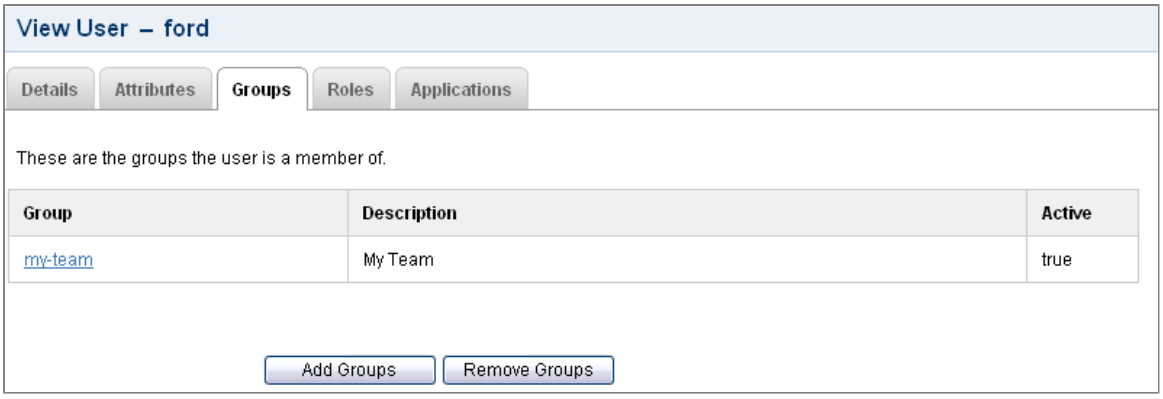
Using the user management screen, you can remove a specific user from the groups that that user belongs to.

To remove a user from one or more groups,

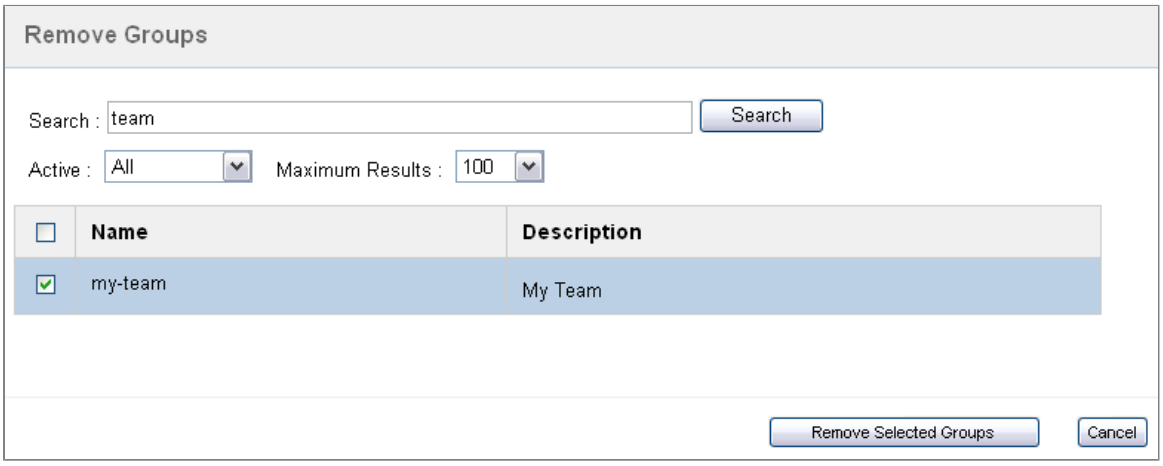
1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, locate the user you wish to remove, and click the link on the user's name.
4. This will display the '**User Details**' screen. Click the '**Groups**' tab.
5. A list of the user's current groups (if any) will appear, as shown below. Click the '**Remove Groups**' button.
6. The '**Remove Groups**' popup screen will appear, as shown below. Enter all or part of the group name in the '**Search**' textbox. Leave the search box empty to match all groups.
7. You can refine your search by choosing '**Active**' or '**Inactive**' groups.
8. You can also set the '**Maximum Results**', i.e. the number of groups to be retrieved.
9. Click the '**Search**' button. Crowd will list the groups that the user belongs to, matching your search criteria in the selected directory.

   Crowd will display a maximum number of groups as specified in the '**Maximum Results**' field. If too many groups match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching groups.)
10. Select the groups by putting a tick in the checkbox next to one or more groups. To select all groups, you can put a tick in the checkbox at the top of the table.
11. Click the '**Remove Selected groups**' button to remove the user from the selected groups.

Screenshot: The groups that a user belongs to



Screenshot: Popup for removing a user from one or more groups



RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
  - Adding a Group or Role
  - Deleting or Deactivating a Group
- Managing Group Members
  - Automatically Assigning New Users to Groups
  - Adding Users to a Group
  - Removing Users from a Group
  - Nested Groups in Crowd
  - Adding a Sub-Group
  - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

Nested Groups in Crowd

This page describes the way Crowd handles **nested groups**, i.e. groups which contain other groups as members and groups which are members of other groups.

On this page:

- Summary of Nested Groups in Crowd
- Definition of Nested Groups
- Supported Directory Types
- Group Management via the Crowd Administration Console
- Verifying a User's Access to an Application
- Presenting Flattened Lists of Users to Integrated Applications
- User Management via Integrated Applications
- Further Notes on Crowd's Processing

Summary of Nested Groups in Crowd

Some user directories allow you to define a group as a member of another group. Groups in such a structure are called '**nested groups**'. In Crowd, you can map any group to an application, including a group which contains other groups. Crowd supports nested groups for LDAP directory connectors, Crowd internal directories, Delegated Authentication directories and custom directories. You can **enable or disable** support for nested groups on each directory individually. For more information, refer to the documentation on configuring a directory.

Here's the effect on authorisation and presentation of group members to integrated applications:

- When verifying a user's login to an integrated application, Crowd will search the mapped group plus all its sub-groups.
- When an integrated application requests a list of users, Crowd will present a flat list of users gathered from the requested group and its sub-groups.

The rest of this page describes the above functionality in more detail.

In addition, you can follow the instructions to:

- Add a sub-group (nested group)
- Remove a sub-group (nested group)

Definition of Nested Groups

A 'nested group' is a group which is a member of another group. If you are using groups to manage permissions, you can create nested groups to allow inheritance of permissions from one group to its sub-groups.

In an LDAP directory, a nested group is defined as a child group entry whose DN (Distinguished Name) is referenced by an attribute contained within a parent group entry.

For example, a parent group '**Group One**' might have an `objectClass=group` attribute and one or more `member=DN` attributes, where the DN can be that of a user *or* that of a group elsewhere in the LDAP tree:

```
1.member=CN=John Smith,OU=Users,OU=OrgUnitA,DC=sub,DC=domain
2.member=CN=Group Two,OU=OrgUnitBGroups,OU=OrgUnitB,DC=sub,DC=domain
```

Supported Directory Types

Crowd supports nested groups for the following directory types:

- LDAP directory connectors
- Internal directories
- Delegated Authentication directories
- Custom directories, provided that the customisation meets the interface requirements of the `RemoteDirectory` API.

The directory importer does ***not*** support nested groups when importing users, groups and roles from LDAP into a delegated authentication directory. See CWD-1334.

Group Management via the Crowd Administration Console

The Crowd administrator can view group memberships, add a group as a member of another group, and remove a group's membership of another group.

Verifying a User's Access to an Application

When verifying a user's login to an integrated application, Crowd will search the groups mapped to the application, plus all their sub-groups. If the username exists in one of the groups, Crowd will allow the user access to the application.

Presenting Flattened Lists of Users to Integrated Applications

Integrated applications may ask Crowd for a list of members in a group. Crowd will present all users who are members of the group and all users belonging its sub-groups, consolidated into one list. We call this list a 'flattened' group. This is necessary because many integrated applications do not understand the concept of nested groups. For that reason, Crowd makes the nesting transparent to integrated applications.

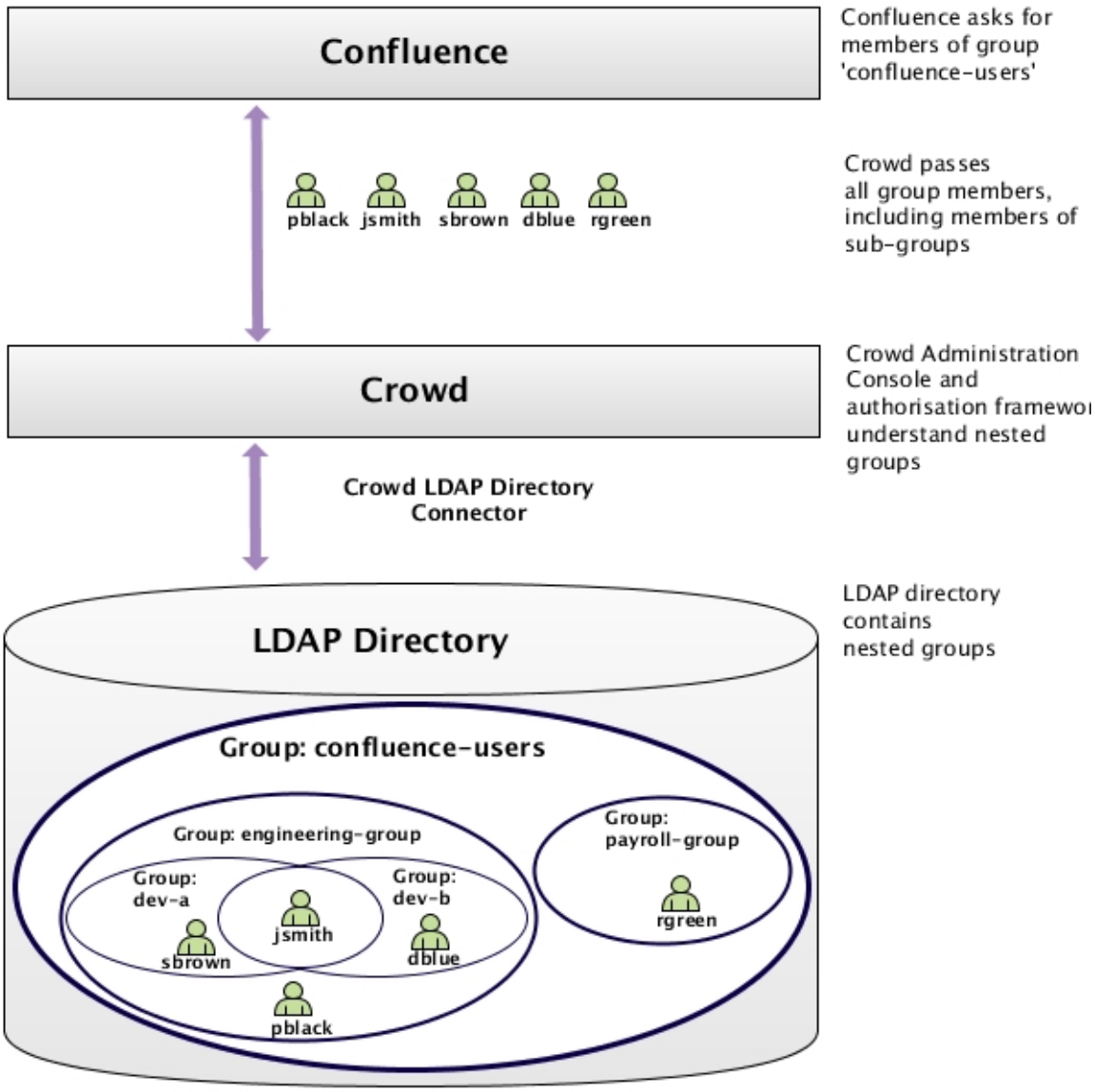
Use Case: Confluence Requests a List of Users in 'confluence-users' group

A Crowd-integrated Confluence instance will see users in sub-groups as members of the parent group, allowing administrators to use nested groups to manage permissions. (This will not affect Confluence instances that are not Crowd-enabled.)

For example:

- In LDAP we have groups '**engineering-group**' and '**payroll-group**'. We want to grant both groups access to our Confluence site.
    1. Using Crowd, we add a group called '**confluence-users**' in the LDAP directory.
    2. Add the '**engineering-group**' as a sub-group of '**confluence-users**'.
    3. Add the '**payroll-group**' as a sub-group of '**confluence-users**'.
- Group memberships are now:
    - **confluence-users** — sub-groups: **engineering-group**, **payroll-group**
    - **engineering-group** — sub-groups: **dev-a**, **dev-b**; users: **pblack**
    - **dev-a** — users: **jsmith**, **sbrown**
    - **dev-b** — users: **jsmith**, **dblue**
    - **payroll-group** — users: **rgreen**
- When Confluence requests a list of users in the '**confluence-users**' group, Crowd will present the following list:
    - **pblack**
    - **jsmith**
    - **sbrown**
    - **dblue**
    - **rgreen**

Diagram: Presenting Flattened Lists of Users to Integrated Applications



User Management via Integrated Applications

> **Recommendation: Enable External User Management**
> If you have JIRA, Confluence, Bamboo, FishEye or Crucible connected to Crowd, and you have nested groups in your directory, we recommend that you turn **on** external user management, via the administration screen of the integrated application. This will avoid confusion in the user-management screens of the integrated application, since these applications do not understand the concept of nested groups.

Use Case: Application Adds a User to a Group

If an integrated application adds a user to a flattened group, the user is added to the named group and not to any of its sub-groups.

Use Case: Application Removes a User from a Group

If an integrated application attempts to remove a user from a flattened group, Crowd will do the following:

- If the user is a member of the top group in the hierarchy (tree) of groups contained in the flattened list (e.g. `confluence-users`), Crowd will remove the user.
- Otherwise, Crowd will return an error stating that the user is not a direct member of the group.

Further Notes on Crowd's Processing

- Crowd handles circular/cyclical references — For example, '**group1**' is a member of '**group2**', '**group2**' is a member of '**group3**', and '**group3**' is in turn a member of '**group1**'.
- Crowd ignores members which are not users or groups — Group members might be computers, printers, etc.
- Crowd gracefully handles unreachable groups — There may be references to groups or members that Crowd cannot enumerate. This might be because the referenced group no longer exists, or the LDAP group structure is not entirely consistent. Crowd will ignore such groups and print a warning to the log file.

RELATED TOPICS

Managing Groups and Roles
Adding a Group or Role
Managing Group Members
Adding a Sub-Group
Removing a Sub-Group
Crowd Documentation

Adding a Sub-Group

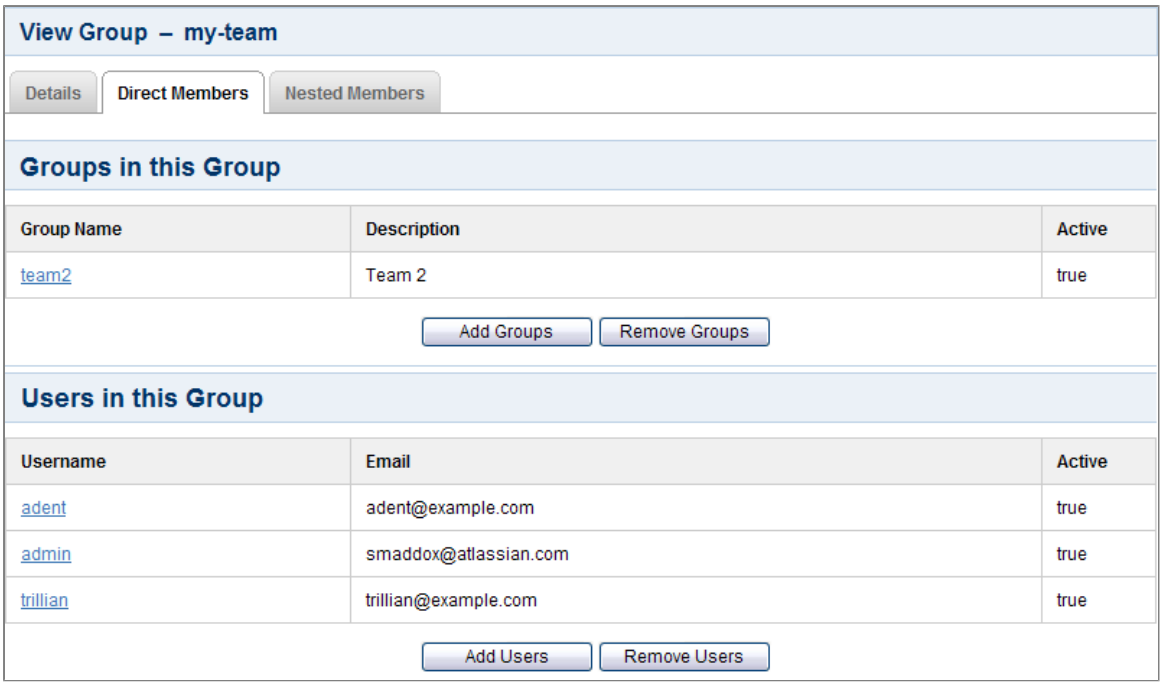
If your directory supports nested groups, you can add a group as a member of another group. This page tells you how to add such a sub-group.

> **About nested groups**
>
> Some user directories allow you to define a group as a member of another group. Groups in such a structure are called '**nested groups**'. In Crowd, you can map any group to an application, including a group which contains other groups. Crowd supports nested groups for LDAP directory connectors, Crowd internal directories, Delegated Authentication directories and custom directories. You can **enable or disable** support for nested groups on each directory individually. For more information, refer to the documentation on configuring a directory.
>
> For more details about nested groups, refer to Nested Groups in Crowd.

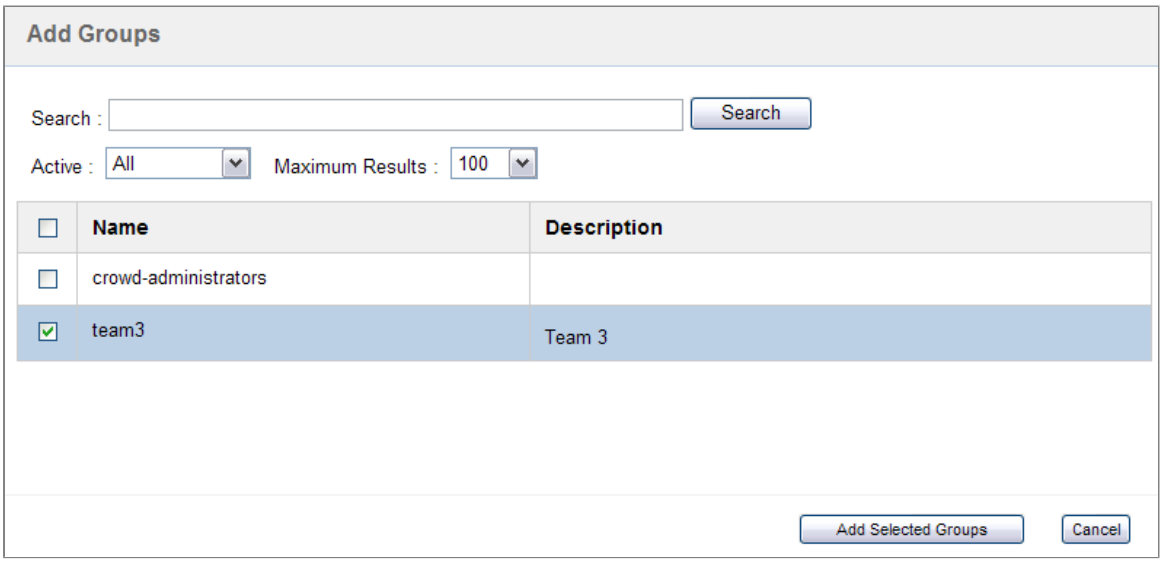
To add a sub-group,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' tab in the top navigation bar.
3. The Group Browser will appear. Select the directory in which you are interested, then click the '**Search**' button to list all the groups that exist in that directory. You can refine your search by specifying a '**Name**' or by choosing '**Active**' or '**Inactive**' groups.
4. If the sub-group does not yet exist in the directory, add it now:
   - Click '**Add Group**' in the left-hand menu.
   - Complete the fields as described in Adding a Group or Role, then click the '**Create**' button.
5. Now, you need to edit the parent group which will contain the sub-group:
   - If the parent group does not yet exist, add it now.
   - If the parent group already exists, find it in the list of groups and click the link on the group name to view the group details.
6. The '**View Group — Details**' screen will appear. Click the '**Direct Members**' tab.
7. This will display a list of the selected group's members, both the groups and the users that are direct members of the group. See the screenshot below. Click the '**Add Groups**' button.
   The 'Add Groups' button will not appear if nested groups are not enabled for your directory. You can enable nested groups via the directory configuration screen.
8. The '**Add Groups**' popup screen will appear, as shown below. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the group name. Leave the search box empty to match all group names.
9. You can refine your search by choosing '**Active**' or '**Inactive**' groups.
10. You can also set the '**Maximum Results**', i.e. the number of groups to be retrieved.
11. Click the '**Search**' button. Crowd will list the groups in the selected directory that match your search criteria, but excluding groups that are already sub-groups of the selected group.
    Crowd will display a maximum number of groups as specified in the '**Maximum Results**' field. If too many groups match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching groups.)
12. Select the groups by putting a tick in the checkbox next to one or more group names. To select all groups, you can put a tick in the checkbox at the top of the table.
13. Click the '**Add Selected groups**' button to add the selected groups to the group.

Screenshot: Direct members of a group



Screenshot: Popup for adding sub-groups



RELATED TOPICS

Nested Groups in Crowd
Managing Groups and Roles
Adding a Group or Role
Crowd Documentation

Removing a Sub-Group

If your directory supports nested groups, the directory may contain groups which are members of other groups. This page tells you how to remove a group's membership of another group. Note that removing a sub-group does **not** delete the group.

About nested groups

Some user directories allow you to define a group as a member of another group. Groups in such a structure are called '**nested groups**'. In Crowd, you can map any group to an application, including a group which contains other groups. Crowd supports nested groups for LDAP directory connectors, Crowd internal directories, Delegated Authentication directories and custom directories. You can **enable or disable** support for nested groups on each directory individually. For more information, refer to the documentation on configuring a directory.

For more details about nested groups, refer to Nested Groups in Crowd.

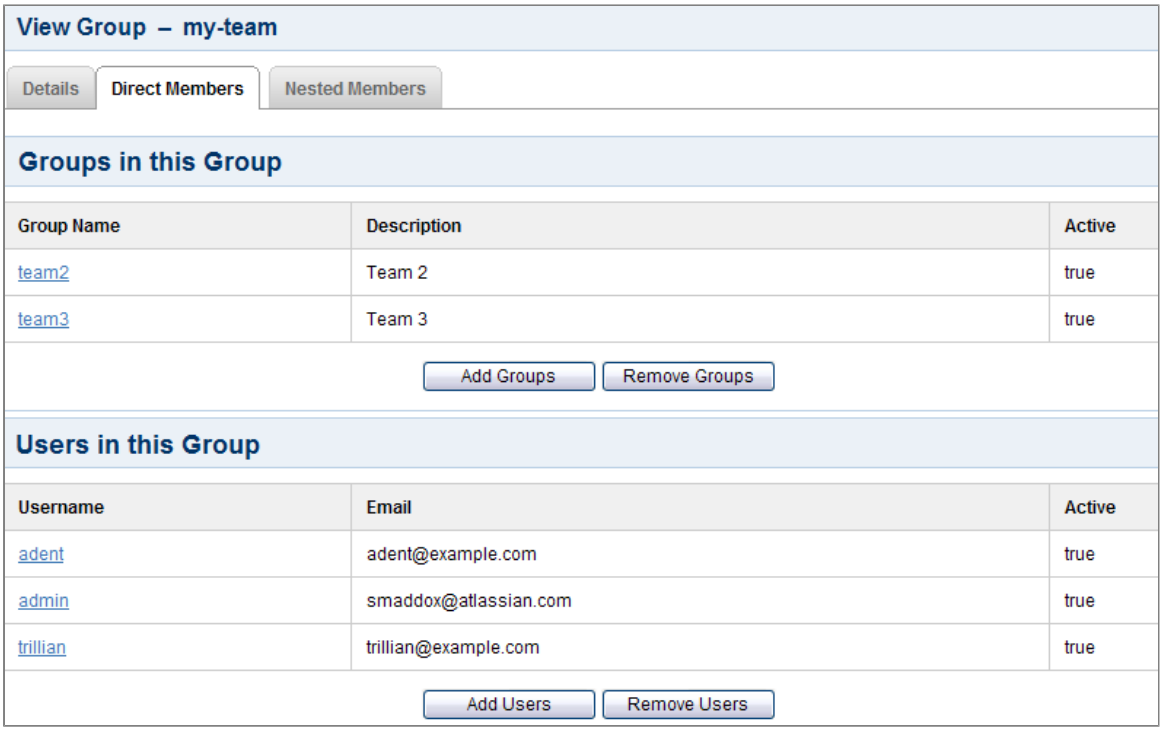
To remove a sub-group,

1. Log in to the Crowd Administration Console.
2. Click the '**Groups**' tab in the top navigation bar.
3. The Group Browser will appear. Select the directory in which you are interested, then click the '**Search**' button to list all the groups that exist in that directory. You can refine your search by specifying a '**Name**' or by choosing '**Active**' or '**Inactive**' groups.
4. Find the parent group in the list of groups and click the link on the group name to view the group details.
5. The '**View Group — Details**' screen will appear. Click the '**Direct Members**' tab.
6. This will display a list of the selected group's members, both the groups and the users that are direct members of the group. See the screenshot below. Click the '**Remove Groups**' button.

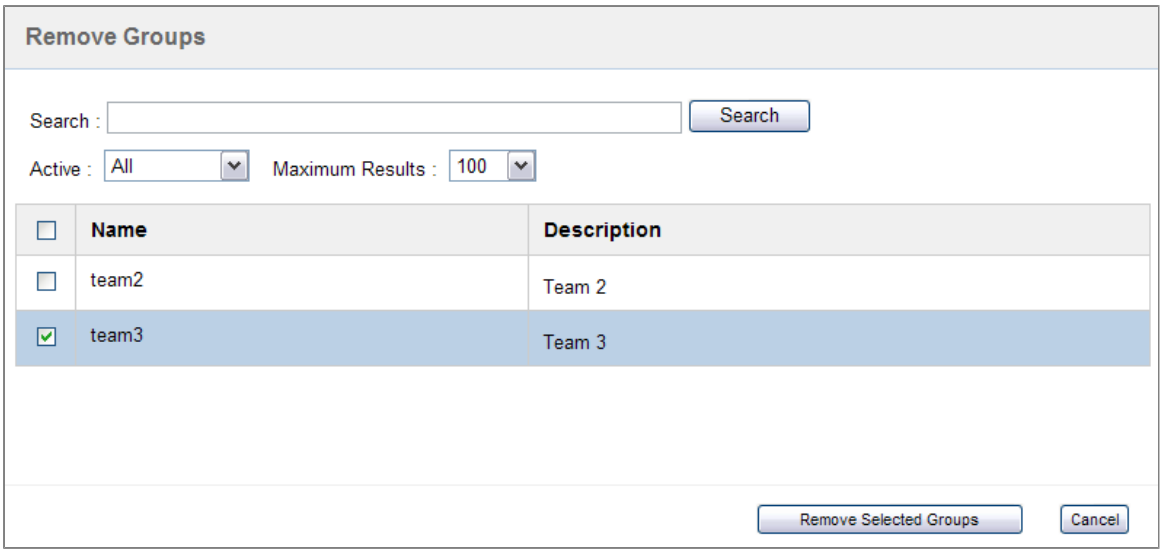
   The 'Remove Groups' button will not appear if nested groups are not enabled for your directory. You can enable nested groups via the directory configuration screen.
7. The '**Remove Groups**' popup screen will appear, as shown below. Enter your search criteria in the '**Search**' textbox. You can enter all or part of the group name. Leave the search box empty to match all group names.
8. You can refine your search by choosing '**Active**' or '**Inactive**' groups.
9. You can also set the '**Maximum Results**', i.e. the number of groups to be retrieved.
10. Click the '**Search**' button. Crowd will list the groups in the selected directory that match your search criteria and are sub-groups of the selected group.

    Crowd will display a maximum number of groups as specified in the '**Maximum Results**' field. If too many groups match the search, you can change the search criteria and click 'Search' again. (There is no way to move to the next page of matching groups.)
11. Select the groups by putting a tick in the checkbox next to one or more group names. To select all groups, you can put a tick in the checkbox at the top of the table.
12. Click the '**Remove Selected Groups**' button to remove the selected sub-groups from the group.

Screenshot: Direct members of a group



Screenshot: Popup for removing sub-groups from a group



RELATED TOPICS

Nested Groups in Crowd
Adding a Sub-Group
Managing Groups and Roles
Crowd Documentation

Case Sensitivity of Usernames, Groups and Roles

This page summarises the way Crowd handles case sensitivity for usernames, group names and role names when storing, matching and searching data and when passing data between directories and applications.

Terminology:

- **Case insensitive** — Upper-case and lower-case letters are assumed to have the same meaning: `JSmith` is the same as `jsmith`.
- **Case preserving** — Upper and lower case are retained when passing or storing information: `JSmith` remains `JSmith`.

Outside Crowd

External to Crowd:

- Most LDAP directory schemas specify the user, group and role names as case insensitive for matching and searching, but case preserving when storing the data and passing it back to the requestor.
- Applications behave in different ways. Some, like JIRA and Confluence, insist on lower-case usernames, groups and roles and store all user-related data in lower case.

The Crowd Solution

Crowd's application caches and LDAP directory caches are case insensitive but case preserving. Crowd will ignore case when comparing usernames, etc (`JSmith` = `jsmith`) and it will preserve case when passing information between applications and directories (`JSmith` remains `JSmith`).

In addition, Crowd Internal and Delegated Authentication directories:

- Are case preserving, i.e. they store usernames, group and role names in mixed case.
- Support case-insensitive matching and searching.

Importing Users, Groups and Roles into Crowd Internal Directories

When you import user information into a Crowd Internal or Delegated Authentication directory, the case of usernames, group names and role names will be preserved.

Enforcing Lower-Case Usernames, Groups and Roles for an Application

In some cases you may wish to convert user, group and role names to lower case when passing them to an application. You can set an option for each application, as described in Enforcing Lower-Case Usernames, Groups and Roles for an Application. When the option is set, Crowd will convert upper-case and mixed-case information obtained from your user directory to lower case before passing the information to the application.

RELATED TOPICS

- Overview of Caching

- Managing Directories

Crowd Documentation

Editing a User's Group and Role Membership

Within any given directory, you can choose the groups and roles to which each user belongs. Note that a user's group membership is particularly important, as groups are often used to control access to applications.

Groups

The Crowd Administration Console provides two ways of adding users to or removing users from a group:

- The group management screen for a specific group — Here you can add many users at once to the selected group.
- The user management screen for a specific user — Here you can add the selected user to one or more groups at a time.

Full instructions are in Adding Users to a Group and Removing Users from a Group.

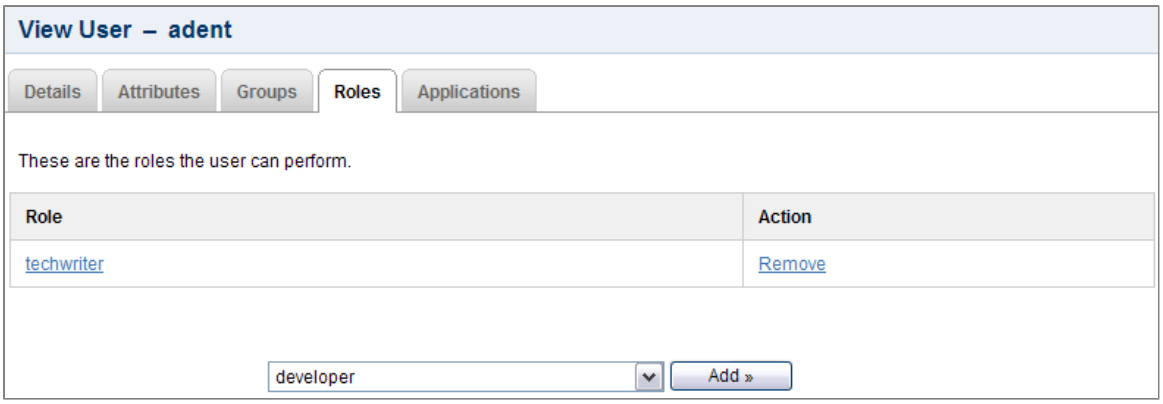
Roles

Crowd's role-based access control could be enhanced
At present, the implementation of roles in Crowd is identical to the implementation of groups. Additional development work would be needed to differentiate the functionality of roles from groups. If you would like to help us to design better role-based access control, please add a comment to the improvement request CWD-931, letting us know how you would like to see it work.

To add a user to a role,

1. Log in to the Crowd Administration Console.
2. Click the '**Users**' link in the top navigation bar.
3. This will display the User Browser. Select the relevant directory, locate the user you wish to add, and click the link on the user's name.
4. This will display the '**User Details**' screen. Click the '**Roles**' tab.
5. A list of the user's current roles (if any) will be displayed, as shown on the screenshot below. Select the relevant role from the drop-down box below the list, then click the '**Add**' button.

Screenshot: Managing a user's roles



Multiple Directories

If the same username exists in more than one directory assigned to an application, Crowd treats these usernames as the same user. Crowd searches all the assigned directories for the user, and amalgamates the group and role memberships.

For example, let's assume you have a user 'jsmith' who exists in both directories 'Customers' and 'Partners', and is a member of group 'G1' in 'Customers' and 'G2' in 'Partners'. Crowd will grant access to the user based on membership of both 'G1' and 'G2'.

RELATED TOPICS

- Using the User Browser
- Adding a User
- Deleting or Deactivating a User
- Managing a User's Session
- Editing a User's Details and Password
- Specifying a User's Attributes
- Specifying a User's Aliases

- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing Groups and Roles
    - Adding a Group or Role
    - Deleting or Deactivating a Group
- Managing Group Members
    - Automatically Assigning New Users to Groups
    - Adding Users to a Group
    - Removing Users from a Group
    - Nested Groups in Crowd
    - Adding a Sub-Group
    - Removing a Sub-Group
- Case Sensitivity of Usernames, Groups and Roles
- Editing a User's Group and Role Membership

Crowd Documentation

System Administration

- Configuring Server Settings
    - Deployment Title
    - Domain
    - Token Seed
    - Session Configuration
    - Authorisation Caching
    - Compression of Server Output
    - Licensing
    - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
    - Performance Profiling
- Overview of Caching

Configuring Server Settings

You can alter the settings which were specified when your Crowd server was installed:

- Deployment Title
- Domain
- Token Seed
- Session Configuration
- Authorisation Caching
- Compression of Server Output
- Licensing
- SSO Cookie

RELATED TOPICS

- Configuring Server Settings
    - Deployment Title
    - Domain
    - Token Seed
    - Session Configuration
    - Authorisation Caching
    - Compression of Server Output
    - Licensing
    - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
    - Performance Profiling
- Overview of Caching

Crowd Documentation

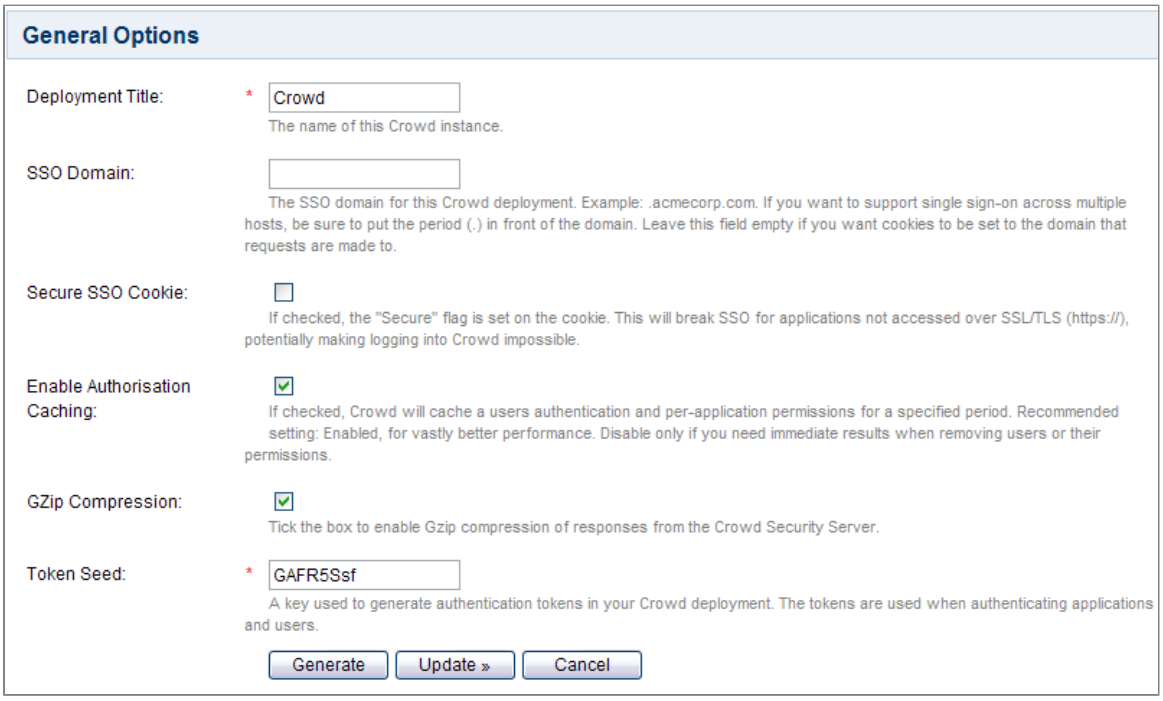
Deployment Title

The deployment title is a unique name for your Crowd instance. The deployment title is used by default in the subject line of email notifications.

To specify the deployment title,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. The '**General Options**' screen will appear. Type the new name into the '**Deployment Title**' field.
4. Click the '**Update**' button.

Screenshot: 'General Options'



RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Domain

The **SSO domain** is used when setting HTTP authentication cookies in a user's browser. If this attribute is not correct, single sign-on (SSO) will not work when the user switches between applications.

The core Crowd functionality supports SSO across applications within a single domain, such as `*.mydomain.com`. Crowd uses a browser cookie to manage SSO. Because your browser limits cookie access to hosts in the same domain, this means that all applications participating in SSO must be in the same domain.

Example 1: If you wish to have single sign-on (SSO) support for ***.mydomain.com**, you will need to configure the SSO domain in Crowd as `.mydomain.com` — including the full stop ('.') at the beginning. All your Crowd-connected applications must be in the same domain. For example:

Crowd	`crowd.mydomain.com`	✓
JIRA	`jira.mydomain.com`	✓
Confluence	`confluence.mydomain.com`	✓
FishEye	`fisheye.mydomain.com`	✓
FishEye in different domain	`fisheye.example.com`	✗

Example 2: If you wish to have single sign-on (SSO) support for **mydomain.com/***, you will need to configure the SSO domain in Crowd as `mydomain.com`. All your Crowd-connected applications must be in the same domain. For example:

Crowd	`mydomain.com/crowd`	✓
JIRA	`mydomain.com/jira`	✓
Confluence	`mydomain.com/confluence`	✓
FishEye	`mydomain.com/fisheye`	✓
FishEye in different domain	`example.com/fisheye`	✗

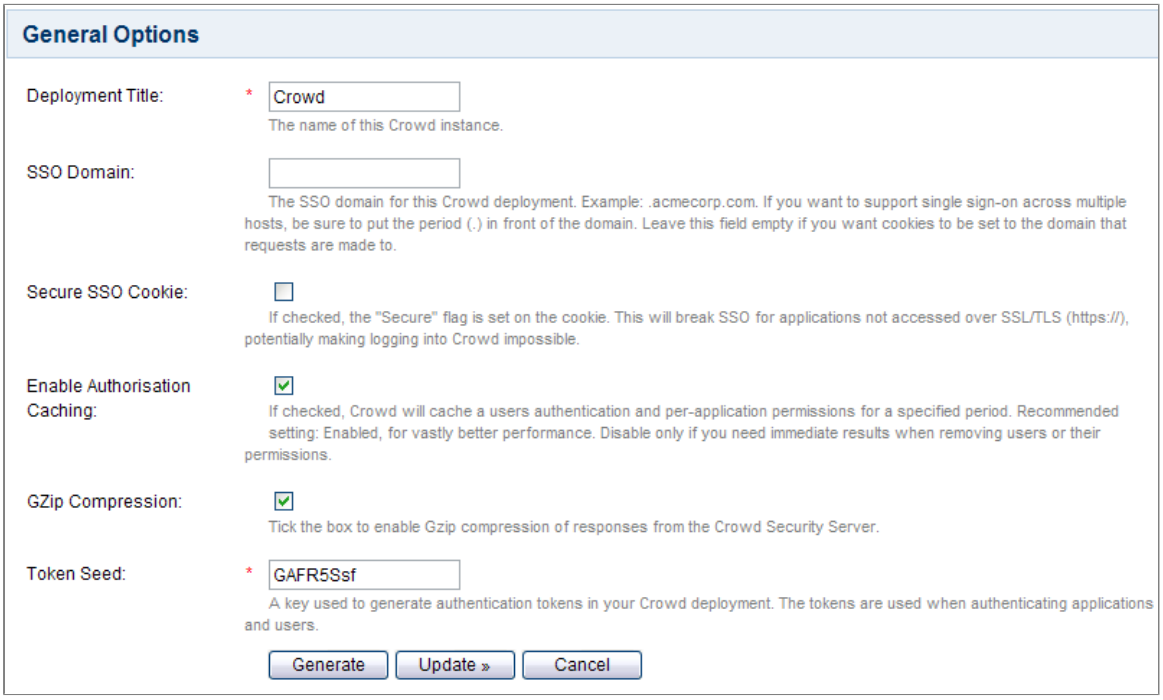
You can find information the comparison of host name strings in RFC 2965 (pages 2 and 3).

When developing on your local machine, you should set the domain to `localhost`.

To specify the domain,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. The '**General Options**' screen will appear. Type the new domain into the '**SSO Domain**' field.
4. Click the '**Update**' button.

Screenshot: 'General Options'



Setting the SSO Domain when Crowd is behind a Proxy Server

If Crowd is being run behind a proxy server, before setting the SSO domain value, make sure that the domain specified in the proxy (that is currently being used to access the Crowd console) was specified in the Tomcat connector **proxyName** attribute. Example:

File: `Apache-Tomcat/conf/server.xml`

```
<Connector acceptCount="100" connectionTimeout="20000" disableUploadTimeout="true" enableLookups="false"
maxHttpHeaderSize="8192" maxSpareThreads="75" maxThreads="150" minSpareThreads="25"
port="8095" redirectPort="8443" useBodyEncodingForURI="true"
proxyName="mycompany.com" />
```

RELATED TOPICS

Overview of SSO
Configuring Trusted Proxy Servers

Crowd Documentation

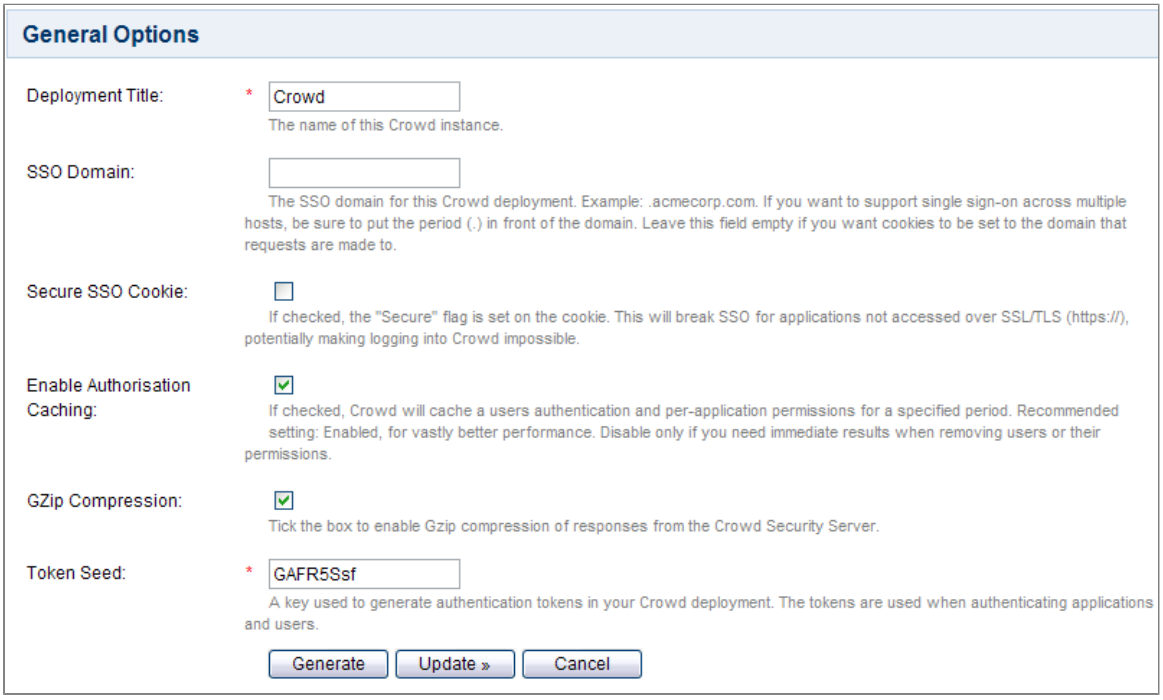
Token Seed

The token seed is a unique key for each site deployment of Crowd. This key is used when generating tokens for an authenticated application.

To specify the token seed,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. The '**General Options**' screen will appear. Now you can either:
   - Type the new key into the '**Token Seed**' field, then click the '**Update**' button.
     OR
   - Click the '**Generate**' button to create a random key automatically.

Screenshot: 'General Options'



RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data

- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Session Configuration

This page tells you how to set the timeout period for a session token and how to enable/disable in-memory token storage.

Session Timeout

When a successful authentication occurs, for either an application or a user, a unique token is assigned. Tokens are valid for the period of time specified as the 'Session Timeout' attribute.

The session timeout determines how long a session will be considered valid during any period of inactivity. This value is specified in minutes and must be greater than `0`.

To specify the session timeout,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Session Config**' in the left-hand menu.
4. The '**Session Config**' screen will appear, as shown below. Type the new value into the '**Session Timeout**' field, then click the '**Update**' button.

Authentication Token Storage

Authentication tokens are used to validate application and user sessions. A token is stored for each active session. By default, they're kept in the Crowd database. Storing these tokens in memory can benefit performance, but with some significant drawbacks:

- Sessions will not be saved across Crowd restarts. If you restart Crowd, all your users will have to log in again.
- Clustering will not be possible. Atlassian does not officially support clustering Crowd, but a number of our customers are successfully using it in this manner. See this knowledge-base article.

Switching from database to in-memory token management does not require a restart of Crowd; nor will sessions be lost or validations failed. However, if you have lots of active sessions, and therefore lots of tokens, it can take some time to copy the token information. During this time, validation requests will be queued and Crowd will appear unresponsive to client applications.

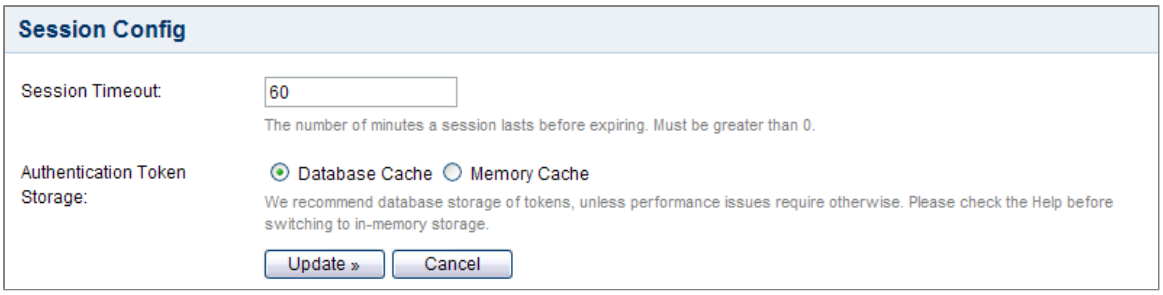
As a guide, below are some benchmarks of time taken to switch from one form of token storage to the other. The measurements were taken on a quad-core Mac Pro, using a lightly-loaded PostgreSQL database:

Number of Tokens:	100	500	1000	5000	10000
Database -> Memory	0.1s	0.7s	1.2s	4.2s	8.2s
Memory -> Database	1.2s	4.8s	9.2s	45s	90s

To switch the token storage location,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Session Config**' in the left-hand menu.
4. The '**Session Config**' screen will appear, as shown below. Select one of the radio buttons next to Authentication Token Storage:
   - '**Database Cache**' — This is the default option. Select it to store your tokens in the Crowd database. We recommend this option unless performance problems require in-memory storage.
   - '**Memory Cache**' — Select this option to store your tokens in memory.
5. Click the '**Update**' button.

Screenshot: 'Session Config'



RELATED TOPICS

- Managing an Application's Session
- Managing a User's Session

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Authorisation Caching

Caching is used to store run-time authentication and authorisation rules, which can be expensive to calculate.

This page describes the cache that can be configured on the Crowd server, to store users' authentication and per-application permissions for a specified period. For an overview of the other types of caching offered by Crowd, please refer to Overview of Caching.

Caching of Users' Application Permissions on the Crowd Server — The Authorisation Cache

Crowd can store users' authentication and per-application permissions in a local cache for a specified period after retrieving the information from the directory and application data. The cached data will answer the following questions:

- For a particular user: Is the user authenticated?
- For a particular user and application: Does the user have access to the application?

You might call this the 'has access' cache, or the '**authorisation cache**'.

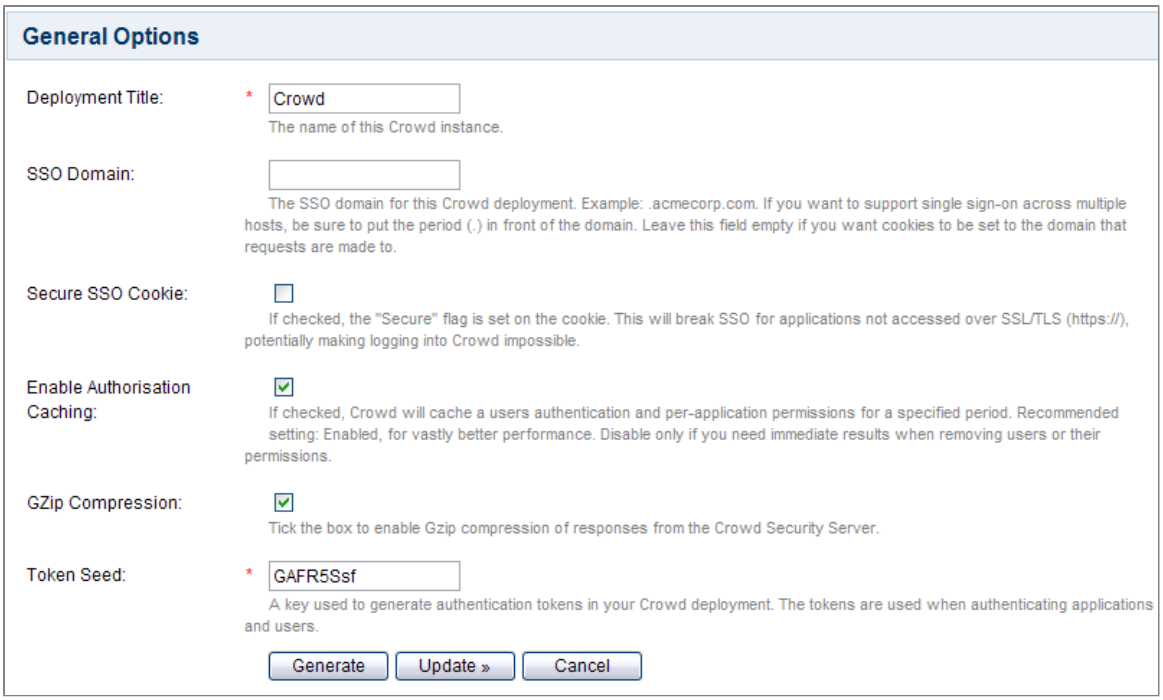
Recommended setting: **Enabled**. For performance reasons, we recommend that the cache be enabled on the Crowd server. This is the default setting.

The effect of caching the data is that users will retain access to applications for a period after their username or permission has been removed, i.e. until the server-side cache expires. You should disable the cache only if you need immediate results when removing users or their permissions.

To enable caching of user-to-application permissions on the Crowd server,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. The '**General Options**' screen will appear. Put a tick in the '**Enable Authorisation Caching**' checkbox.
4. Click the '**Update**' button.

Screenshot: 'Caching'



Some applications may enable/disable caching based on the Crowd server setting
The Crowd API allows an application to query whether caching is enabled on the Crowd server (`isCacheEnabled`). The Crowd Java client does not make use of this API feature, because it makes more sense to have application caching configured entirely on the application side. If you have a Crowd-integrated custom application which does make use of this API call, then the setting on the Crowd server will affect your application-side caching as well.

RELATED TOPICS

- Overview of Caching
- Configuring Caching for an LDAP Directory
- Configuring Caching for an Application

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Compression of Server Output

By default, Crowd compresses the output from the security server, using the Gzip compression format, before sending the data to the client over the network. Compression of server output is optional. You can turn it on or off via the Crowd Administration Console.

Here are some reasons why you may want to turn compression off:

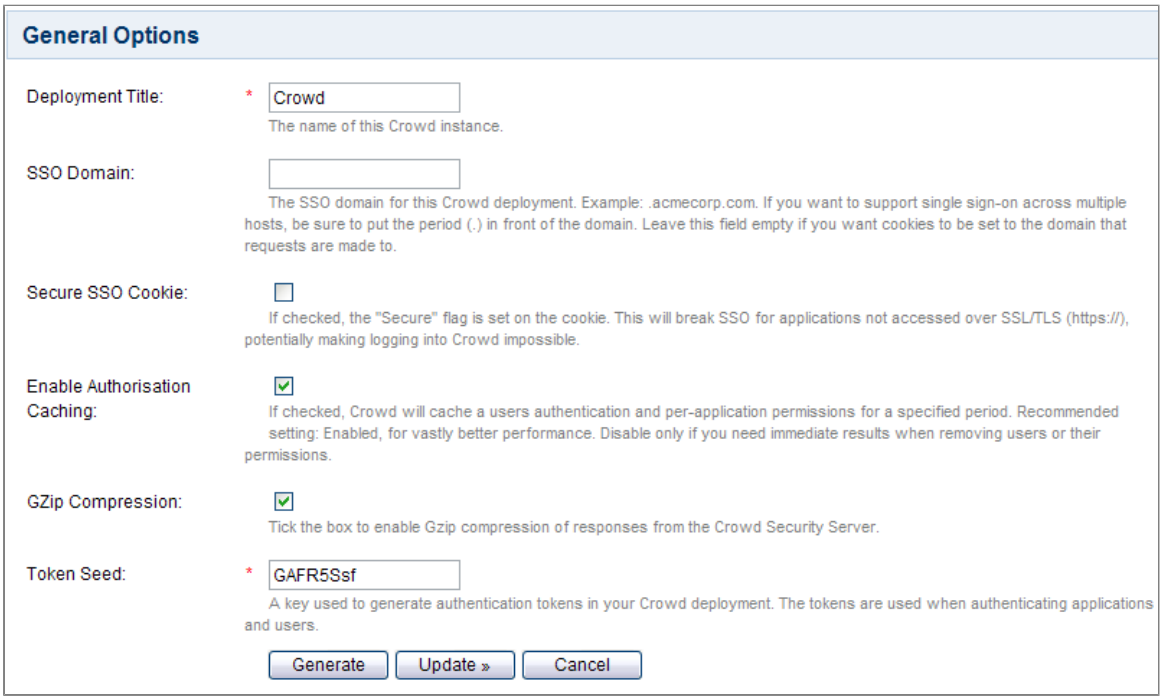
- It may be easier to debug problems using uncompressed data.
- Some agents, such as older versions of Internet Explorer, have problems with the Gzip format.

If you're proxying Crowd behind Apache, check to see if you're using mod_deflate. You do not need to enable Gzip compression if Apache already provides it or you may encounter this issue: CWD-1398.

To enable/disable compression of server output,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. The '**General Options**' screen will appear. Set the '**Gzip Compression**' option as follows:
    - Put a tick in the checkbox to instruct the Crowd Security Server to use Gzip compression when sending responses.
    - Leave the checkbox empty to instruct Crowd to send uncompressed data.

Screenshot: 'Setting the Compression of Server Output'



RELATED TOPICS

- Configuring Server Settings
    - Deployment Title
    - Domain
    - Token Seed
    - Session Configuration
    - Authorisation Caching
    - Compression of Server Output
    - Licensing
    - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
    - Performance Profiling
- Overview of Caching

Crowd Documentation

Licensing

Crowd licenses are based on the number of end-users who will log in to the applications that are integrated with Crowd.

You can obtain an evaluation license from the Atlassian website. When you obtain an evaluation license — or purchase, renew or upgrade your license — you will receive a license key via email or on the Atlassian website. You will need to enter your license key into your Crowd server as described below.

On this page:

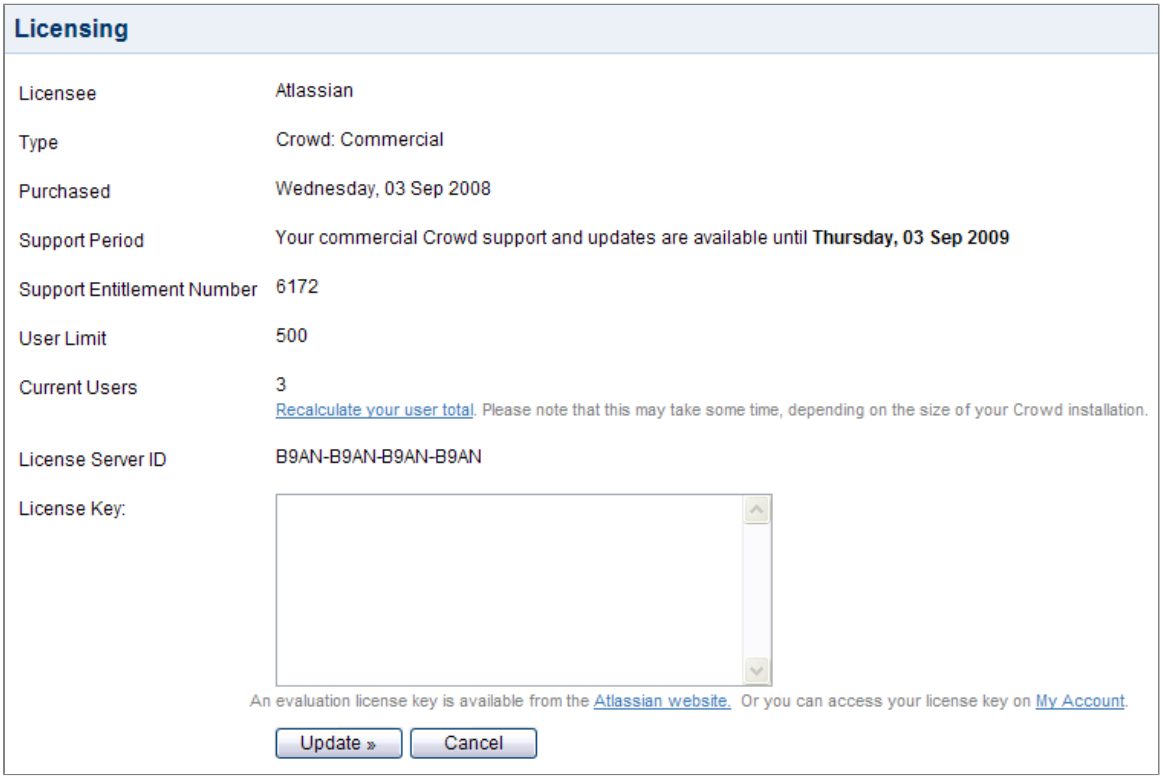
- Entering your License Key
- Warning when Number of Users approaches License Limit
- What to Do if the Number of Users Exceeds your License Limit
- Minimising your Licensing Cost
- Recalculating your User Total
- Server ID and Support Entitlement Number

Entering your License Key

To enter your license key,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Licensing**' in the left-hand menu.
4. Type (or paste) your license key into the '**License Key**' field.
5. Click the '**Update**' button.

Screenshot: 'Licensing'



Warning when Number of Users approaches License Limit

Whenever the number of users reaches 90% of the number allowed by the license, Crowd will send an email informing the administrator about the license limit and the current number of users. The email is sent to the email notification address, as defined on the '**Mail Configuration**' screen in the Crowd Administration Console. (See Configuring your Mail Server.)

This warning should help the administrator to take action and avoid exceeding the license limit.

What to Do if the Number of Users Exceeds your License Limit

If the number of users who are allowed to log in to the Crowd framework exceeds the user license limit, no-one will be able to log in to any applications (other than the Crowd Administration Console). If this happens, you can secure sufficient time to resolve this situation by contacting Atlassian to obtain a 30-day evaluation license. Thirty days should give you enough time to do one of the following:

- either buy a license for a higher user count.
- Or work out which users to remove, in order to bring the number of users under the user limit.

Minimising your Licensing Cost

If you have more than one directory, ensure that the same user does not exist in multiple directories.

We recommend that you allow only particular groups to log in to each application, rather than entire directories.

Note that a mapped application can 'see' all users in a directory, even if not all of them can log in to the application. For example, a Human Resources application might be mapped to your entire Active Directory server, but only the HR group is allowed to log in to the application.

Recalculating your User Total

The Licensing screen shows the number of users who currently count towards your license. This total is updated automatically at regular intervals. If you have recently added or removed users, the total may not be up to date when you view the screen. You can update the count immediately, as described below.

To recalculate your user total,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Licensing**' in the left-hand menu.
4. Click the link labelled '**Recalculate your user total**'.
   The recalculation may take a while, depending on the size of your user base.

Server ID and Support Entitlement Number

Your **License Server ID** is generated automatically, based on your license key.

The **Support Entitlement Number** will appear only on newer licenses. If your License Server ID starts with a 'B', you should also have a Support Entitlement Number. This number is not currently used, but will be used by Atlassian Support in the future.

RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

SSO Cookie

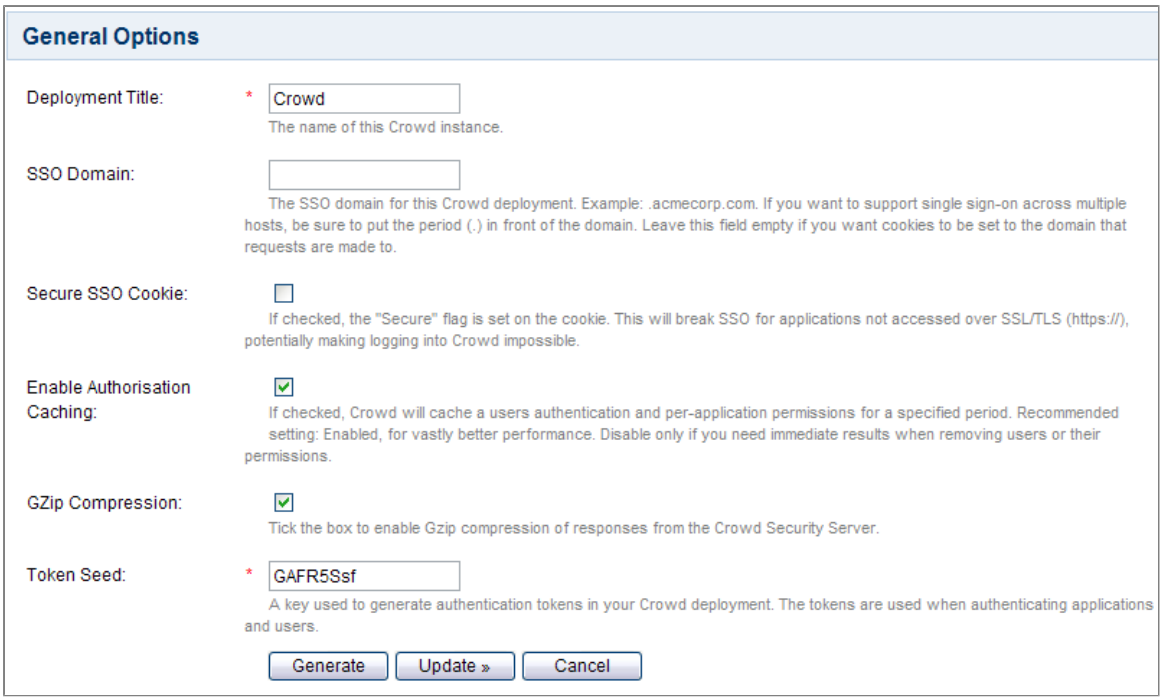
When using Crowd for single sign-on (SSO), you can specify that the 'secure' flag is set on the SSO cookie. This will enforce a secured connection, such as SSL, for all SSO requests.

Unsecured connections will be rejected
If you set this flag, any applications not using a secure connection will not be able to participate in SSO and users will not be able to log in. Potentially, this may make it impossible to log in to Crowd, if your Crowd Administration Console application is not accessed via SSL.

To specify the secure flag on the SSO cookie,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. The '**General Options**' screen will appear. Tick or untick the '**Secure SSO Cookie**' checkbox as required:
   - Ticked — The 'secure' attribute will be included on the SSO cookie. A secured connection, such as SSL or TLS, is required for all SSO requests. Unsecured connections will be refused.
   - Not ticked — This is the default. The 'secure' attribute will not be included on the SSO cookie. This means that the SSO cookie may be transmitted over an unsecured connection.
4. Click the '**Update**' button.

Screenshot: Secure SSO Cookie in Crowd General Options



RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Configuring your Mail Server

Once you have configured your mail server as described below, Crowd can send email notifications to users at specific events, such as when a user's password is reset or a server event occurs.

> To customise the password notification message, please see Creating an Email Notification Template.

On this page:

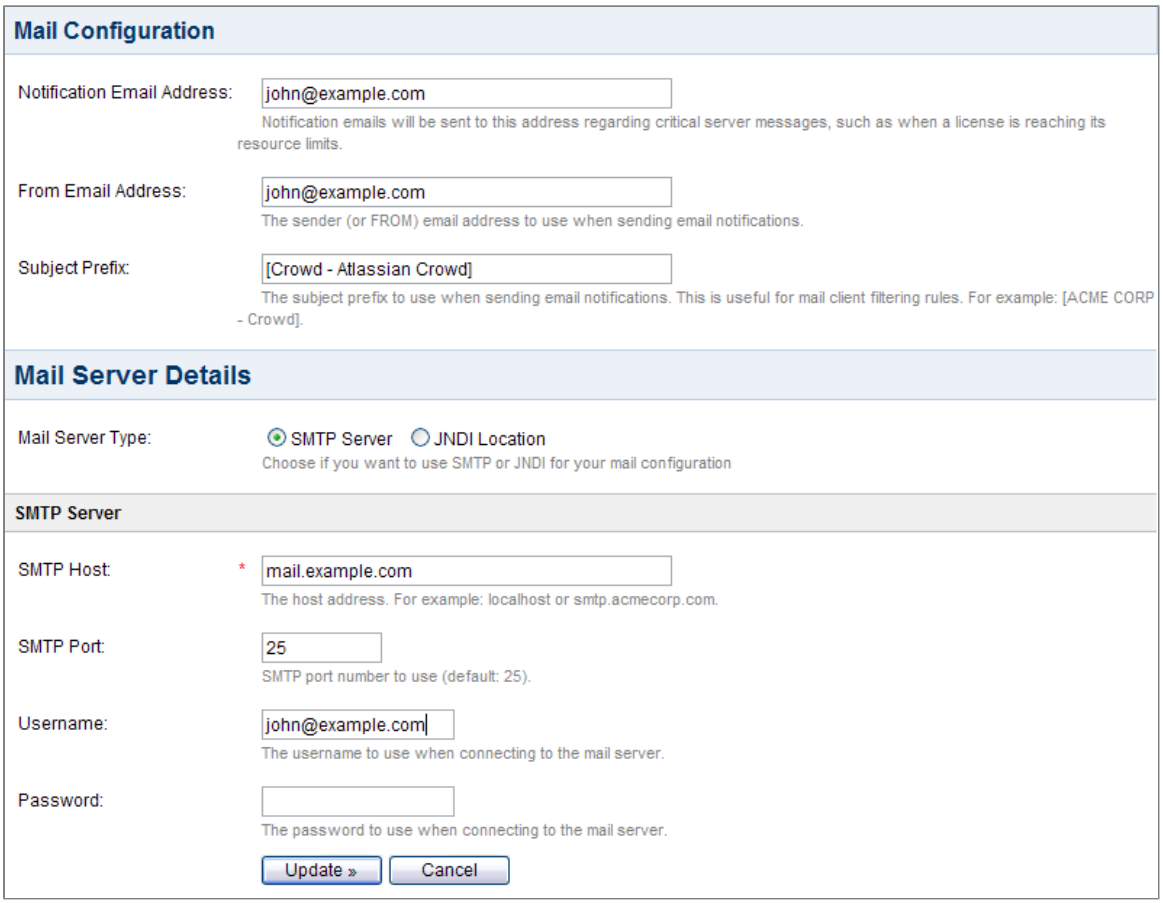
- Accessing the Mail Configuration Screen
- Mail Server Option 1: SMTP
- Mail Server Option 2: JNDI Location
  - Configuring the JNDI Resource
  - SMTP over SSL

Accessing the Mail Configuration Screen

To configure SMTP email,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Mail Configuration**' in the left-hand menu.
4. The 'Mail Configuration' screen allows you to choose between an SMTP and a JNDI mail server. Enter the details of your mail server as described below, then click the '**Update**' button.

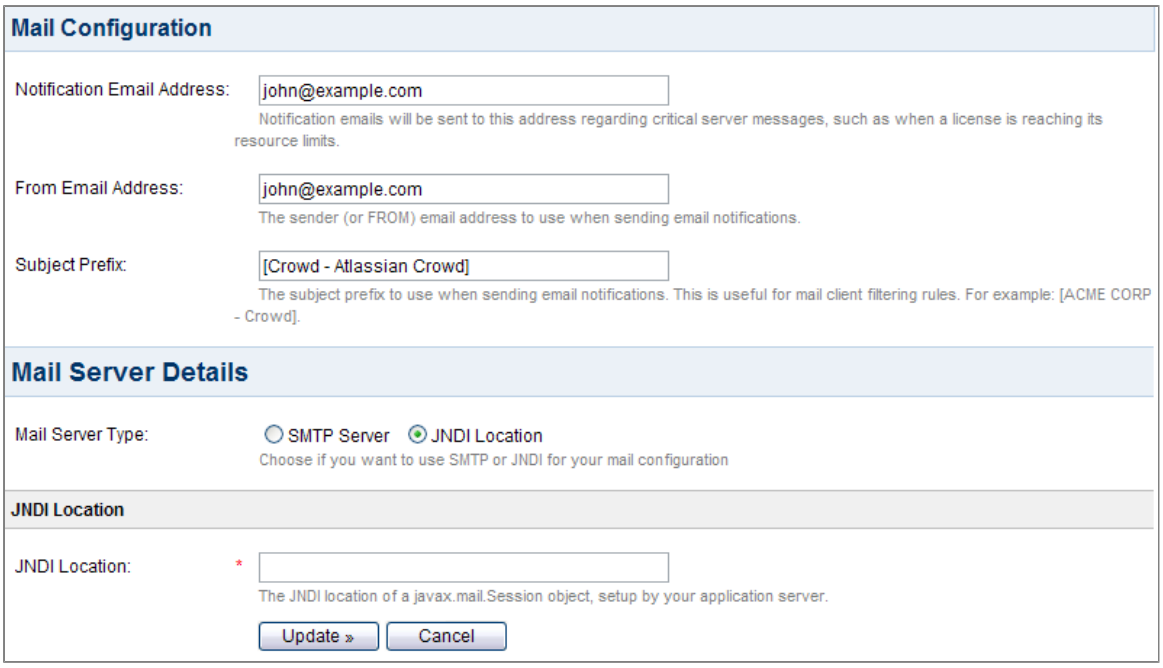
Mail Server Option 1: SMTP



Enter the details as follows:

- **Notification Email Address** — The email address which will receive notifications about server events. For example, Crowd will send an email warning to this address when the number of users approaches the license limit.
- **From Email Address** — Crowd will add this email address as the 'sender' on the emails generated by Crowd and sent to users.
- **Subject Prefix** — The prefix which will appear at the start of the email subject, for all emails generated by Crowd. This can be useful for email client programs that offer filtering rules.
- **Mail Server Type** — Select the '**SMTP Server**' radio button.
- **SMTP Host** — The hostname of the SMTP mail server, e.g. 'localhost' or 'smtp.acme.com'.
- **SMTP Port** — The port on which the SMTP mail server listens. The default is '25'.
- **Username** — The username that your Crowd server will use when it logs in to your mail server.
- **Password** — The password that your Crowd server will use when it logs in to your mail server.

Mail Server Option 2: JNDI Location



Select the '**JNDI Location**' if you want to connect to a mail server via a datasource managed by your application server.

Enter the details as follows:

- **Notification Email Address** — The email address which will receive notifications about server events.
- **From Email Address** — Crowd will add this email address as the 'sender' on the emails generated by Crowd and sent to users.
- **Subject Prefix** — The prefix which will appear at the start of the email subject, for all emails generated by Crowd. This can be useful for email client programs that offer filtering rules.
- **Mail Server Type** — Select the '**JNDI Location**' radio button.
- **JNDI Location** — The datasource name of a `javax.mail.Session` object which has been set up by your application server.

Configuring the JNDI Resource

For example, in Tomcat 5.5 (the default application server that is bundled with Crowd Standalone), your JNDI location would be `java:comp/env/mail/CrowdMailServer`, and you would add the following section in `conf/server.xml` or `conf/Catalina/localhost/crowd.xml`, inside the `<Context>` node:

```
<Context path="/crowd" docBase="/path/to/atlassian-crowd-war-directory" reloadable="false">
    <Resource name="mail/CrowdMailServer"
         auth="Container"
         type="javax.mail.Session"
         mail.smtp.host="yourmailserver.example.com"
         mail.smtp.port="25"
         mail.transport.protocol="smtp"
         mail.smtp.auth="true"
         mail.smtp.user="your_userid"
         password="your_password"
       />
</Context>
```

If you have problems connecting, add a **mail.debug="true"** parameter, which will let you see SMTP-level details when testing the connection.

You will also need to ensure that the JavaMail classes and Java Beans Activation Framework are present in your application server's classpath.

If JavaMail is not present in your application server installation, you will receive the following error in your log file:

```
java.lang.NoClassDefFoundError: javax/mail/Authenticator
```

If the Activation Framework is not present in your application server installation, you will receive the following error in your log file:

```
java.lang.NoClassDefFoundError: javax/activation/DataSource
```

SMTP over SSL

You can encrypt email communications between Crowd and your mail server via SSL, provided your mail server supports SSL.

To do this, edit your mail server connection properties and specify `starttls` and `SSLSocketFactory`, e.g.:

```
<Context path="/crowd" docBase="/path/to/atlassian-crowd-war-directory" reloadable="false">
    <Resource name="mail/CrowdMailServer"
        auth="Container"
        type="javax.mail.Session"
        mail.smtp.host="yourmailserver.example.com"
        mail.smtp.port="465"
        mail.transport.protocol="smtp"
        mail.smtp.auth="true"
        mail.smtp.user="your_userid"
        password="your_password"
        mail.smtp.starttls.enable="true"
        mail.smtp.socketFactory.class="javax.net.ssl.SSLSocketFactory"

      />
</Context>
```

Additionally, as you are connecting to an SSL service, you will need to import the SMTP server certificate into a Java keystore. The process is described in Configuring Crowd to Work with SSL.

RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

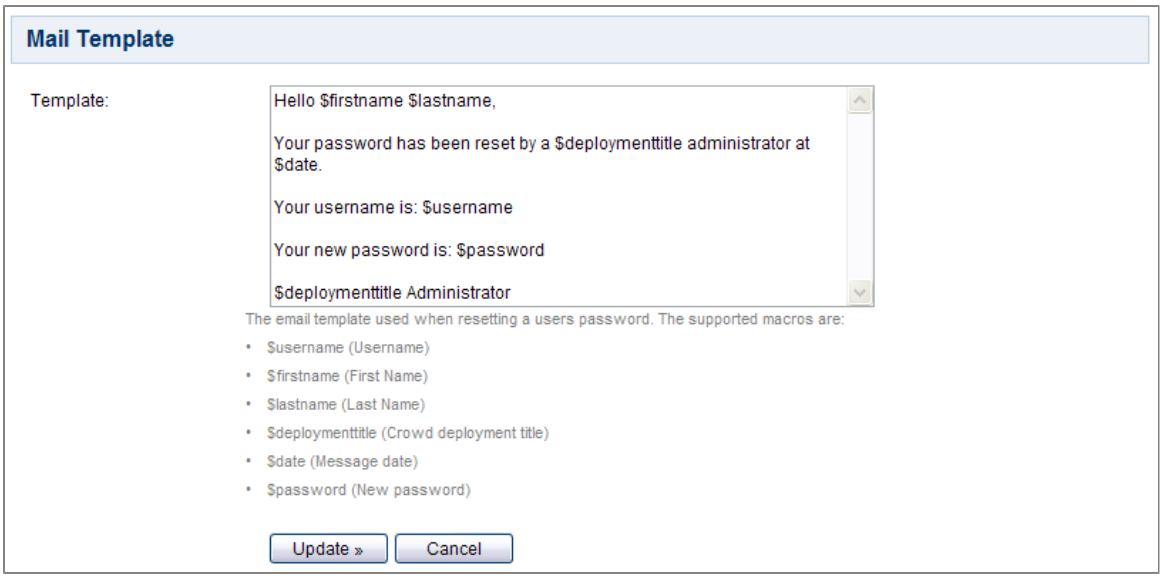
Creating an Email Notification Template

Crowd uses the email template when sending an automatic notification to a user, such as when you reset a user's password.

To set up the email template,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Mail Template**' in the left-hand menu.
4. Enter the text and macros that will form the body of the email message. Use a macro when you want to include a variable into the email text. Crowd will replace the macro with the relevant value when it sends the email. Below are the available macros and their replacement values:
   - **$username** – The username of the person who will receive the email. (This macro is available in **Crowd 2.0.3** and later.)
   - **$firstname** – The user's first name.
   - **$lastname** – The user's last name.
   - **$deploymenttitle** – The title of your Crowd site, as defined in Deployment Title.
   - **$date** – The date/time of the message event.
   - **$password** – The user's new password.
5. Click '**Update**'.

Screenshot: Mail Template



RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Configuring Trusted Proxy Servers

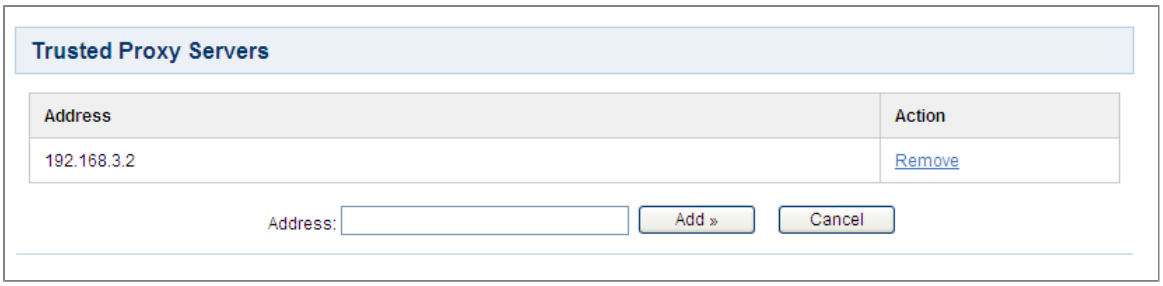
If you are running applications behind one or more proxy servers, you may find it useful to configure Crowd to trust the proxies' IP addresses. When a proxy server forwards an HTTP request, Crowd will recognise the request as coming from the request's originator, not the proxy server. This is particularly useful if you want single sign-on amongst several applications running behind different proxy servers.

Configuring a trusted proxy server means that Crowd will use the rightmost IP address of the `X-Fowarded-For:` header when verifying the client's IP address.

To configure Crowd to trust a proxy server,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Trusted Proxy Servers**' in the left-hand menu.
4. The '**Trusted Proxy Servers**' screen appears. Type the IP address of the proxy server.
5. Click the '**Add**' button.

Screenshot: Trusted Proxy Servers



RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Viewing Crowd's System Information

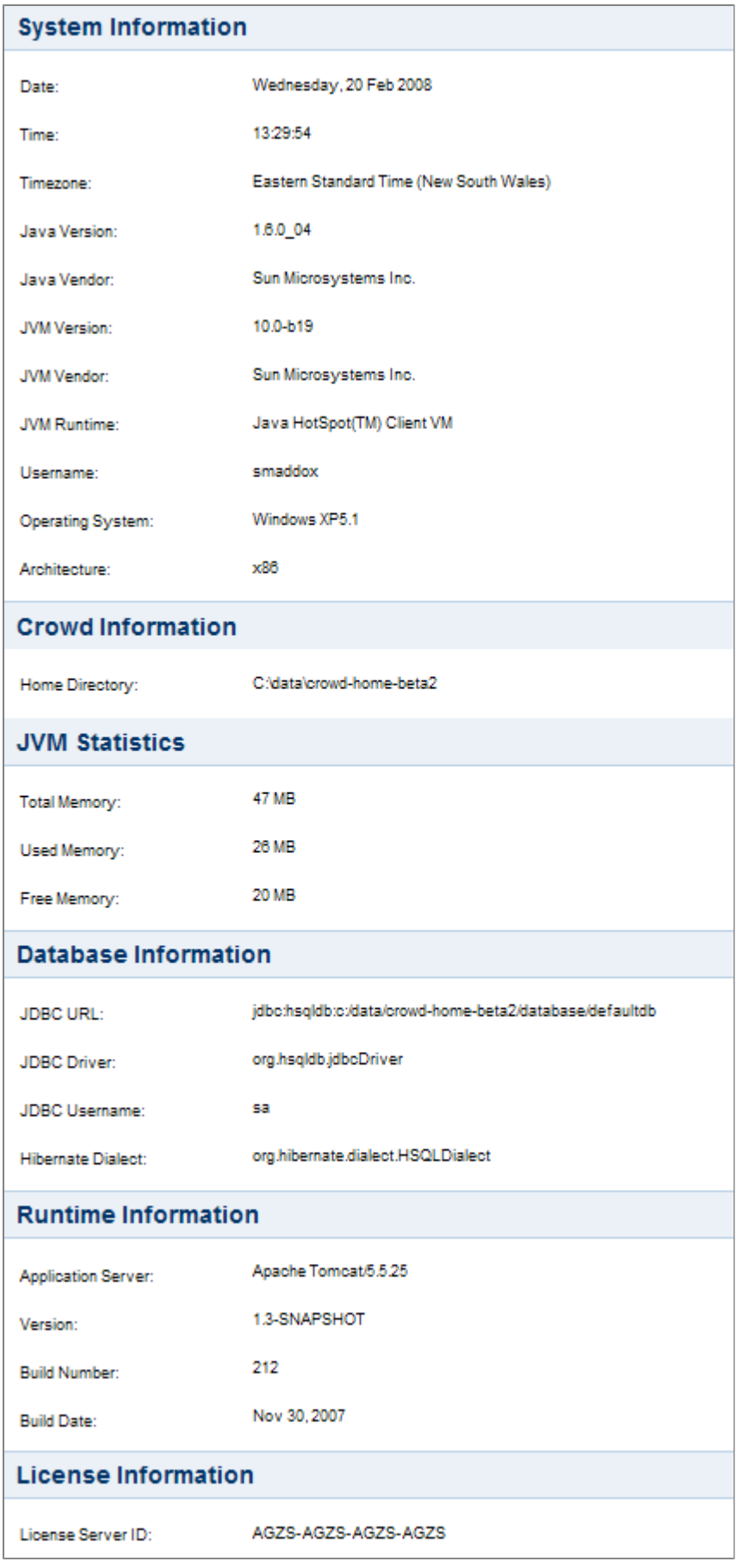
Crowd provides a useful summary of your server's system information, including:

- Time and date information
- Java version
- Location of your Crowd Home directory
- Memory usage
- Application server details
- Database information
- Server ID (see Licensing for more details)

To view your Crowd server's system information,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**System Info**' in the left-hand menu.

Screenshot: 'System Information'



System Information

Date:	Wednesday, 20 Feb 2008
Time:	13:29:54
Timezone:	Eastern Standard Time (New South Wales)
Java Version:	1.6.0_04
Java Vendor:	Sun Microsystems Inc.
JVM Version:	10.0-b19
JVM Vendor:	Sun Microsystems Inc.
JVM Runtime:	Java HotSpot(TM) Client VM
Username:	smaddox
Operating System:	Windows XP5.1
Architecture:	x86

Crowd Information

Home Directory:	C:\data\crowd-home-beta2

JVM Statistics

Total Memory:	47 MB
Used Memory:	26 MB
Free Memory:	20 MB

Database Information

JDBC URL:	jdbc:hsqldb:c:/data/crowd-home-beta2/database/defaultdb
JDBC Driver:	org.hsqldb.jdbcDriver
JDBC Username:	sa
Hibernate Dialect:	org.hibernate.dialect.HSQLDialect

Runtime Information

Application Server:	Apache Tomcat/5.5.25
Version:	1.3-SNAPSHOT
Build Number:	212
Build Date:	Nov 30, 2007

License Information

License Server ID:	AGZS-AGZS-AGZS-AGZS

RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain

    - Token Seed
    - Session Configuration
    - Authorisation Caching
    - Compression of Server Output
    - Licensing
    - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
    - Performance Profiling
- Overview of Caching

Crowd Documentation

Backing Up and Restoring Data

You can back up your Crowd data by exporting it to an XML file. The data includes:

- Your Crowd server configuration details, including connection details for all your directories and applications.
- Any internal directories that exist.

Important Note about Crowd Backup Functionality
At present, Crowd does not allow you to schedule periodic backups. We do have an open feature request for this. Until this feature is added to Crowd, we recommend using alternative backup methods such as:

- A periodic backup or dump of your database using tools provided by your database.
- A backup of your Crowd Home directory using external backup tools.

We recommend that you back up your data regularly, especially after any significant configuration changes. You should also perform regular backups of your database.

To back up your Crowd data,

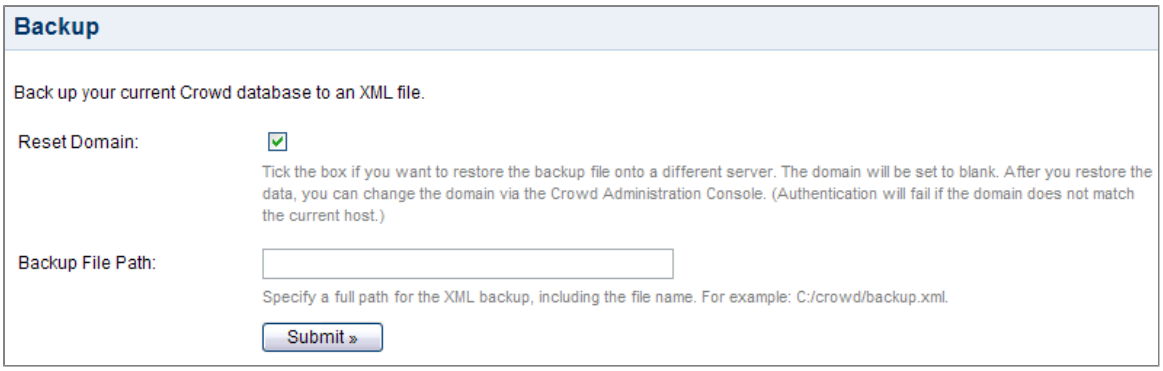
1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Backup**' in the left-hand menu.
4. Select the '**Reset Domain**' checkbox if the backup file will be restored onto a different server. Selecting '**Reset Domain**' will reset the domain to blank. (After you restore the data, you can change the domain as described in Domain.)
5. Type an appropriate '**Backup File Path**', including the name of the XML file.
6. Click the '**Submit**' button.

To restore your Crowd data,

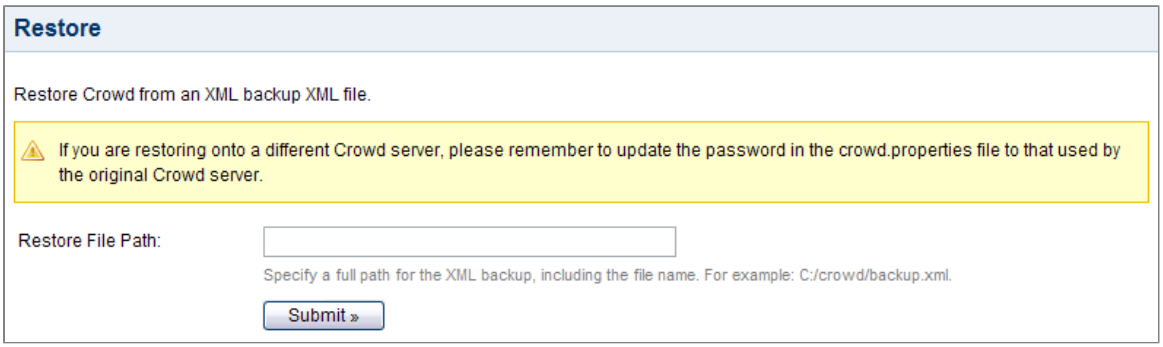
Before you begin: If you created the XML backup file on a different server, edit the `crowd.properties` file and change the password to match the password of the server on which you created the XML backup file.

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Restore**' in the left-hand menu.
4. In the '**Restore File Path**' field, type the path to the backup file, including the name of the XML file.
5. Click the '**Submit**' button.

Screenshot 1: 'Backup'



Screenshot 2: 'Restore'



RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template
- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
  - Performance Profiling
- Overview of Caching

Crowd Documentation

Logging and Profiling

When troubleshooting problems with your Crowd installation, it is often useful to change the level of information provided by your Crowd server so that more information, messages and warnings are shown than usual. This page describes how to:

- Adjust the settings which affect Crowd's logging.
- Enable performance profiling.

  With performance profiling turned on, your system output console will show a record of the time it takes (in milliseconds) to complete each Crowd action. This will help with diagnosing performance problems. The resulting output will be large, so you should not enable it for long periods.

  You can see an example of performance profiling output here.

On this page:

- Summary of the Logging Levels
- Finding the Crowd Log File
- Changing the Log Settings
- Configuring the Log Settings and Performance Profiling via the Administration Console
- Advanced Log Configuration
  - Finding the Log Configuration File
  - Editing the Log Configuration File

- Changing the Destination of the Crowd Log File
- Adjusting the Log Settings for CrowdID

Summary of the Logging Levels

Crowd uses Apache's log4j logging service. The amount of information written to the log file is determined by the logging 'level'. The type of message output at each level is as follows:

Level	Type of Message Written to the Log
DEBUG	Fine-grained information that is useful for debugging only. These are low-level details that most people never need to know about.
INFO	Informational messages about what Crowd is doing. Usually not interesting.
WARN	Warnings that something may have gone wrong, or other messages a system administrator may wish to know. These are conditions that, while not errors in themselves, may indicate that the system is running sub-optimally.
ERROR	Indications that something has gone wrong in Crowd. The person responsible for configuring Crowd should be notified.
FATAL	Indications that something has gone wrong so badly that the system cannot recover.
ALL	All possible log messages.

Finding the Crowd Log File

When you report a problem to Atlassian Support, we may ask you to send us your `atlassian-crowd.log` file. The location of the log file may vary, depending on your Crowd installation type. Provided that you have not changed the log file location from the default, the Crowd log file is at the location described below.

Installation Type	Location of Log File
Crowd Standalone edition	The root directory of your Crowd application, e.g. `atlassian-crowd-2.0.0/atlassian-crowd.log`
Crowd Standalone running as a Windows service	`C:\Windows\system32\atlassian-crowd.log`
Crowd WAR edition	The directory from which you start the application server, e.g. `apache-tomcat-6.0.16/bin/atlassian-crowd.log`

Changing the Log Settings

You can change the log settings in two ways:

- Set the logging levels at runtime via the Administration Console, as described immediately below. Your changes will be in effect only until you next restart Crowd.
- Or edit the log configuration file, as described in the Advanced section below. Your changes will take effect next time you start Crowd, and for all subsequent sessions.

Configuring the Log Settings and Performance Profiling via the Administration Console

If necessary, you can edit the configuration file directly
If you change the log settings via the Administration Console, the changes are not written to the `log4j.properties` file and are therefore discarded when you next stop Crowd. Also, not all logging behaviour can be changed via the Administration Console. For logging configuration not mentioned below, or to change the log settings permanently, you will need to stop Crowd and then edit the log configuration file instead.

The '**Logging & Profiling**' screen tells you whether performance profiling is currently on or off, and shows a list of all currently defined loggers. On this screen you can:

- Turn **performance profiling** on or off.

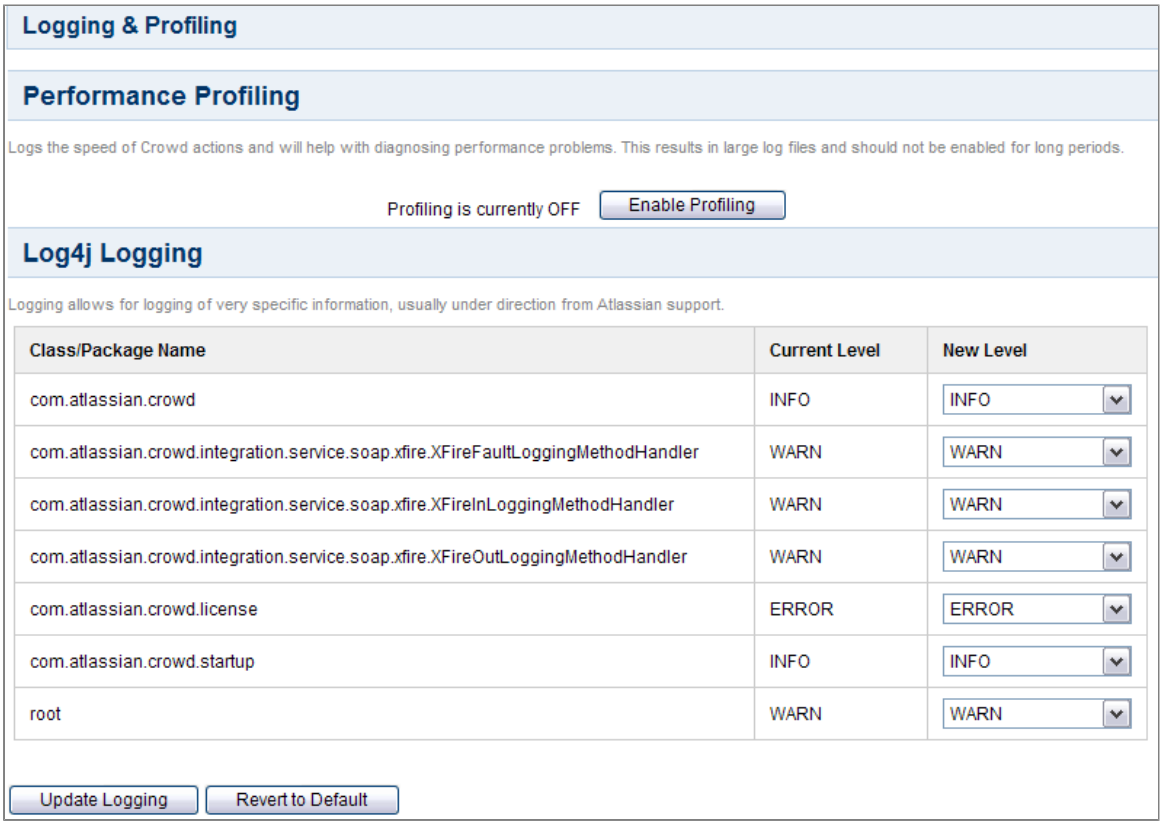
  With performance profiling turned on, your system output console will show a record of the time it takes (in milliseconds) to complete each Crowd action. This will help with diagnosing performance problems. The resulting output will be large, so you should not enable it for long periods.

  You can see an example of performance profiling output here.
- Set the **logging level** for each class or package name, or reset all logging levels to the default setting.
  Refer to the section on logging levels above. Any changes made in this way will apply only to the currently-running Crowd lifetime.

To configure profiling and logging,

1. Log in to the Crowd Administration Console.
2. Click the '**Administration**' tab in the top navigation bar.
3. Click '**Logging & Profiling**' in the left-hand menu.
4. The '**Logging and Profiling**' screen appears, as shown below. The screen has the following sections:
    - '**Performance Profiling**' — Click the '**Enable Profiling**' button to turn profiling on, or '**Disable Profiling**' to turn it off. (You will only see one of these buttons.)
    - '**Log4j Logging**' — This section shows the loggers currently in action for your Crowd instance.
        - You can change the logging level by selecting a value from the '**New Level**' dropdown list. Above is a definition of each level. You can also read the Apache documentation for more information.
        - You can click the '**Revert to Default**' button if you want to reset the logging levels to the values shipped with your Crowd installation.
5. Click the '**Update Logging**' button to save any changes you have made in the '**Log4j Logging**' section.

Screenshot: Changing Log Levels and Profiling



Description of the loggers:

Logger	Description
`com.atlassian.crowd`	This is the parent of the **crowd** package loggers. Any children which do not have a level assigned to them will inherit the level from their parent.
`com.atlassian.crowd....XFireFaultLoggingMethodHandler`	Can be helpful if a Crowd SOAP service fault is thrown. It is best to enable DEBUG for all three XFire classes simultaneously when troubleshooting Crowd's SOAP service.
`com.atlassian.crowd....XFireOutLoggingMethodHandler`	The Crowd server outputs the incoming SOAP request method and parameters. This is useful when debugging your applications or monitoring the level of traffic for an integrated application.
`com.atlassian.crowd....XFireInLoggingMethodHandler`	The Crowd server outputs the outgoing SOAP request method and parameters. This is useful when debugging your applications or monitoring the level of traffic for an integrated application.
`com.atlassian.crowd.license`	Useful for troubleshooting certain licensing issues in Crowd.
`com.atlassian.crowd.startup`	Can be helpful for troubleshooting startup errors in Crowd.

`root`	This is the root of the logger hierarchy, i.e. it is the parent of all loggers. The level assigned to the root will be the default level for any loggers which do not have a specific level and do not inherit from another parent.

Advanced Log Configuration

Terminology: In log4j, a 'logger' is a named entity. Logger names are case sensitive and follow a hierarchical naming standard. For example, the logger named `com.foo` is a parent of the logger named `com.foo.Bar`.

Finding the Log Configuration File

Crowd's logging behaviour is defined in the following properties file:

- For Standalone installations of Crowd: {`CROWD-STANDALONE-INSTALL`}`/crowd-webapp/WEB-INF/classes/log4j.properties`
- For WAR installations: {`CROWD-WAR-INSTALL`}`/WEB-INF/classes/log4j.properties`

This file is a standard log4j configuration file, as described in the Apache log4j documentation.

Editing the Log Configuration File

To configure the logging levels and other settings on a permanent basis:

1. Stop Crowd.
2. With a text editor, open the `log4j.properties` file in the location described above.
3. Adjust the output level to the required level of importance listed in the section on levels above.
4. Save the `log4j.properties` file.
5. Restart Crowd to have the new log settings take effect.

When diagnosing a server problem you need to adjust Crowd's package logging to:
`log4j.logger.com.atlassian.crowd=DEBUG`

Changing the Destination of the Crowd Log File

Terminology: In log4j, an output destination is called an 'appender'.

To change the destination of the Crowd log file:

1. Stop Crowd.
2. With a text editor, open the `log4j.properties` file in the location described above.
3. Look for the `org.apache.log4j.RollingFileAppender` entry in the '**Log File Locations**' section of the file. This appender controls the default logging destination described above.
4. Edit the following line, and replace `atlassian-crowd.log` with the full path and file name for the required logging destination: `log4j.appender.filelog.File=atlassian-crowd.log`.
5. Save the `log4j.properties` file.
6. Restart Crowd to have the new log settings take effect.

Adjusting the Log Settings for CrowdID

The Crowd Administration Console does not give access to the **CrowdID** log settings. To adjust the logging levels of the CrowdID OpenID server, you will need to modify the configuration file at this location:

- For Standalone installations of CrowdID: {`CROWDID-STANDALONE-INSTALL`}`/crowd-openidserver-webapp/WEB-INF/classes/log4j.properties`
- For WAR installations: {`CROWDID-WAR-INSTALL`}`/WEB-INF/classes/log4j.properties`

RELATED TOPICS

- Finding the atlassian-crowd.log File
- Configuring Server Settings
  - Deployment Title
  - Domain
  - Token Seed
  - Session Configuration
  - Authorisation Caching
  - Compression of Server Output
  - Licensing
  - SSO Cookie
- Configuring your Mail Server
- Creating an Email Notification Template

- Configuring Trusted Proxy Servers
- Viewing Crowd's System Information
- Backing Up and Restoring Data
- Logging and Profiling
    - Performance Profiling
- Overview of Caching

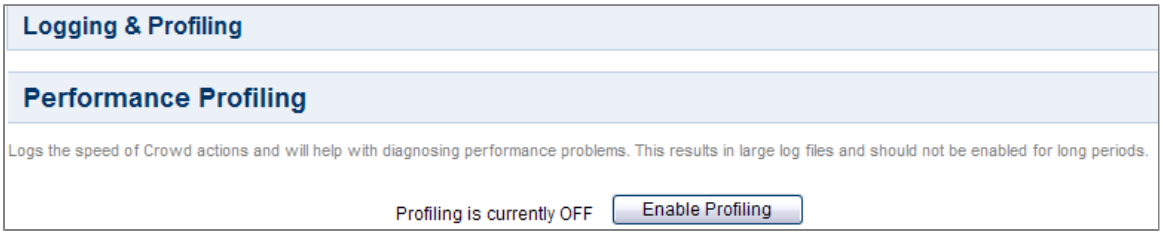
Crowd Documentation

Performance Profiling

When troubleshooting problems with your Crowd installation, it is often useful to turn on performance profiling.

To enable profiling, go to the '**Logging & Profiling**' tab under '**Administration**' in the Crowd Administration Console. Full instructions are in the section on logging and profiling.

Screenshot: Performance Profiling



With performance profiling turned on, your system output console will show a record of the time it takes (in milliseconds) to complete each Crowd action. This will help with diagnosing performance problems. The resulting output will be large, so you should not enable it for long periods.

Here is an example of the performance profiling output, when search for and viewing a user via the Crowd Administration Console:

```
01.[15ms] - AOP: SecurityServer.findPrincipalByToken()
02.
03.[15ms] - AOP: SecurityServer.isValidPrincipalToken()
04.
05.[15ms] - AOP: SecurityServer.isValidPrincipalToken()
06.     [15ms] - AOP: SOAPService.validateSOAPService()
07.
08.[15ms] - AOP: SecurityServer.isValidPrincipalToken()
09.
10.[16ms] - AOP: SecurityServer.getDomain()
11.     [16ms] - AOP: SOAPService.validateSOAPService()
```

RELATED TOPICS

Logging and Profiling

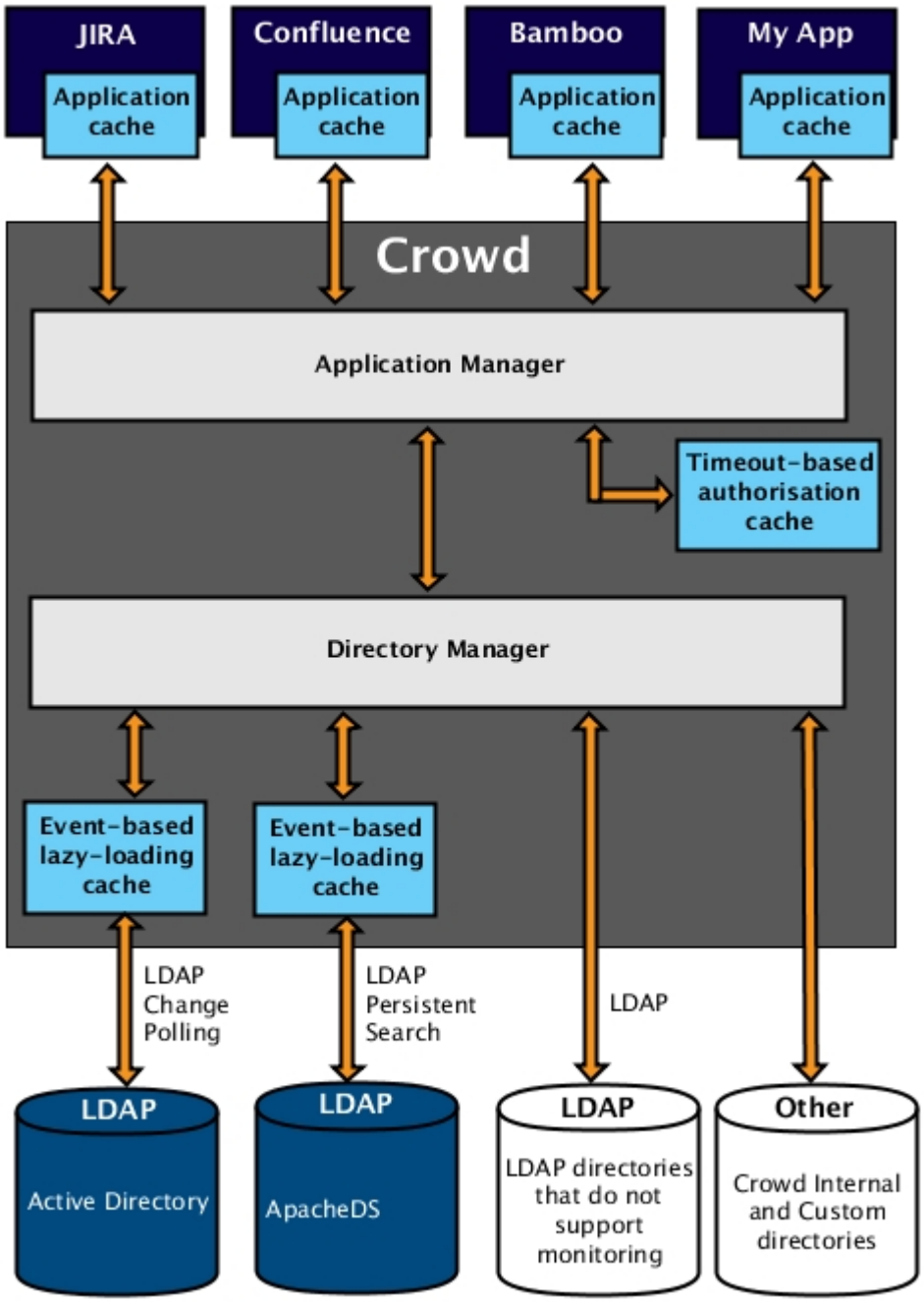
Overview of Caching

Caching is used to store run-time authentication and authorisation rules, which can be expensive to calculate.

In Crowd, data caching occurs in three main areas:

- **Application caches in the applications that are connected to Crowd** — Applications such as JIRA, Confluence and Bamboo can store user, group and role data in a local cache. This helps improve the performance of Crowd, since these applications do not have to repeatedly request information from Crowd. Generally it is not necessary to configure application caching, although this depends on the size of your application deployments. You can set the options for application caching in the cache configuration file for that application. See Configuring Caching for an Application.
- **An authorisation cache on the Crowd server** — To improve performance, Crowd can store users' authentication and per-application permissions in a local cache for a specified period. You can enable or disable this cache via an option on the 'General Options' screen in the Crowd Administration Console. See Authorisation Caching.
- **LDAP directory caches on the Crowd server** — Where the LDAP directory supports it, Crowd will keep an up-to-date cache of all user and group information from the directory. Crowd monitors changes from the remote directory and incrementally updates the cache. You can configure this cache on the Directory Connector screen. See Configuring Caching for an LDAP Directory.

This diagram gives a conceptual overview of the caches described above:



RELATED TOPICS

- Configuring Caching for an LDAP Directory
- Authorisation Caching
- Configuring Caching for an Application
- A blog post by a Crowd developer on Caching in Crowd 1.6

Crowd Documentation

Crowd Security Advisories and Fixes

This page has information on how to report any security bugs you might find in Crowd, and what we will do to fix the problem and announce the solution.

On this page:

- Finding and Reporting a Security Vulnerability
- Publication of Security Advisories
- Severity Levels

- Patches and Fixes
- Published Security Advisories

Finding and Reporting a Security Vulnerability

Atlassian's approach to reporting security vulnerabilities is detailed in How to Report a Security Issue.

Publication of Security Advisories

Atlassian's approach to releasing security advisories is detailed in Security Advisory Publishing Policy.

Severity Levels

Atlassian's approach to ranking security issues is detailed in Severity Levels for Security Issues.

Patches and Fixes

Atlassian's approach to releasing patches for security issues is detailed in Security Patch Policy.

Published Security Advisories

- Crowd Security Advisory 2008-10-14 - Parameter Injection Vulnerability

Crowd Security Advisory 2008-10-14 - Parameter Injection Vulnerability

In this advisory:

- Parameter Injection Vulnerability in Crowd
  - Severity
  - Risk Assessment
  - Risk Mitigation
  - Vulnerability
  - Fix

Parameter Injection Vulnerability in Crowd

Severity

Atlassian rates this vulnerability as **critical**, according to the scale published in Crowd Security Advisories and Fixes. The scale allows us to rank a vulnerability as critical, high, moderate or low.

Risk Assessment

We have identified and fixed a flaw which would allow a malicious user (hacker) to inject their own values into a Crowd request by adding parameters to the URL string. This would allow a hacker to bypass Crowd's security checks and perform actions that they are not authorised to perform.

Risk Mitigation

To address the issue, you should upgrade Crowd as soon as possible. Please follow the instructions in the 'Fix' section below. If you judge it necessary, you can block all untrusted IP addresses from accessing Crowd.

Vulnerability

A hacker can design a URL string containing parameters which perform specific actions on the Crowd server, bypassing Crowd's security checks. This is because Crowd does not adequately sanitise user input before applying it as an action on the server.

Exploiting this issue could allow an attacker to access or modify data and compromise the Crowd application.

The following Crowd versions are vulnerable: All versions from **1.0 to 1.5.0** inclusive.

Fix

Please download the relevant upgrade file for your version of Crowd from the download centre as follows:

- If you have Crowd 1.5.0 — upgrade to **Crowd 1.5.1** (see the release notes and upgrade guide).
- If you have Crowd 1.4.x — upgrade to **Crowd 1.4.7** (see the release notes and upgrade guide).
- If you have Crowd 1.3.x — upgrade to **Crowd 1.3.3** (see the release notes and upgrade guide).
- If you have Crowd 1.2.x — upgrade to **Crowd 1.2.4** (see the release notes and upgrade guide).

Crowd Installation & Upgrade Guide

- Crowd Release Notes
- Installing Crowd
- Upgrading Crowd
- Migrating Crowd between Servers

Crowd Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

Installation

Information for installing Crowd can be found here. If upgrading from a previous version, please follow the Upgrade Guide.

Crowd Release Notes

- Crowd Release Summary
- Crowd 2.0.3 Release Notes
- Crowd 2.0.2 Release Notes
- Crowd 2.0.1 Release Notes
- Crowd 2.0 Release Notes
- Crowd 2.0 Beta Release Notes
- Crowd 1.6.1 Release Notes
- Crowd 1.6 Release Notes
- Crowd 1.5.2 Release Notes
- Crowd 1.5.1 Release Notes
- Crowd 1.5 Release Notes
- Crowd 1.4.7 Release Notes
- Crowd 1.4.4 Release Notes
- Crowd 1.4.3 Release Notes
- Crowd 1.4.2 Release Notes
- Crowd 1.4.1 Release Notes
- Crowd 1.4 Release Notes
- Crowd 1.3.3 Release Notes
- Crowd 1.3.2 Release Notes
- Crowd 1.3.1 Release Notes
- Crowd 1.3 Release Notes
- Crowd 1.3 Beta Release Notes
- Crowd 1.2.4 Release Notes
- Crowd 1.2.2 Release Notes
- Crowd 1.2.1 Release Notes
- Crowd 1.2 Release Notes
- Crowd 1.1.2 Release Notes
- Crowd 1.1.1 Release Notes
- Crowd 1.1.0 Release Notes
- Crowd 1.0.7 Release Notes
- Crowd 1.0.6 Release Notes
- Crowd 1.0.5 Release Notes
- Crowd 1.0.4 Release Notes
- Crowd 1.0.3 Release Notes
- Crowd 1.0.2 Release Notes
- Crowd 1.0.1 Release Notes
- Crowd 1.0.0 Release Notes
- Crowd 0.4.5 Beta Release Notes
- Crowd 0.4.4 Beta Release Notes
- Crowd 0.4.3 Beta Release Notes
- Crowd 0.4.2 Beta Release Notes
- Crowd 0.4.1 Beta Release Notes
- Crowd 0.4 Beta Release Notes
- Crowd 0.3.3 Beta Release Notes
- Crowd 0.3.2 Beta Release Notes
- Crowd 0.3 Beta Release Notes
- Crowd 0.2 Beta Release Notes

Crowd Release Summary

This page shows the highlights of the major Crowd releases.

Current Release

For information about the latest release, please go to the Crowd Release Notes.

Crowd 2.0 — 30 July 2009

- Introducing User Aliases
- Nested Groups in All Crowd Directories
- Automatic Group Membership for New Users
- Improved User and Group Management UI
- Improved Performance
- Improved Database Support
- New REST API
- Plugin Framework 2.2 and REST Module
- More in the Crowd 2.0 release notes

Crowd 1.6 — 17 December 2008

- Smarter Caching
- Quick Application Setup
- Connectors for OpenDS, Fedora DS and OpenLDAP (Posix)
- Spring Security 2
- More in the Crowd 1.6 release notes

Crowd 1.5 — 4 September 2008

- Single Sign-On to Google Apps
- Connector for Apple Open Directory
- Plugin Framework 2.0 and API
- More in the Crowd 1.5 release notes

Crowd 1.4 — 8 May 2008

- Nested Groups
- Self-Service Console
- Novell eDirectory Connector
- Posix Support for LDAP Directories
- Plugin Framework
- More in the release notes

Crowd 1.3 — 4 March 2008

- LDAP Authentication with Crowd Groups and Roles
- Cross-Directory User Importer
- Streamlined User Interface
- Simplified Installation, Setup and Integration
- Configuration of Logging and Profiling via Console
- Improved Performance and Efficiency
- Highlights for the Developers
- Plus Over 60 Improvements and Bug-Fixes
- More in the release notes

Crowd 1.2 — 27 November 2007

- Directory Permissions per Application
- Group and Role Membership Browser
- Improved Browser for OpenID Login History
- NTLM Support
- Improved Integration with Jive Forums
- Acegi Application Connector
- Group-Based Authorisation Added for Subversion
- New Importer for Bamboo Users
- More in the release notes

Crowd 1.1 — 20 June 2007

- OpenID
- More in the release notes

Crowd 1.0 — 5 March 2007

- UI improvements with new screen layouts.
- Import and Export process for XML.
- LDAP Fixes for OpenLDAP and Microsoft Active Directory.
- Improved error reporting.
- Apache / Subversion support
- More in the release notes

Crowd 2.0.3 Release Notes

14 December 2009

The Atlassian Crowd team is delighted to present **Crowd 2.0.3**. This is a bug-fix release with a couple of nice improvements.

Crowd's email template now includes a username macro. This is useful for the automatic emails sent when you change someone's password. You can now include the user's username in the email text, as well as their full name and the new password.

We have also added a number of checks that help prevent you from removing your Crowd administration rights by mistake.

Don't have Crowd 2.0 yet?
Take a look at the new features and other highlights in the Crowd 2.0 Release Notes.

Download Latest Version

Complete List of Fixes in This Release

JIRA Issues (14 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1752	Users with special/unicode chars in the login name or password can't login to Crowd console	↓	Resolved
CWD-1741	Duplicate filter-mapping in web.xml	↓	Resolved
CWD-1736	Crowd installation wizard with MS SQL fails if the Base URL does not contain *localhost*	↓	Resolved
CWD-1728	If the user data is edited using the Self-Service console, the user is set to *Inactive*	↑	Resolved
CWD-1695	Crowd 2.0.X creates NULL attribute value for any blank text box in a Directory definition with Oracle DBs	↑	Resolved
CWD-1687	Make sure that the Admin will not Apply the Self-Service Console configuration to the crowd-administrators group/directory	↓	Resolved
CWD-1675	Directory Importer does not work with LDAP Connector as a Source or Target Directory	↑	Resolved
CWD-1655	Test Search always fails in the Directory Connector *Configuration* tab	↓	Resolved
CWD-1626	Make sure the current Console Administrator cannot unassign a group from the Crowd console that would stop them accessing the console.	↑	Resolved
CWD-1612	When adding an application, you should be able to enter an IP address range	↑	Resolved
CWD-1577	Make sure the current Console Administrator cannot unassign a directory from the Crowd console that would remove them as a Crowd administrator	↑	Resolved
CWD-1576	Make sure the current Console administrator cannot delete a directory they are currently authenticated from	↑	Resolved
CWD-1575	Make sure the current Console administrator cannot remove themselves from a group that will no longer make them a console administrator	↑	Resolved
CWD-1183	Add username macro in email template used when resetting a users password	↑	Resolved

Crowd 2.0.2 Release Notes

6 October 2009

The Atlassian Crowd team is delighted to present **Crowd 2.0.2**. This release contains some good improvements and bug fixes.

Crowd now supports the Atlassian Plugin SDK, so plugin developers can quickly build a Crowd plugin.

The SOAP API has two new methods that return all the attributes associated with the user or group: `findPrincipalWithAttributesByName` and `findGroupWithAttributesByName`.

This release fixes a bug that prevented Crowd from building the group memberships correctly if *both* LDAP caching and nested groups were

enabled in a directory connector. Before this fix, Crowd would build the memberships correctly if LDAP caching was enabled without nested groups, or if nested groups were enabled without LDAP caching, or if neither were enabled. But if both were enabled at the same time, Crowd did not build memberships correctly.

We have upgraded to Apache Tomcat 6 because Tomcat 5 is out of date and has some holes in it. The Crowd standalone distribution now ships with Apache Tomcat 6.0.20. Note that this means a change in the location of database driver JARs. With Tomcat 5, you would add your database driver JAR to your `{CROWD_INSTALL}\apache-tomcat\common\lib` directory. Now with Tomcat 6, you will add your database driver JAR to your `{CROWD_INSTALL}\apache-tomcat\lib` directory.

Don't have Crowd 2.0 yet?
Take a look at the new features and other highlights in the Crowd 2.0 Release Notes.

Download Latest Version

Complete List of Fixes in This Release

JIRA Issues (10 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1707	We need to improve the CacheImpl in the integration client to programatically add caches to the ehcache manager.		Resolved
CWD-1705	Allow dynamically reloadable plugins		Resolved
CWD-1702	Move crowd's standalone distribution to the latest Tomcat distribution		Resolved
CWD-1701	Embedded DB creation fails if the Home-Directory name contains the home directory variable name *crowd.home*		Resolved
CWD-1694	Application IP address 127.0.0.1 should be added automatically in Add Application wizard		Resolved
CWD-1685	Crowd is not building the Group Memberships correctly if both LDAP Caching and Nested Groups are enabled in a connector		Resolved
CWD-1683	Browse token pages are currently using the wrong accessors for createdDate and lastAccessedDate for Token		Resolved
CWD-1677	Add support for subscriptions		Resolved
CWD-1672	Add ability to return User and Groups in the current SecurityServerClient "with attributes"		Resolved
CWD-1608	Multiple sessions appearing at the *Current Sessions* admin page		Resolved

Crowd 2.0.1 Release Notes

27 August 2009

The Atlassian Crowd team is delighted to present **Crowd 2.0.1**.

Crowd now supports the 'range' attribute for retrieving group members from Microsoft Active Directory. For large groups, with more than 1000 members (in AD 2000) or 1500 members (in AD 2003+), Active Directory returns the first 999/1499 members and offers the range attribute for retrieving the next batch of members. Crowd 2.0.1 will make use of this attribute to retrieve the members of large groups.

Take a look at the full list of fixes below.

Don't have Crowd 2.0 yet?
Take a look at the new features and other highlights in the Crowd 2.0 Release Notes.

Download Latest Version

Complete List of Fixes in This Release

JIRA Issues (13 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1660	No Import Passwords checkbox even if Directory is selected with Atlassian-SHA1 password encryption		Closed

CWD-1657	Expose the AliasManager to be available to plugins		Resolved
CWD-1656	Google Apps Integration shows stack-trace when an App is accessed (ie. mail.company_domain.com)		Resolved
CWD-1651	Crowd is providing Uppercased login names to the Apps even if the "force lowercase output" is enabled		Resolved
CWD-1650	Performing a group search with the Search Restriction SearchContext.GROUP_PRINCIPAL_MEMBER currently fails and returns all groups		Resolved
CWD-1647	Error "Illegal Capacity: -1" is displayed and User Memberships are not built if memberOf is used for group membership		Resolved
CWD-1644	entity_picker.js include not aware of server context		Resolved
CWD-1634	Cannot add user to a group if that group is managed in a read only directory		Resolved
CWD-1628	Roles observe group permissioning as opposed to role permissioning		Resolved
CWD-1618	Group Mod Error Handling		Closed
CWD-1572	Make sure the current user cannot delete themselves from Crowd		Resolved
CWD-1445	Support *range* attribute for Active Directory		Resolved
CWD-933	In-memory tokens expire after 5 minutes of inactivity		Resolved

Crowd 2.0 Release Notes

30 July 2009

The Atlassian Crowd team is delighted to present the insanely fast, supremely nested **Crowd 2.0**.

Highlights of this release:

- Introducing User Aliases
- Nested Groups in All Crowd Directories
- Automatic Group Membership for New Users
- Improved User and Group Management UI
- Improved Performance
- Improved Database Support
- New REST API
- Plugin Framework 2.2 and REST Module
- Other Things Worth Mentioning
- Complete List of Improvements and Fixes

Responding to your feedback:

More than 220 votes satisfied

Keep logging your votes and issues. They help us decide what needs doing!

DOWNLOAD latest version

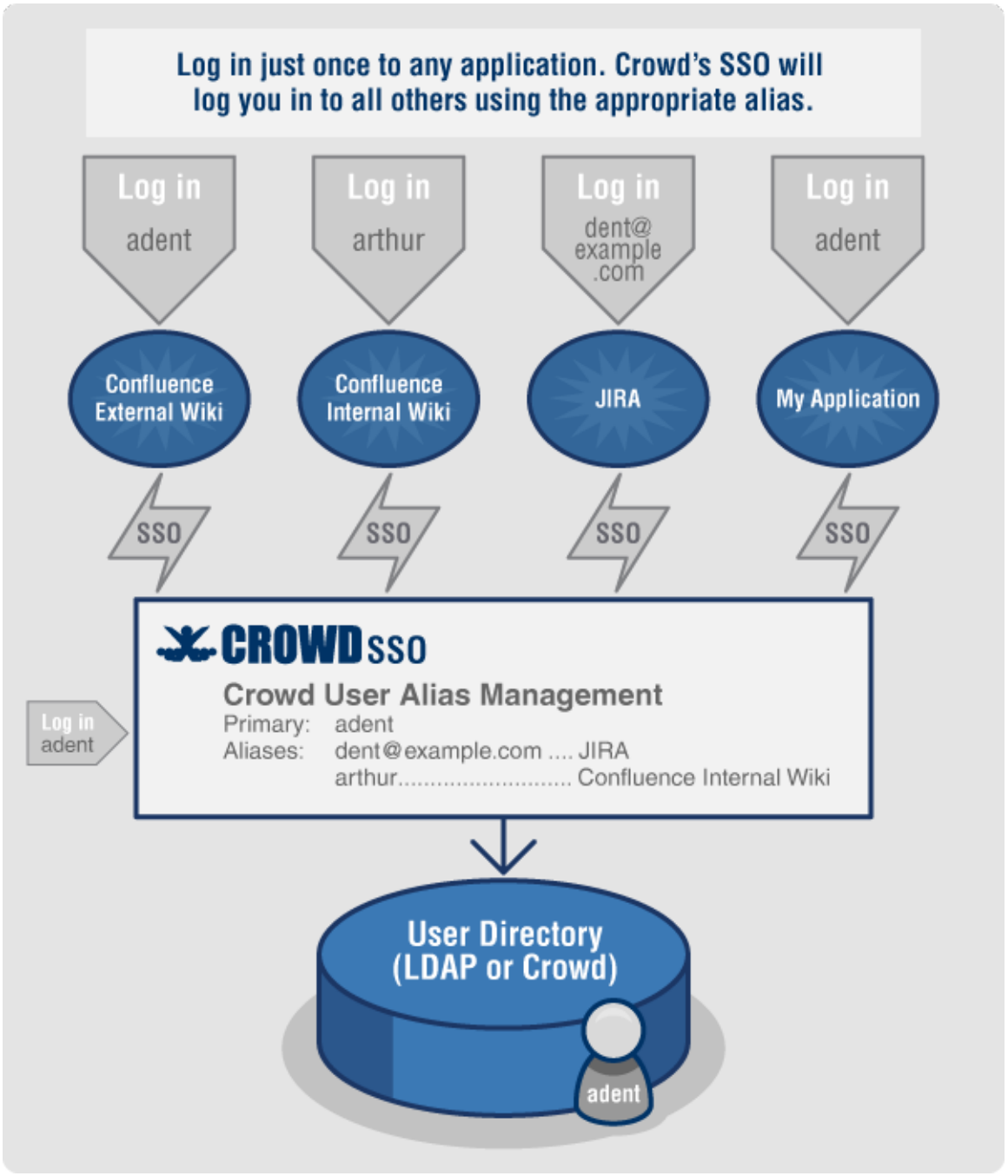
Upgrading to Crowd 2.0
You can download Crowd from the Atlassian website. If upgrading from a previous version, please read the Crowd 2.0 Upgrade Notes.

Highlights of Crowd 2.0

Introducing User Aliases

A single user can now have different usernames in different applications. For example, Arthur Dent might have username 'dent@example.com' in your JIRA issue tracker, 'arthur' in your internal Confluence wiki and 'adent' in your public-facing Confluence wiki.

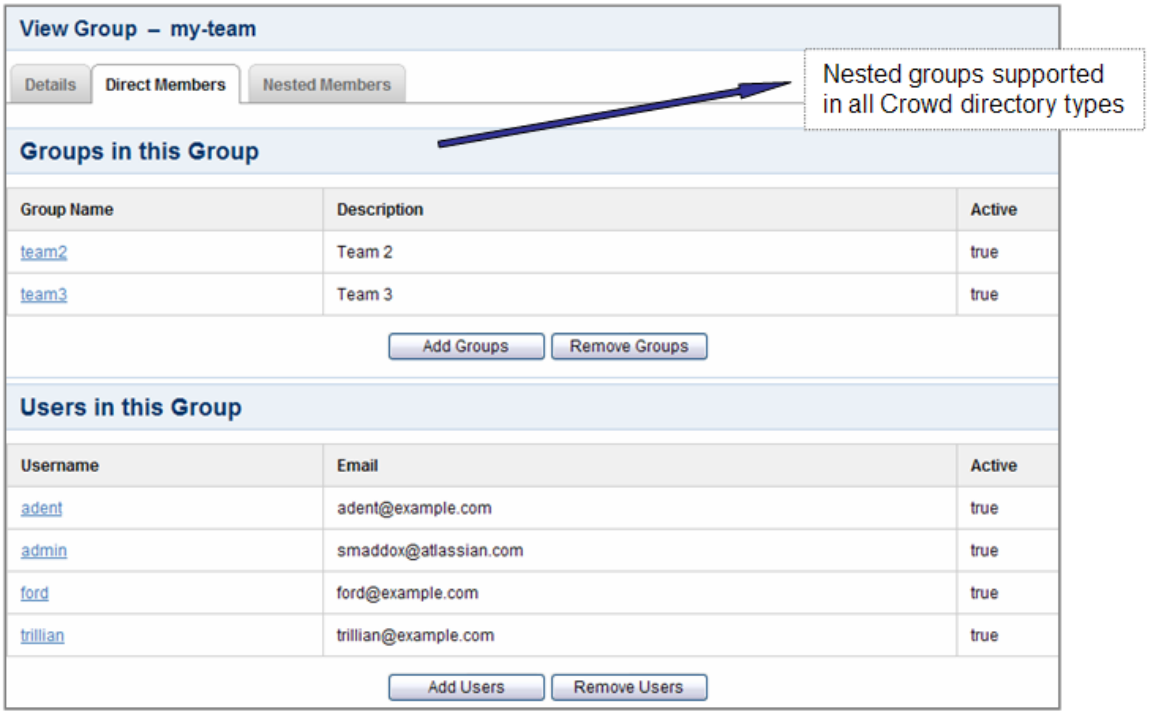
- Using Crowd, Arthur can link a number of usernames as aliases of his main login ID.
- Arthur can log in just once, to any Crowd-connected application. He will be automatically logged into the other applications via single sign-on (SSO).
- Crowd's Administration Console makes it easy for a system administrator to track and manage the username, aliases and application authorisations for each user.
- Crowd's user aliasing allows you to work around the problem that occurs when you want to implement a single user base for a number of existing systems, where users may have different usernames in each system.
- When someone gets married or changes their name, you may wish to rename a user in your LDAP directory, such as Microsoft Active Directory. To avoid problems in applications which do not allow user renaming, you can now link the new LDAP username to an alias in Crowd.
- Some systems may use email addresses as usernames, while in others this may expose users to email spambots. Using Crowd aliasing, you can use different username formats to suit your application requirements.
- Our documentation has the details.



Nested Groups in All Crowd Directories

With Crowd 1.4, we introduced support for nested groups in Crowd-connected LDAP directories. This means that you can have a group as a member of another group. Now Crowd 2.0 supports nested groups for Crowd Internal and Delegated Authentication directories too. Your custom directories will also support nested groups, provided that they meet the interface requirements of the RemoteDirectory API.

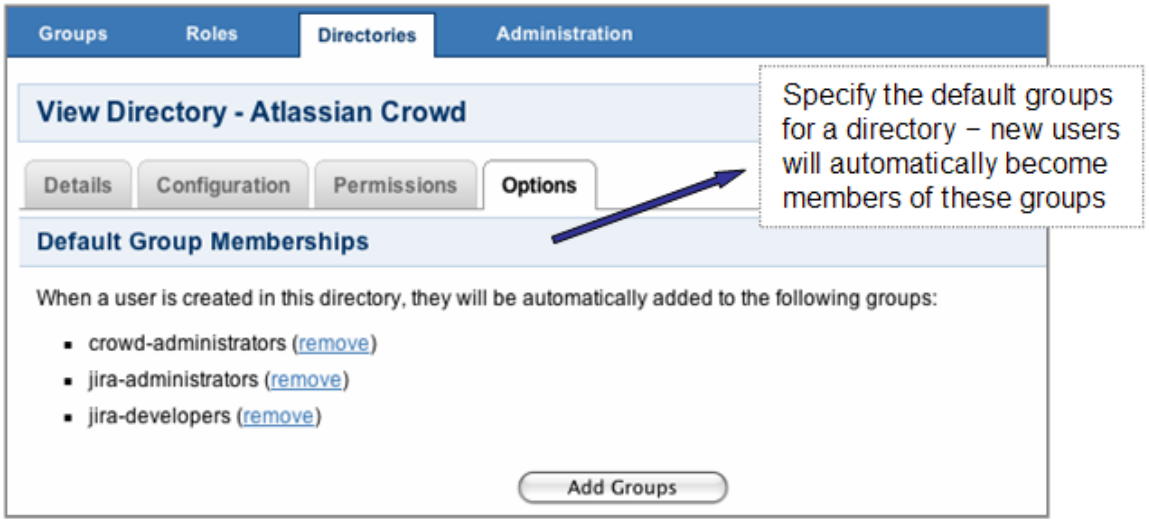
- When verifying a user's login to a Crowd-connected application, Crowd will search the groups mapped to the application plus all their sub-groups.
- When an application requests a list of users in a group, Crowd will present a flat list of users gathered from the requested group and its sub-groups.



Automatic Group Membership for New Users

You can now configure Crowd to assign new users to specific groups automatically.

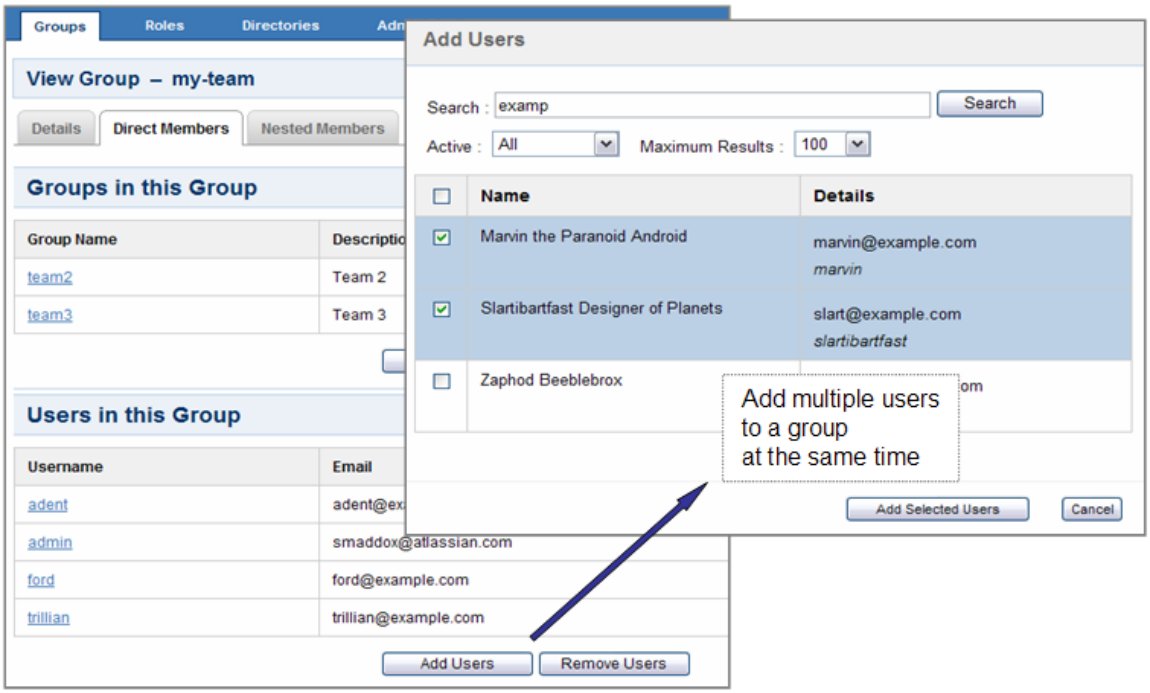
- You can define default groups for each directory.
- A new user automatically becomes a member of these groups, whether added via the Crowd Administration Console or via a Crowd-connected application.
- Note that the automatic group membership does not work when importing users and groups via Crowd's external user importer.
- You can read more in our documentation.



Improved User and Group Management UI

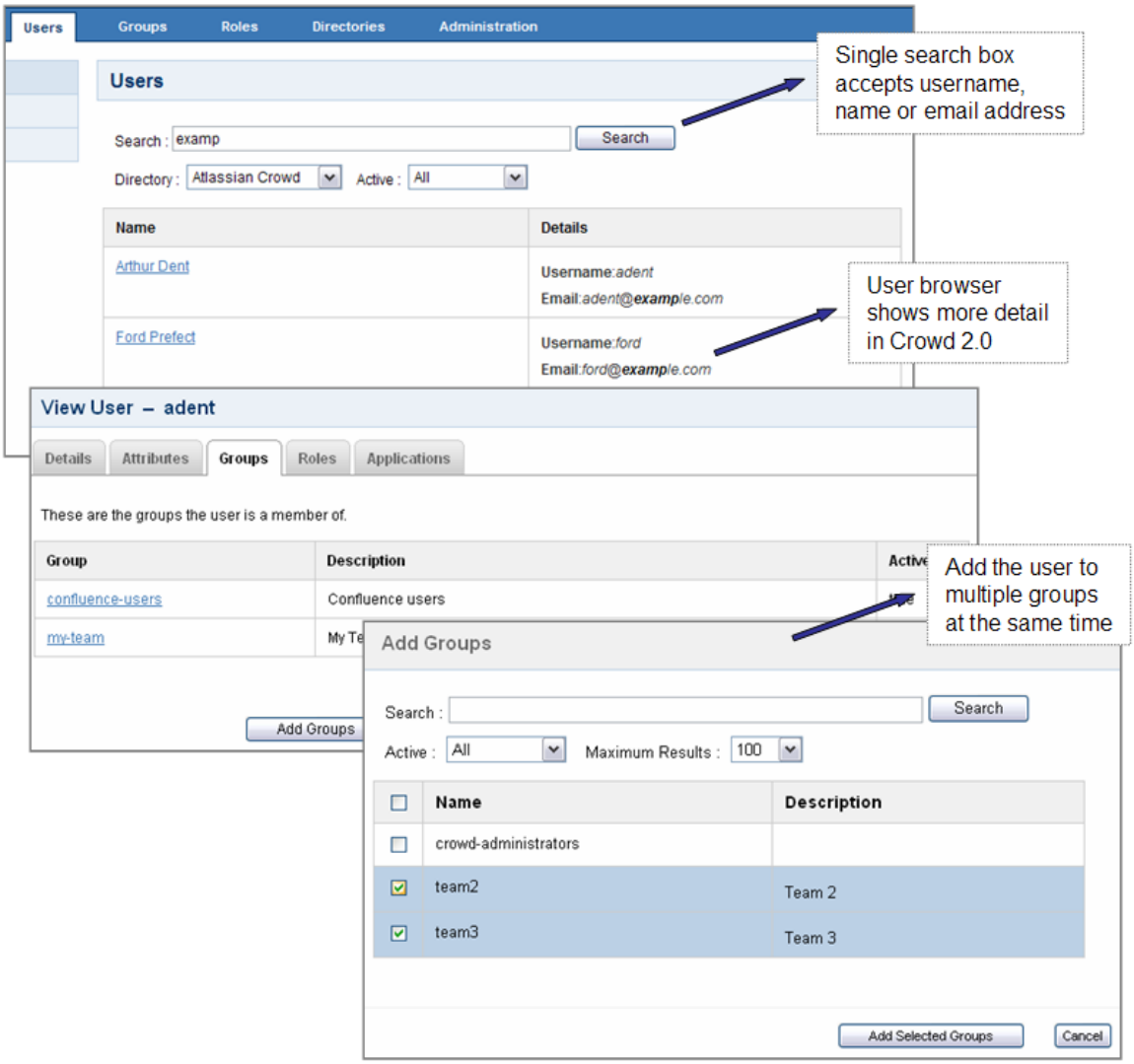
Looking to relieve the administrative pain that user and group management often entail, we have enhanced the management screens in the Crowd Administration Console and added bulk user and group administration for the first time in Crowd.

- You can add multiple users to a group at the same time.



On the user management side:

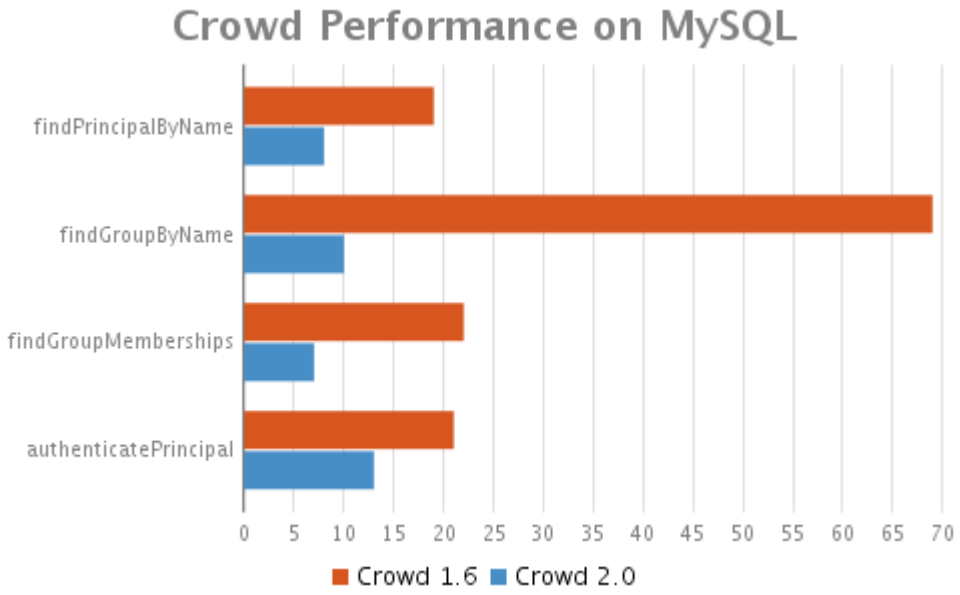
- You can add a user to multiple groups at the same time.
- When searching for a user, just enter all or part of a name, username or email address in a single search box to find the matching users.
- The user browser now shows every user's full name, as well as their usernames and email addresses.

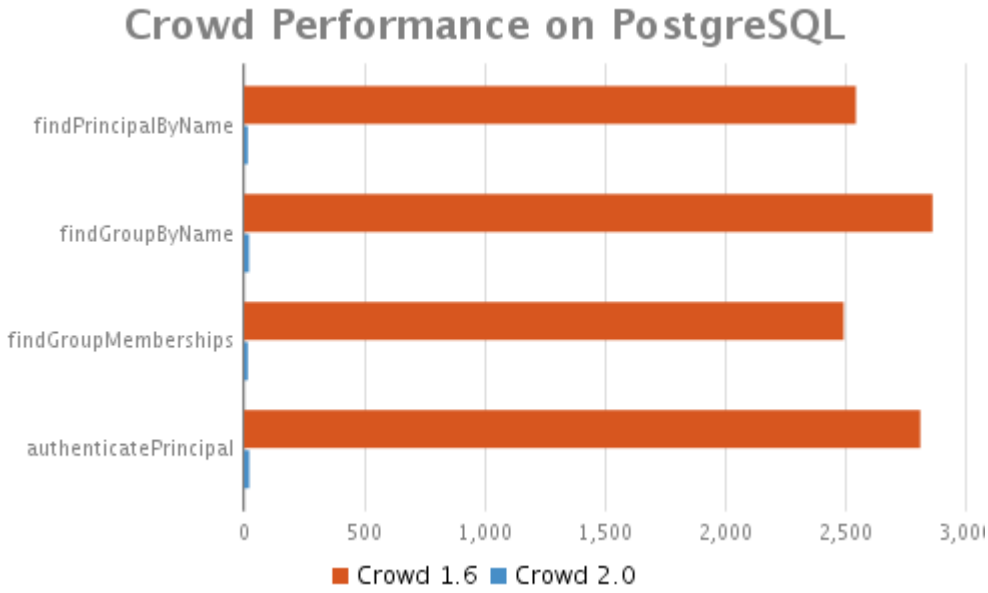


Improved Performance

The Crowd team have done a lot of under-the-cover work in this release, chiefly on updating Crowd's database schema. This work will put us in good stead to provide shiny new features in later releases. For Crowd 2.0, the biggest gain is in the performance of Crowd Internal and Delegated Authentication directories. Comparisons of Crowd 2.0 with the previous release have generated the following statistics in our test environment, running on a Crowd Internal directory with 60000 users, 5000 groups and 240000 group memberships.

- Most operations are about twice as fast.
- Retrieving all users is a gigantic 15 times faster. This request is used when an application asks for all users at once, such as when JIRA's cache expires.
- Searching on fields such as name and email address is more than twice as fast.
- Authenticating a user is 60% faster.





We haven't even tried to represent the **`searchPrincipals`** and **`findAllGroupRelationships`** requests graphically, because the performance improvement is off the charts:

- MySQL is 15 times faster.
- PostgreSQL is 100 to 1000 times faster.

Improved Database Support

The updated Crowd database schema provides some wins in the area of database support too.

- UTF-8 character encoding is now supported for MySQL databases. Before this release, Crowd required Latin 1 character encoding.
- The Crowd database schema uses case-insensitive table names, so for people who are using PostgreSQL, there is no longer any need for silly quotes in your SQL queries.
- Crowd's mail template size is no longer limited to 255 characters.

New REST API

Crowd 2.0 exposes a new REST API that provides access to resources (data entities) via URI paths. This is useful for developers wanting to integrate Crowd into their application and for administrators needing to script interactions with the Crowd server.

- To use a REST API, your application will make an HTTP request and parse the response.
- You can request a response format of XML or JSON.
- Your methods will be the standard HTTP methods like GET, PUT, POST and DELETE.
- Because the REST API is based on open standards, you can use any web development language to access the API.
- Our documentation tells you more.

Plugin Framework 2.2 and REST Module

Crowd 2.0 supports version 2.2 of the Atlassian Plugin Framework, the latest plugin framework release to date. Crowd now also bundles the new REST plugin module type. We have used the REST plugin module type to develop the Crowd 2.0 REST APIs mentioned above.

- Developers can use the REST module type to create plugin points easily in Crowd by exposing services and data entities as REST APIs.
- The REST module type also makes it easier to develop cross-application plugins i.e. plugins which work in more than one application, because the module type helps developers to ensure consistency of REST APIs across Atlassian applications.
- There's more in our documentation.

Other Things Worth Mentioning

- You can now use wildcard IP ranges (CIDR notation) when specifying IP restrictions for an application.
- We now offer full support for Tomcat 6.
- We have enhanced the remote directory API to support finer-grained control in searches. The new API is type safe, supports 'AND' and 'OR' queries and allows you to make finer-grained requests based on primary or custom attributes. For example, you might search for users whose favourite colour is 'pink'. The details are in the JavaDocs.

Complete List of Improvements and Fixes

JIRA Issues (109 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1674	Get users error - LDAP		Closed
CWD-1631	Internal Directory group names in 2.x are lower case by design and incompatible with 1.x		Resolved
CWD-1624	Restoring from an XML backup that used in-memory tokens will revert back to database backed tokens		Resolved
CWD-1621	Updating user attributes causes database error		Resolved
CWD-1616	Creating a user via an atlassian-user based applicaiton fails because a password is not supplied on user creation		Resolved
CWD-1610	Fix link in UI text pointing to docs on application "Options" tab		Resolved
CWD-1606	Clicking *View* Session shows a StackTrace		Resolved
CWD-1605	Add help link for directory "Options" tab plus all Delegated Auth "view/update" links		Resolved
CWD-1597	REST "directory" resource returns two levels of "<directories>" element		Resolved
CWD-1596	Updating aliases with a mix of valid/invalid update can cause strange behaviour		Resolved
CWD-1595	Allow to associate many Groups to a User(s) in a single operation		Closed

CWD-1589	The search in the new user picker in group management does not match on name		Resolved
CWD-1586	Help link is wrong after use of the "Add Group" wizard on the User "Groups" tab		Resolved
CWD-1585	Application "Users" tab does not show any users if one directory is unavailable		Resolved
CWD-1584	An LDAP reference that points to an invalid DN throws a fatal exception		Resolved
CWD-1566	RemoteDirectory requires a more advanced search API to replace the current SearchContext approach used in the SecurityServer		Resolved
CWD-1561	Test Trusted Application support with Aliased applications		Resolved
CWD-1560	Update the database schema documentation for Crowd 2.0		Resolved
CWD-1558	Create Crowd 2.0 artifact		Resolved
CWD-1557	The delegated directory does not fire a UserCreatedEvent when a successfully authenticated user is replicated into the local crowd database.		Resolved
CWD-1556	Review REST and finialise work		Resolved
CWD-1555	Textual change on "Direct Members" tab of Group Browser		Resolved
CWD-1545	UI improvement for User page, add a "group picker' similar to the Group pages.		Resolved
CWD-1538	Adding group from JIRA where group exists with a different case fails		Resolved
CWD-1535	Crowd Client Cache is not refreshed when a Group is deleted using JIRA Admin console		Resolved
CWD-1533	Test Crowd 2.0 integration with JIRA/Confluence/Bamboo/FishEye		Resolved
CWD-1532	Build closed beta of Crowd 2.0		Resolved
CWD-1530	Legacy user/group/membership import needs to be batched.		Resolved
CWD-1529	SecurityServerClient does not correctly segregate roles and groups for container searches.		Resolved
CWD-1528	Verify all code from trunk post making the 2.0 branch is migrated to branch.		Resolved
CWD-1527	Run performance tests against the current 2.0 spike version		Closed
CWD-1526	Fix Crowd PluginPropertyManager and sal-crowd-plugin		Resolved
CWD-1525	UI Improvements for Group Membership Management		Resolved
CWD-1524	Search by Alias and other User attributes		Resolved
CWD-1523	XML Migration for Alias Information		Resolved
CWD-1522	Alias Object Model + Hibenate DAO		Resolved
CWD-1521	Implementation of the AliasService/Manager		Resolved

CWD-1519	Test Crowd on All Supported Databases		Resolved
CWD-1517	Add Role selection to LDAP queries and updates		Resolved
CWD-1516	On import check for any Group & Role name clashes		Resolved
CWD-1514	Configuration Errors need to be displayed for an LDAP directory if Roles are enabled and the DN's for both Groups and Roles overlap.		Resolved
CWD-1507	Crowd Schema + Domain Model update to improve performance and cross-database compliance		Resolved
CWD-1505	User/Group/Membership Import fails when using MySQL		Resolved
CWD-1503	Installation Wizard last step "fails"		Resolved
CWD-1498	Delegated directory attributes only accessible via 'view' link, not by clicking on directory name		Resolved
CWD-1497	Change text "In-Active" to "Inactive" in dropdown lists for user and group status		Resolved
CWD-1493	Performance issue when amalgamating groups for a findAllGroupRelationships call		Resolved
CWD-1491	com.atlassian.crowd.console.filter.CrowdGzipFilterIntegration.useGzip hits database on every invocation		Resolved
CWD-1488	Update SAL to 2.0 to enable REST interfaces		Resolved
CWD-1487	Amalgamation is broken thanks to equals/hashcode using directoryId on directory entities - maybe we need application entities		Resolved
CWD-1480	Upgrade Crowd to atlassian-core 4.2		Resolved
CWD-1479	Upgrade Crowd to Plugins 2.2.0.rc2		Resolved
CWD-1477	Implement a PluginPersistentStateStore for Crowd that isn't an in-memory one. This will need to be database backed.		Resolved
CWD-1476	Allow the Crowd admin to know when a proxy should be added to the Trusted Proxy list		Resolved
CWD-1472	ClientPropertiesImpl.generateBaseURL() assumes that server URL contains /services		Resolved
CWD-1470	Re-enable by-email search tests when new schema lands on trunk		Resolved
CWD-1468	Add alias information to user UI		Resolved
CWD-1467	Highlight application-specific alias when searching in the context of that application		Resolved
CWD-1464	Documentation link for new Users screen in Application and other help links		Resolved
CWD-1460	Remove "If you have set the SSO Domain..." bullet point		Resolved
CWD-1459	Group/User memberships do not obey the tree scope or object filters		Resolved
CWD-1458	Added crosses for removing group in Add Application Wizard		Resolved
CWD-1457	Removing expired tokens from the database token repository requires all tokens to be loaded into memory		Resolved

CWD-1448	Test buttons for directory pages		Resolved
CWD-1446	Disable roles by default on newly created LDAP directories for 2.0		Resolved
CWD-1443	Upgrade Crowd to Plugins 2.2.0		Resolved
CWD-1441	Wrong license user count when users still members of an application group		Resolved
CWD-1435	Google Apps SSO with Crowd results in Bad Request Error during authentication for IE7		Resolved
CWD-1419	Directory Encryption Type is not available for generic Posix or OpenLDAP Posix directories		Resolved
CWD-1411	Make Crowd database schema lowercase		Resolved
CWD-1409	crowd-integration-saml plugin bundles too many jar files		Resolved
CWD-1408	Provide api to access currently logged in user		Resolved
CWD-1406	security filter should be added to path "/plugins/servlet" in web.xml		Resolved
CWD-1405	pluginManager and pluginEventManager beans should be available to plugins		Resolved
CWD-1399	Re-add MYSQL + UTF-8 documentation to mysql.properties		Resolved
CWD-1398	Content-Encoding is unset for SOAP requests		Resolved
CWD-1390	Provide user feedback if SSO Domain setting is preventing users from logging in		Resolved
CWD-1384	Please update Crowd's Evaluation Expiry message		Resolved
CWD-1374	JavaScript error in the Add Application Wizard		Resolved
CWD-1373	Improve UI for removing groups in the Add Application Wizard		Resolved
CWD-1372	Crowd creates new tokens for applications and users even if valid ones already exist		Resolved
CWD-1370	CSV importer fails with '|' used as separator		Resolved
CWD-1357	Remote Addresses not added when enter pressed		Resolved
CWD-1337	Provide support for OS X Open Directory 10.5.6		Resolved
CWD-1327	NullPointerException when using "Reset Password" function		Resolved
CWD-1309	REST API for Crowd		Resolved
CWD-1293	toLower when importing mixedCase usernames from LDAP into a Crowd internal directory.		Resolved
CWD-1292	Officially support Tomcat 6		Resolved
CWD-1187	Nested groups do not work with JIRA Global Permissions		Resolved

CWD-1180	Retain Test Connection & Search after adding Directory		Resolved
CWD-1030	Investigate ability to add account aliases for "change username" capability for Atlassian apps.		Closed
CWD-996	Check if user is active before counting against license		Resolved
CWD-991	Need better user/group managment UI that included ability to bulk add users to groups (like JIRA)		Resolved
CWD-990	UTF-8 support for MySQL		Resolved
CWD-980	Add Nested Groups for Internal Directories		Resolved
CWD-919	Place a "Test Search" button on the Delegated Directory Configuration tab and also on the Configuration tab when viewing a directory		Resolved
CWD-879	Allow admin to designate local Crowd groups for auto-assignment on creation/import of users.		Resolved
CWD-770	Automated adding of users to groups/roles		Resolved
CWD-732	Crowd client should pass version, configuration information to server		Resolved
CWD-635	Edit members of the group or role		Resolved
CWD-605	Bulk change of principals		Resolved
CWD-386	SQL error while importing users from Jira and Confluence to Crowd 1.1 with MSSQL 2000		Closed
CWD-310	Mail Template size is limited to 255 characters		Resolved
CWD-174	Add wildcard support for application IP restrictions.		Resolved
CWD-147	Table names are too long for MySQL with UTF-8		Resolved
CWD-133	Move away from Hibernate `` (ticks) -- Postgres requites double quotes		Resolved
CWD-84	Allow specifying network addresses by netblock		Resolved
CWD-76	Aliases needed for legacy integration		Resolved
CWD-33	Improve searching attributes on a principal.		Resolved

Crowd 2.0 Beta Release Notes

1 June 2009

Crowd 2.0 will be launched in June/July 2009. A beta release is currently undergoing internal testing and is also available to a limited number of customers for review. These release notes apply to **Crowd 2.0 Beta**. We'll publish the final release notes when we release the production-ready version of Crowd 2.0.

The beta release does not yet contain all the features that will be in the final Crowd 2.0 release.

If you would like to participate in testing the beta release, please contact Crowd Support.

Do not use a beta release on production servers

- Beta releases are not safe. A beta release is a snapshot of the ongoing Crowd development process. While we try to keep these releases stable, they have not undergone the same degree of testing as a full release.
- Features in beta releases may be incomplete, or may change or be removed before the next full release.
- Because beta releases represent work in progress, we cannot provide a supported upgrade path between beta releases, or from any beta to the eventual final release. Therefore it is possible that you will not be able to migrate data stored in a Crowd beta release to a future Crowd release.

What's New in Crowd 2.0 Beta

Updated Database Schema

We have spent a lot of time refactoring the database layer of Crowd for 2.0. In particular, you should notice:

- Improved speed and efficiency, especially when you are using an internal directory.
- Support for case-insensitive searching. LDAP supports this feature natively, but now it is also available when you are using a Crowd internal directory.
- UTF-8 character encoding for MySQL databases. Before this release, Crowd required Latin 1 character encoding.
- Many other long-outstanding database issues in Crowd.

Nested Groups in Internal Directories

Crowd now supports nested groups in internal directories, a feature that many people have requested.

Easier Management of Group Memberships

We have improved Crowd's user interface for managing users and groups.

- You can add many users to a group at the same time, via the group management screen.
- With the new user picker, you can find the required user(s) quickly by entering all or part of the user's name, email address or username.

Wildcard Support in Application IP Restrictions

Crowd now supports the use of netblocks for an application's remote address. This means you can specify a complete IP range for an application instead of individual addresses.

- Use CIDR notation. For example: `192.168.10.1/16`
- Wikipedia has a good summary.

Early Adopter's Guide to Reviewing Crowd 2.0 Beta

Upgrading to Crowd 2.0 Beta

Because of the database schema changes, you will need to:

- Export your existing Crowd database to XML: From the Administration Console, select '**Administration**', '**Backup**'. See the instructions.
- Install Crowd 2.0 Beta, following the installation instructions. Please ensure that when starting up Crowd you point Crowd to a **new** crowd home directory, please do not use your current crowd home.
- Select '**Import data from an XML Backup**' when running the Setup Wizard, as described in the setup instructions.

Targets for your Testing

We would love to have your feedback on this beta release, and in particular on the following aspects of the release:

- Support for nested groups in internal directories.
- The group and user pickers on the group management screen.
- Performance comparisons, particularly when using a Crowd internal directory.

Updates and Fixes in this Release

New Features	
CWD-174	Add wildcard support for application IP restrictions.
CWD-635	Edit members of the group or role
CWD-980	Add Nested Groups for Internal Directories
CWD-1337	Provide support for OS X Open Directory 10.5.6
Improvements	
CWD-84	Allow specifying network addresses by netblock
CWD-310	Mail Template size is limted to 255 characters
CWD-732	Crowd client should pass version, configuration information to server
CWD-990	UTF-8 support for MySQL
CWD-1405	pluginManager and pluginEventManager beans should be available to plugins
CWD-1406	security filter should be added to path "/plugins/servlet" in web.xml
CWD-1446	Disable roles by default on newly created LDAP directories for 2.0
CWD-1472	ClientPropertiesImpl.generateBaseURL() assumes that server URL contains /services
CWD-1476	Allow the Crowd admin to know when a proxy should be added to the Trusted Proxy list
CWD-1507	Crowd Schema + Domain Model update to improve performance and cross-database compliance
CWD-1525	UI Improvements for Group Membership Management
Bug Fixes	
CWD-1187	Nested groups do not work with JIRA Global Permissions
CWD-1372	Crowd creates new tokens for applications and users even if valid ones already exist
CWD-1398	Content-Encoding is unset for SOAP requests
CWD-1411	Make Crowd database schema lowercase
CWD-1419	Directory Encryption Type is not available for generic Posix or OpenLDAP Posix directories
CWD-1441	Wrong license user count when users still members of an application group
CWD-1459	Group/User memberships do not obey the tree scope or object filters
CWD-1493	Performance issue when amalgamating groups for a findAllGroupRelationships call
CWD-1498	Delegated directory attributes only accessible via 'view' link, not by clicking on directory name
CWD-1512	The runtime environment of Crowd will not allow Roles and Caching to be enabled at the same time
CWD-1514	Configuration Errors need to be displayed for an LDAP directory if Roles are enabled and the DN's for both Groups and Roles overlap.
CWD-1529	SecurityServerClient does not correctly segregate roles and groups for container searches.

Crowd 1.6.1 Release Notes

17 February 2009

The Atlassian Crowd team is delighted to present **Crowd 1.6.1**.

This release focuses on solving problems with case sensitivity. Crowd's internal directories, client caches and LDAP directory caches are now all case insensitive but case preserving. Crowd will ignore case when comparing usernames, etc ('JSmith' = 'jsmith') and it will preserve case when passing information between applications and directories ('JSmith' remains 'JSmith'). This results in the expected behaviour in the Crowd-connected directories as well as Crowd-connected applications such as JIRA and Confluence.

In addition, Crowd now allows you to enforce lower-case conversion of usernames, groups and roles for a specific application. Where is this useful? Let's assume you have previously integrated JIRA with an LDAP directory that allows mixed-case usernames (e.g. 'JSmith'). JIRA enforces lower-case usernames (e.g. 'jsmith'), so you have existing lower-case usernames in JIRA. And now you want to integrate JIRA with Crowd. You can configure Crowd to convert all usernames, etc, to lower case before passing them to JIRA.

We have also fixed a few bugs, including a problem with finding group members in Posix directories and a problem with Gzip compression for SOAP requests.

Don't have Crowd 1.6 yet?

Take a look at the new features and other highlights in the Crowd 1.6 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.6.1

JIRA Issues (17 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1424	Add help link to help-paths.properties for new application tab		Resolved
CWD-1421	UI text in Crowd 1.6.1		Resolved
CWD-1420	findGroupRelationships() is broken for Posix directories		Resolved
CWD-1419	Directory Encryption Type is not available for generic Posix or OpenLDAP Posix directories		Resolved
CWD-1404	Make Crowd Client impersonate a more modern browser (User-Agent header)		Resolved
CWD-1398	Content-Encoding is unset for SOAP requests		Resolved
CWD-1396	Make the InternalDirectory case-insensitive		Resolved
CWD-1395	Make the client caches case-insensitive		Resolved
CWD-1394	Make the DirectoryCache case-insensitive		Resolved
CWD-1382	Selection of 'Enable Caching' does not immediately show the additional config options in IE		Resolved
CWD-1378	Setup Wizard for MySQL connection doesn't specify characterEncoding		Resolved
CWD-1377	Relaxed DN Standardisation option should appear only for caching-enabled directories		Resolved
CWD-1252	Username case matters in JIRA if you're using Crowd 1.4.x or 1.5, it didn't used to in Crowd 1.3.x		Resolved
CWD-1118	Allow Crowd admin to specify property to enforce toLower on username for JIRA/Confluence integration		Resolved
CWD-781	Support case-insensitivity for the Internal Directory, this is in the aim of providing support for RFC-2798		Resolved
CWD-732	Crowd client should pass version, configuration information to server		Resolved
CWD-140	Change over adding an application to be a setup wizard.		Resolved

Crowd 1.6 Release Notes

18 December 2008

The Atlassian Crowd team is proud to present **Crowd 1.6**.

Crowd 1.6 introduces a new, more intelligent caching system that will improve performance of Crowd with LDAP, particularly for large and off-site directories.

This release also brings a quicker setup process for Atlassian applications. The Crowd Administration Console allows you to choose the application you want to integrate (JIRA, Confluence, Bamboo, FishEye or Crucible), prompts you for the necessary information and automatically adds the required directory and groups.

There are new directory connectors for OpenDS, Fedora Directory Server and OpenLDAP (based on the Posix/NIS schema).

You'll find a number of smaller improvements in this release too. More unusual characters are supported in the UI and in LDAP directories. Using Crowd's new authentication-related API events, you can create plugins that react when a user logs in, logs out, changes their password, and so on.

Highlights of this release:

- Smarter Caching
- Quick Application Setup
- Connectors for OpenDS, Fedora DS and OpenLDAP (Posix)
- Spring Security 2
- Other Good Things
- Complete List of Improvements and Fixes

Responding to your feedback:

38 votes satisfied
Keep logging your votes and issues. They help us decide what needs doing!

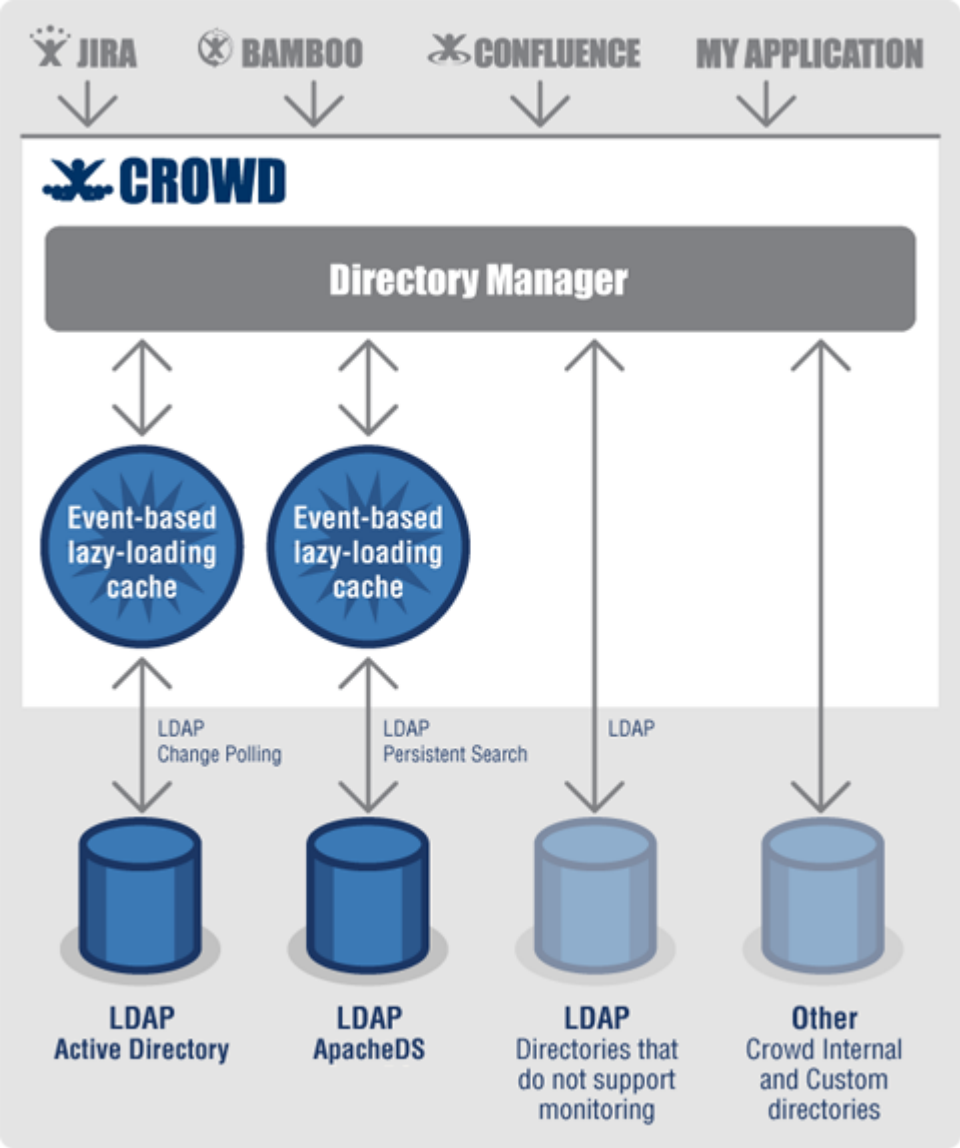
Upgrading to Crowd 1.6
You can download Crowd from the Atlassian website. If upgrading from a previous version, please read the Crowd 1.6 Upgrade Notes.

Highlights of Crowd 1.6

Smarter Caching

Crowd 1.6 introduces a new, more intelligent caching system that will improve performance of Crowd with Microsoft Active Directory and ApacheDS. You should notice the improvement particularly in directories which are large, slow or off site.

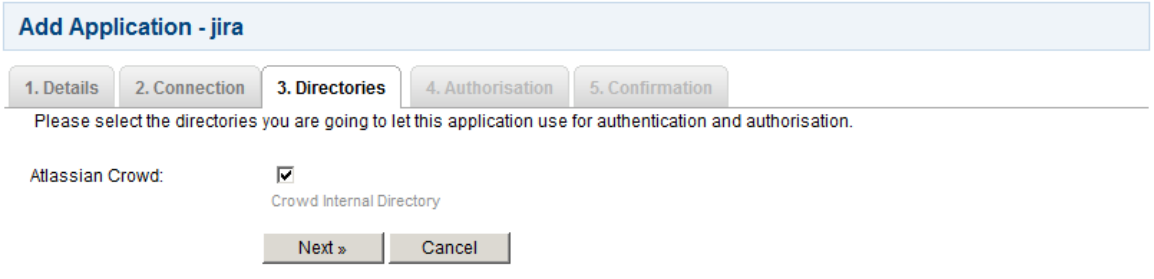
- Crowd now keeps an up-to-date cache of user, group and role information retrieved from the LDAP directory.
- The cache uses lazy loading where possible, storing only the information that is required rather than loading the entire directory into the cache.
- Crowd ensures that the cache remains up to date by monitoring the LDAP directory for updates. When a change occurs, Crowd updates the server-side cache incrementally.
- Refer to our documentation for an overview of Crowd caching and details of the LDAP caching.



Quick Application Setup

Crowd 1.6 brings a quicker setup process for Atlassian applications. Crowd now supports specific application types for JIRA, Confluence, Bamboo, FishEye and Crucible.

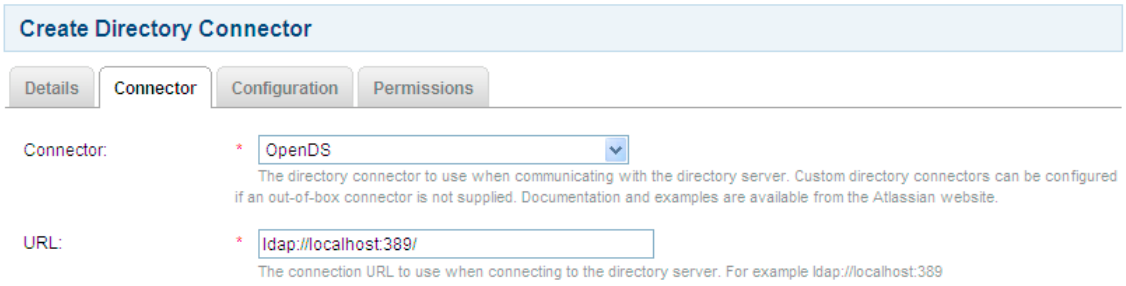
- The Crowd Administration Console allows you to choose the type of application you want to integrate and prompts you for the necessary information.
- Crowd automatically adds the required directory and groups. For example, if you are integrating Crowd with JIRA, Crowd will add the 'jira-users', 'jira-developers' and 'jira-administrators' groups for you.
- The setup process will prompt you to import the users from JIRA or the relevant application.
- Then you can move quickly to the next stage, configuring the application's libraries and other settings, which is still a manual process.



Connectors for OpenDS, Fedora DS and OpenLDAP (Posix)

Crowd 1.6 provides three new built-in directory connectors. The new connectors do not affect any directories already configured. They will make it easier to set up your directory if you are starting from scratch.

- OpenDS.
- Fedora Directory Server and OpenLDAP, based on the Posix/NIS schema.



Spring Security 2

- We've updated Crowd to use and support Spring Security 2. See our tutorials on how to set it up, or to use it with the latest version of Appfuse.

Other Good Things

- In Crowd 1.5, we introduced an early version of the Atlassian Plugin Framework 2. Crowd 1.6 now supports version 2.1 of the Atlassian Plugin Framework.
- Crowd now fires a number of API events related to authentication and change of password. This allows developers to create listener plugins that spring into action when a user logs in, logs out, changes their password, and so on.

Complete List of Improvements and Fixes

JIRA Issues (37 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1368	Crowd client not properly locating crowd-ehcache.xml causing caching not to occur		Resolved
CWD-1360	AppTypes: Wording Suggestions		Resolved
CWD-1346	Implement server-side remote directory caching		Resolved
CWD-1345	Properly find the Deleted Objects container if the baseDN is not the root of the AD domain		Resolved
CWD-1344	Implement "flush cache" button for event caches.		Resolved
CWD-1343	Implement role disable checkbox for caching directories to avoid object duplication		Resolved
CWD-1342	Spring-ldap 1.3-RC1 changed the way authentication happens with Open Directory - maintain compatibility		Resolved
CWD-1341	Configure redirection of context-sensitive online help links for existing 1.5 release		

			Resolved
CWD-1339	If baseDN is not the root of the tree, deleted objects detection does not work		Resolved
CWD-1336	Update Crowd to Plugins 2.1.2		Resolved
CWD-1332	Directory connector dropdown should default to Microsoft Active Directory		Resolved
CWD-1328	Re-word the label and description for the 'has access' cache checkbox on the Admin Console		Resolved
CWD-1325	Add UI options for new directory types and clean up the descriptions		Resolved
CWD-1323	Possible Bug in Token Random Numbers		Resolved
CWD-1314	Update MySQL Hibernate dialect to create transactional InnoDB tables by default		Resolved
CWD-1306	Upgrade Crowd to plugins 2.1		Resolved
CWD-1304	Change DirectoryEntity.compareTo() to correctly compare subclasses		Resolved
CWD-1301	Can no longer change Delegated Directory User Configuration. Changes do not save.		Resolved
CWD-1279	SafeParametersInterceptor has broken the AtlassianImporter		Resolved
CWD-1275	handles funtion in CrowdCredentialsProvider throws exception with NULL parameter		Resolved
CWD-1273	SecurityServer.authenticatePrincipalSimple() overwrites InvalidAuthenticationException text with an incorrect message.		Resolved
CWD-1264	Remove user link does not work for usernames with plus sign (+)		Resolved
CWD-1262	Speed up OpenLDAP user listings using memberOf group membership attribute		Resolved
CWD-1206	Throw IllegalArgumentException in CrowdCredentialProvider's changePassword to throw a friendlier 500 page exception (JRA-13685)		Resolved
CWD-1205	Crowd profiling label always displays 'off' after logging levels are updated		Resolved
CWD-1194	Support SUN OpenDS LDAP Server		Resolved
CWD-1170	Cannot delete Group when name contains a +		Resolved
CWD-1159	BadLdapGrammarException occurs when Crowd attempts to read a DN		Resolved
CWD-1119	Unhelpful Exception with resetPrincipalCredential() when SMTP Server rejects email address		Resolved
CWD-1114	Groups can be created with '&' in name but cannot subsequently be deleted		Resolved
CWD-1099	Unable to create username with a plus (+) character with Open LDAP		Resolved
CWD-1069	Groups that contain backlashes ('\') cannot be modified from Crowd		Resolved
CWD-1048	Create FedoraDS connector class		Resolved
CWD-844	Upgrade Acegi integration libraries to Spring Security 2.0		

			Resolved
CWD-779	Sun OpenDS is not supported		Resolved
CWD-772	Crowd client libraries and caching need further review to improve JIRA performance		Resolved
CWD-562	Cannot create group in ApacheDS 1.5.1		Resolved

Crowd 1.5.2 Release Notes

31 October 2008
The Atlassian Crowd team is delighted to present **Crowd 1.5.2**.

This release fixes the import of users from JIRA or other Atlassian products, which was broken in Crowd 1.5.1.

When configuring an LDAP directory connector, you can now enable or disable the use of the group membership attribute on the user, for group membership searches. By default, this option will be disabled. If your directory supports 'memberOf' or another group membership attribute on the user, then you should enable the option to speed up your group membership queries.

Don't have Crowd 1.5 yet?
Take a look at the new features and other highlights in the Crowd 1.5 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.5.2

JIRA Issues (6 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1282	User import from JIRA is impossible		Resolved
CWD-1281	Document the two "use User Membership Attribute" options		Resolved
CWD-1279	SafeParametersInterceptor has broken the AtlassianImporter		Resolved
CWD-1262	Speed up OpenLDAP user listings using memberOf group membership attribute		Resolved
CWD-1216	Aggressive caching in CachingGroupManager causes performance problems		Resolved
CWD-1148	Not all AD configurations use memberOf attribute. Need to provide toggle for this in Crowd.		Resolved

Crowd 1.5.1 Release Notes

14 October 2008
The Atlassian Crowd team is delighted to present **Crowd 1.5.1**.

Crowd 1.5.1 is a recommended upgrade which fixes a parameter injection vulnerability and other issues. Please refer to the security advisory for details of the security vulnerability, risk assessment and mitigation strategies.

When using Crowd for single sign-on (SSO), you can now specify that the 'secure' flag is set on the SSO cookie. This will enforce a secured connection, such as SSL, for all SSO requests. Note that if you set this flag, any applications not using a secure connection will not be able to participate in SSO. Potentially, this may make it impossible to log in to Crowd.

When generating session tokens, Crowd now includes a very large random number as part of the hash value. This makes it more difficult for a malicious third party to impersonate a legitimate Crowd user.

This release also brings a number of improvements to search functionality, particularly for LDAP directories and for Confluence instances integrated with Crowd.

Don't have Crowd 1.5 yet?
Take a look at the new features and other highlights in the Crowd 1.5 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.5.1

JIRA Issues (22 issues)			
Key	**Summary**	**Priority**	**Status**

CWD-1276	Create how-to documentation for language JARs for Crowd		Resolved
CWD-1268	Make XWork ParametersInterceptor safe from parameter injection attacks		Resolved
CWD-1254	Latest version of Appfuse not working with Crowd's Acegi/Appfuse Tutorial		Resolved
CWD-1251	On startup Crowd displays an EHCache error about duplicate disk store paths & no configuration for Property		Resolved
CWD-1249	Updating a RemotePrincipal does not add new attributes to LDAP		Resolved
CWD-1245	Full name searches return all users if the underlying Crowd directory is LDAP-based		Resolved
CWD-1244	crowd-ehcache.xml in Crowd's client directory does not contain defaultCache value		Resolved
CWD-1242	Crowd dependency check on startup		Resolved
CWD-1201	Group search requires exact case		Resolved
CWD-1199	In-memory token storage will not permit expiration of user session, throws exception		Resolved
CWD-1190	OS User fullname and email updates are not reflected in cache		Resolved
CWD-1156	Crowd Search API currently allows searches for PRINCIPAL_FULLNAME on Crowd internal directories, not LDAP		Resolved
CWD-1134	Removing user from Crowd does not remove tokens from TOKEN table for this user.		Resolved
CWD-1110	CrowdEntityQueryParser doesn't search groups by wildcards		Resolved
CWD-1040	Crowd session tokens need to be random and unique to avoid Session Hijacking!!!		Resolved
CWD-1039	Change Hibernate dialect for Oracle 9i/10g to use Oracle9iDialect		Closed
CWD-994	Multiple field query in Confluence user manager throws exception		Resolved
CWD-960	Increase ATTRIBUTEVALUES.VALUE size from 255 for more complex object filters		Resolved
CWD-912	Internal Directory and LDAP searches behave differently		Resolved
CWD-893	Option to set secure flag on SSO cookie		Resolved
CWD-701	Need to ensure that special characters are escaped properly for UTF-8 in XML backup.		Closed
CWD-156	When adding a user from a Crowded Confluence, the email address and full name are missed		Resolved

Crowd 1.5 Release Notes

4 September 2008

The Atlassian Crowd team is proud to present **Crowd 1.5**.

Crowd now supports single sign-on (SSO) to Google Apps. Do you use Google Apps for your office documentation, calendar and collaboration tools? Using Crowd's SSO, your users can log in once then move seamlessly between Google Apps and other Crowd-integrated applications like JIRA, Confluence, Jive Forums and others.

Crowd 1.5 has a new directory connector, supporting read-only connections to Apple's OS X Open Directory server.

Developers will be interested in Atlassian's new Plugin Framework, now supported in Crowd 1.5. The new Google Apps connector,

implemented as a plugin, provides a useful example for developers wanting to extend Crowd's functionality by building a Crowd plugin.

CrowdID has been updated to the latest OpenID 2.0 specification. CrowdID, shipped with Crowd, allows your corporation to act as OpenID provider for your employees.

This release brings many improvements and fixes, including much faster user imports and database imports, JNDI mail configuration and a cleaner upgrade process.

Highlights of this release:

- Single Sign-On to Google Apps
- Connector for Apple Open Directory
- Plugin Framework 2.0 and API
- Other Improvements and Bug-Fixes
- Complete List of Improvements and Fixes

Responding to your feedback:

45 votes satisfied

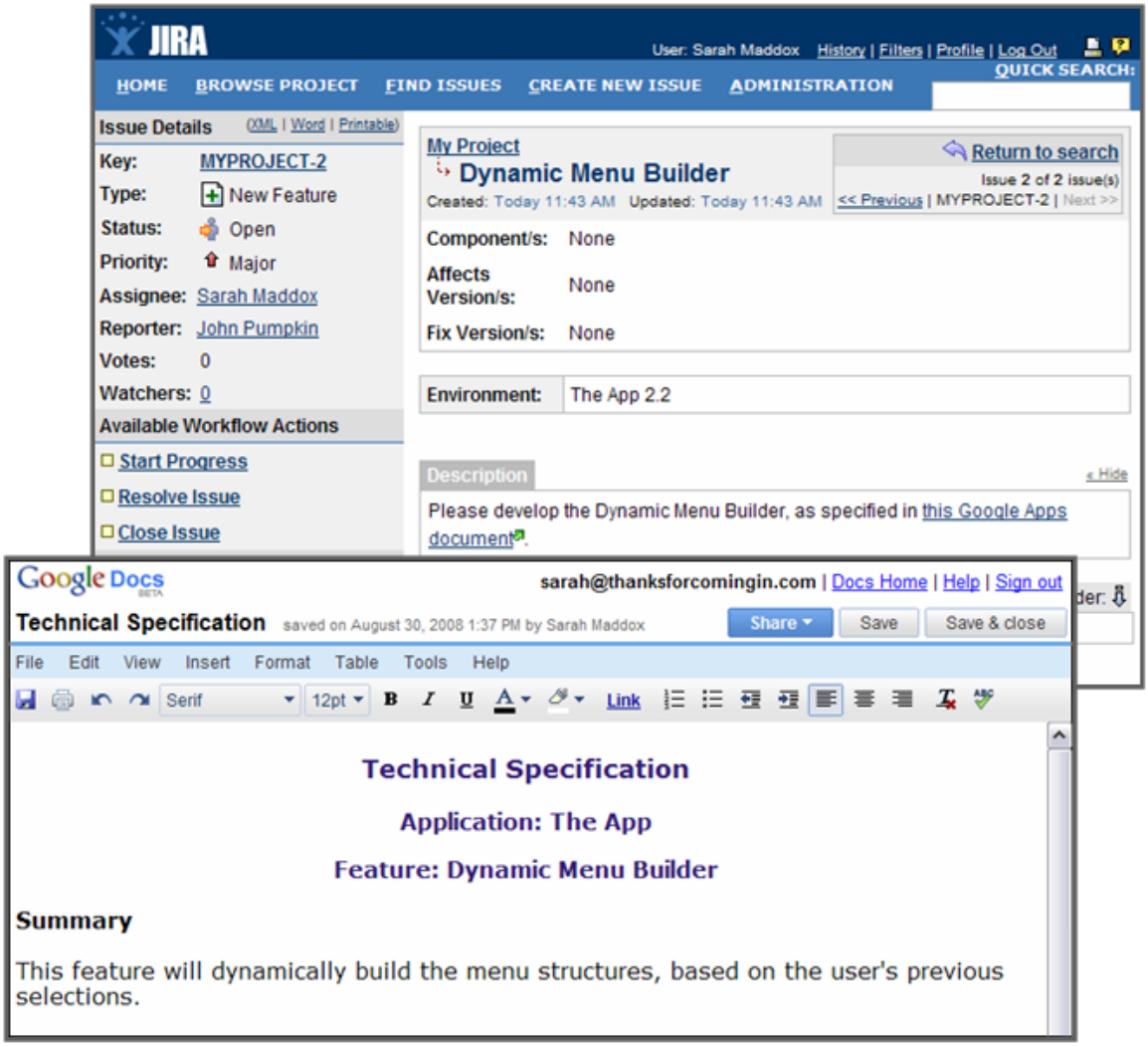
Keep logging your votes and issues. They help us decide what needs doing!

Upgrading to Crowd 1.5
You can download Crowd from the Atlassian website. If upgrading from a previous version, please read the Crowd 1.5 Upgrade Notes.

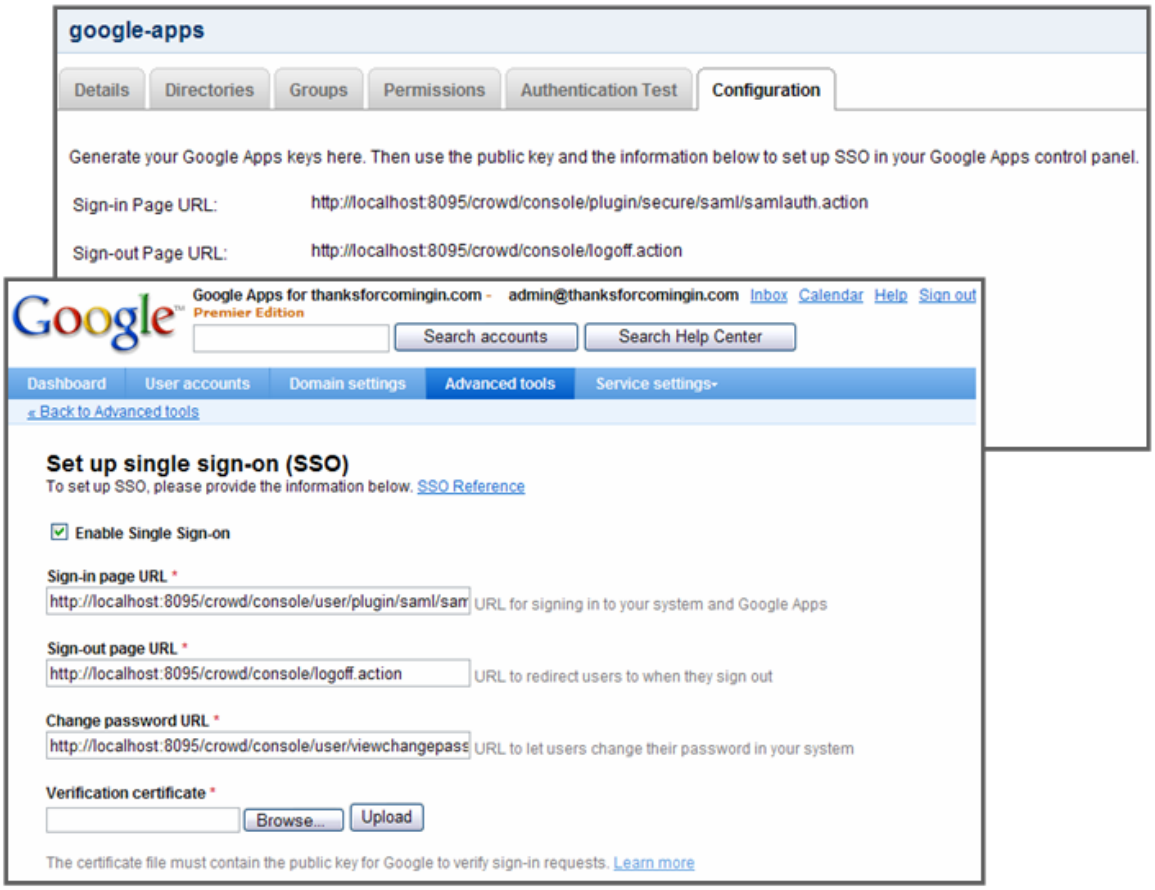
Highlights of Crowd 1.5

Single Sign-On to Google Apps

- Crowd now supports single sign-on (SSO) to Google Apps.
- Users can log in to Google Apps using their corporate username and password.
- An example of Google Apps SSO in action: A user clicks through from a link in a JIRA issue. The document opens directly in Google Apps. No need to log in again, no need to remember a different password.

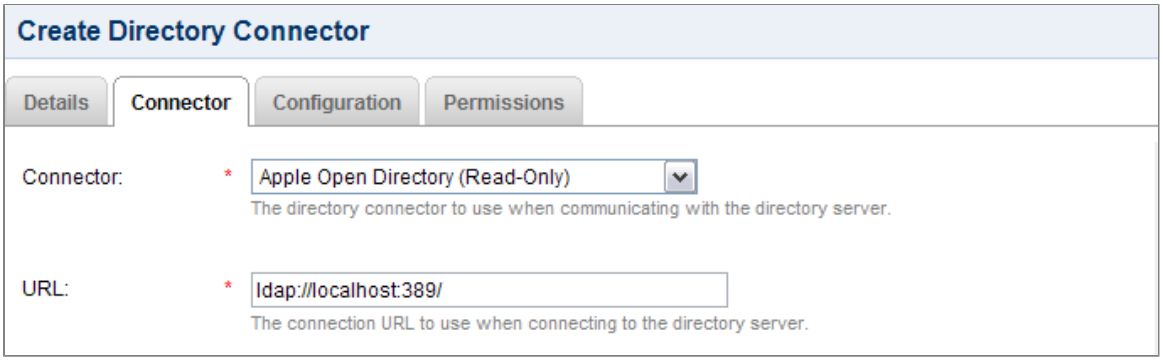


- Administrators can use Crowd's groups to authorise access to Google Apps.
- Enjoy the security and convenience of managing all your users in one place.
- Set up Google Apps SSO in two easy steps: Generate the keys in Crowd then enter the information in Google Apps.



Connector for Apple Open Directory

- Crowd 1.5 supports read-only connections to Apple OS X Open Directory server.
- Our documentation has the full details.



Plugin Framework 2.0 and API

- Crowd 1.5 comes with Atlassian's new Plugin Framework, based on Spring Dynamic Modules using an embedded OSGi container.
- The new Google Apps connector is implemented as a plugin, using the new Plugin Framework. This provides a useful example for developers wanting to extend Crowd's functionality by building a Crowd plugin.
- The Plugin Framework is experimental at this stage. We'd be delighted to have your feedback via our JIRA project.
- Take a look at our developer documentation, also currently under development.
- Crowd now fires an API event when a create/update/delete operation is performed at directory level. Developers can create listener plugins which spring into action when a specific event occurs. For example, the plugin might do something when a user is created, or when a group is deleted, and so on.

Other Improvements and Bug-Fixes

- When configuring your mail server, you can now choose between SMTP and a JNDI location. This allows you to use an SSL connection to your mail server.
- Importing users into a Crowd directory from Atlassian applications or a CSV file is now much faster when dealing with large user bases.
- Importing Crowd data from an XML backup is also much faster, due to the use of JDBC batching.
- CrowdID has been updated to the latest OpenID 2.0 specification. CrowdID, shipped with Crowd, allows your corporation to act as an OpenID provider for your employees.
- We have moved the `crowd.properties` file for the Crowd Administration Console to the Crowd Home directory, so that upgrading Crowd will be cleaner and easier from now on.
- Crowd will respond to a 'require password change' attribute and force the user to change their password before logging in.

Complete List of Improvements and Fixes

JIRA Issues (45 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1235	Configure redirection of context-sensitive online help links for existing 1.4 release		Resolved
CWD-1227	Please update the help-paths.properties file for Crowd 1.5		Resolved
CWD-1217	Make the license expiration notice less scary		Resolved
CWD-1214	Setting SearchContext.GROUP_POPULATE_MEMBERSHIPS to "none" still retrieves group memberships.		Resolved
CWD-1195	Authentication Token Storage Reverts to Database Cache after Crowd Restart.		Resolved
CWD-1176	XML Imports from 1.0.x versions of Crowd do not contain the ldap.pagedresults.size attribute. This causes an exception in certain cases.		Closed
CWD-1163	Certain LDAP-related errors that generate XFireRuntimeExceptions can actually cause JIRA's comment field to not appear.		Resolved
CWD-1155	Trusted Proxy values also need to be added to application remote address list or application receives "Client host is invalid" error		Resolved
CWD-1120	Administration > Current Sessions > User Sessions(Session Browser) the link to the users associated Directory is using the wrong id.		Resolved
CWD-1109	Cannot browse users or groups if Use Paged Results is enabled on ApacheDS directory		Resolved
CWD-1105	Default Generic LDAP connector attributes for groups are incorrect		Resolved
CWD-1097	Optimize group search algorithm for Confluence/JIRA		Resolved
CWD-1095	RFC2307MemberParser.fetchDirectMembers can return null elements		Resolved
CWD-1089	A LDAP reference that points to a deleted user throws a fatal exception		Resolved
CWD-1080	Removing a group from an LDAP server from a client application (eg Confluence, JIRA) does not work		Resolved
CWD-1079	Authentication of client applications against Crowd fails with NullPointerException		Resolved
CWD-1077	Crowd's client/lib directory appears to have a few too many dependencies in it		Resolved
CWD-1070	Add unicode support for MS SQL Server 2000 & 2005		Resolved

CWD-1063	Under heavy load client libraries will leak sockets into CLOSE_WAIT.		Resolved
CWD-1058	Improve JIRA integration by consolidating the findAllPrincipalNames() call with the individual calls to retrieve users.		Resolved
CWD-1049	The 'crowd' context is hardcoded into the login.jsp page		Resolved
CWD-1043	Java 5 and xfire-java5 are required for crowd-integration-client to honour the http.nonProxyHosts system property		Resolved
CWD-1027	'No config properties were found for importing!' error in logs when importing XML		Resolved
CWD-1026	Set up wizard announces success even when setup failed		Resolved
CWD-1019	When applying new license to a Crowd instance whose users exceed the license, the message should be more explicit than "Invalid License"		Resolved
CWD-982	If requirePasswordChange attribute is true, provide method of forcing user to change password in Crowd user console		Resolved
CWD-973	If LDAP directory becomes unavailable, cannot remove from application in Crowd		Closed
CWD-964	Add actual license to the Licensing screen.		Resolved
CWD-939	Exceptions in the user console are obscured by an UnexpectedRollbackException		Resolved
CWD-938	Decouple ClientProperties and PropertyUtils		Resolved
CWD-848	Using the Importers (CSV, Atlassian Importer, LDAP) will be slow with large datasets.		Resolved
CWD-784	Move crowd.properties outside the crowd-web-app		Resolved
CWD-756	Allow Mail Server configuration through JNDI location (+TLS for connections)		Resolved
CWD-735	Separate the concept of users, groups and memberships		Resolved
CWD-721	Remove manual caching of Server Properties		Resolved
CWD-719	Faster XML import/export for large backups		Resolved
CWD-693	Need to trim Remote Address values for Applications.		Resolved
CWD-613	Apple OpenDirectory connector		Resolved
CWD-541	Allow specification of Trusted Proxy Servers		Resolved
CWD-511	Update Crowd OpenID libraries to be in line with OpenID 2.0 Final		Resolved
CWD-333	Implement SSO for Google Apps		Resolved
CWD-241	OS X Directory Server Connector		Resolved
CWD-217	Crowd throws obscure exception when attempting to add a principal to a non-existent group		Resolved
CWD-183	Problems with LDAP group or user names that contain / or \.		Resolved

CWD-153	Fedora DS		Resolved

Crowd 1.4.7 Release Notes

14 October 2008
Crowd 1.4.7 is a recommended upgrade which fixes a parameter injection vulnerability, as described in the security advisory. Please refer to the advisory for details of the security vulnerability, risk assessment and mitigation strategies.

The latest version of Crowd, at the time of these release notes, is Crowd 1.5.1. The previous public release of Crowd 1.4.x was version 1.4.4. Versions 1.4.5 and 1.4.6 were internal releases. We are supplying version 1.4.7 as an upgrade for versions 1.4.x, to fix the security vulnerability.

Don't have Crowd 1.5 yet?
Take a look at the new features and other highlights in the Crowd 1.5 Release Notes.

Download Latest Version

Crowd 1.4.4 Release Notes

1 July 2008
The Atlassian Crowd team is delighted to present **Crowd 1.4.4**.

You can now enable or disable support for nested groups on each LDAP user directory. If you upgrade your Crowd installation with existing LDAP directory connectors, nested group support will remain enabled for those directories. To configure nested group support for new or existing LDAP connectors, go to the connector configuration screen in your Administration Console.

When using Crowd for single sign-on (SSO), you can now specify the SSO cookie name for each application. Under the standard configuration, Crowd will use a single, default cookie name for all Crowd-connected applications. For more information, read about the crowd.properties file.

Don't have Crowd 1.4 yet?
Take a look at the new features and other highlights in the Crowd 1.4 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.4.4

JIRA Issues (15 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1175	Please add a note about this bug on our import docs		Closed
CWD-1169	Add note to Crowd-Bamboo integration docs on configuring explicit cache to bypass CWD-1167		Closed
CWD-1167	If crowd-ehcache.xml is misconfigured, Bamboo Crowd Integration Client uses wrong default cache		Resolved
CWD-1128	DelegatedAuthenticationDirectory.authenticate returns an LDAP-backed principal		Resolved
CWD-1127	Enable the ability to configure nested group support for a given directory		Resolved
CWD-1125	The directoryID is not populated on principals added automatically after authentication with a delegated directory		Resolved
CWD-1121	CrowdUserDetailsServiceImpl.generateAuthoritiesFromGroupNames throws a NullPointer if no groups are associated		Resolved
CWD-1115	Per-application cookies.		Resolved
CWD-1112	Checking if a user is a member of a group on large user ldap directories is slow and casuses OutOfMemoryException -- FAIL		Resolved
CWD-1111	JIRA Performance with Crowd 1.4.3 and AD is very slow		Resolved

CWD-1074	AD directory connector should check "Use Node Referrals" as the default		Resolved
CWD-1059	Admin Cookie name should be different to application/user cookie name		Resolved
CWD-894	Dont hard code SSO cookie name		Resolved
CWD-690	Add setenv.sh file in the apache-tomcat-5.5.20/bin directory for easier memory management.		Resolved
CWD-125	Provide Lotus Domino Support		Resolved

Crowd 1.4.3 Release Notes

5 June 2008
The Atlassian Crowd team is delighted to present **Crowd 1.4.3**, bringing significant performance improvements in JIRA when integrated with Crowd.

We have optimised the code and modified the caching behaviour in the Crowd client libraries. This will dramatically improve the performance of a JIRA-Crowd integration for large LDAP user directories.

Don't have Crowd 1.4 yet?
Take a look at the new features and other highlights in the Crowd 1.4 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.4.3

JIRA Issues (5 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1097	Optimize group search algorithm for Confluence/JIRA		Resolved
CWD-1095	RFC2307MemberParser.fetchDirectMembers can return null elements		Resolved
CWD-1058	Improve JIRA integration by consolidating the findAllPrincipalNames() call with the individual calls to retrieve users.		Resolved
CWD-937	Confluence crowd-ehcache.xml		Closed
CWD-344	Problems with Licence Key In Some Locales		Closed

Crowd 1.4.2 Release Notes

29 May 2008
The Atlassian Crowd team presents **Crowd 1.4.2**. This release includes some good bug fixes and an improvement to the Spring configuration libraries.

A note for those integrating Crowd with JIRA: If you are using JIRA 3.12.2 or earlier, you will need to update JIRA's xfire libraries as described in the Upgrade Notes.

Don't have Crowd 1.4 yet?
Take a look at the new features and other highlights in the Crowd 1.4 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.4.2

JIRA Issues (7 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1082	crowd-integration-client-1.4.1.jar is incompatible with versions of JIRA earlier than 3.12.3		Closed
	The client spring XML files are missing references to the cacheManagers, since they are in the		

CWD-1081	applicationContext-CrowdSecurity.xml		Resolved
CWD-1080	Removing a group from an LDAP server from a client application (eg Confluence, JIRA) does not work		Resolved
CWD-1079	Authentication of client applications against Crowd fails with NullPointerException		Resolved
CWD-1076	crowd-acegi integration null pointer exception + documentation errors		Resolved
CWD-1062	Full Name can't be changed from Confluence		Resolved
CWD-929	Crowd-Acegi Integration Tutorial may need updating		Resolved

Crowd 1.4.1 Release Notes

23 May 2008
The Atlassian Crowd team is delighted to present **Crowd 1.4.1**. This release includes a few bug fixes and a new feature — trusted proxy servers.

If you are running applications behind one or more proxy servers, you may find it useful to configure Crowd to trust the proxies' IP addresses. When a proxy server forwards an HTTP request, Crowd will recognise the request as coming from the request's originator, not the proxy server. This is particularly useful if you want single sign-on amongst several applications running behind different proxy servers. Our documentation tells you how to set this up.

Don't have Crowd 1.4 yet?
Take a look at the new features and other highlights in the Crowd 1.4 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.4.1

JIRA Issues (7 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1063	Under heavy load client libraries will leak sockets into CLOSE_WAIT.		Resolved
CWD-1061	crowd-integration-client-1.4 does not work with Jive in a clustered environment, because the CrowdUser object contains a non-serializable member field (securityServerClient).		Resolved
CWD-1051	Confluence error: Illegal configuration. No default cache is configured		Resolved
CWD-1049	The 'crowd' context is hardcoded into the login.jsp page		Resolved
CWD-1046	Profiling (under Admin -> Logging & Profiling) is not working		Resolved
CWD-541	Allow specification of Trusted Proxy Servers		Resolved
CWD-183	Problems with LDAP group or user names that contain / or \.		Resolved

Crowd 1.4 Release Notes

8 May 2008

The Atlassian Crowd team is proud to release **Crowd 1.4**.

Crowd 1.4 supports nested groups in LDAP directories. This means a group can now be a member of another group, making management of permissions much easier. For example, a Crowd-integrated Confluence or JIRA site will see users in sub-groups as members of the parent group.

The new Self-Service Console gives you the option to allow any authorised Crowd user to update their own user profile and password and to view their authorisation details.

There's a new directory connector for Novell eDirectory. Crowd also supports read-only connections to an LDAP directory using the Posix

schema. This is useful if you have a Unix installation and want to integrate it with an LDAP directory.

For the development community, a new plugin framework supports customised event listeners and password encoders.

Highlights of this release:

- Nested Groups
- Self-Service Console
- Novell eDirectory Connector
- Posix Support for LDAP Directories
- Plugin Framework
- More than 30 Improvements and Bug-Fixes

Responding to your feedback:

4 new feature requests implemented

90 votes satisfied

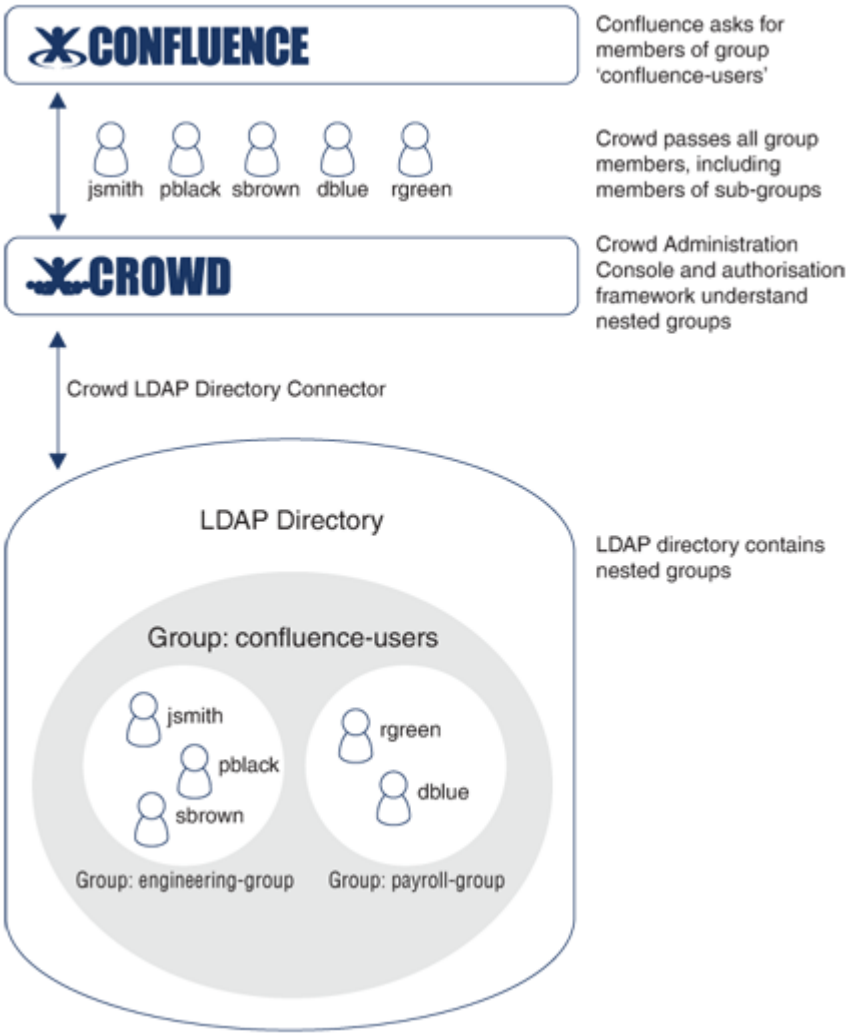
Keep logging your votes and issues. They help us decide what needs doing!

Upgrading to Crowd 1.4
You can download Crowd from the Atlassian website. If upgrading from a previous version, please read the Crowd 1.4 Upgrade Notes.

Highlights of Crowd 1.4

Nested Groups

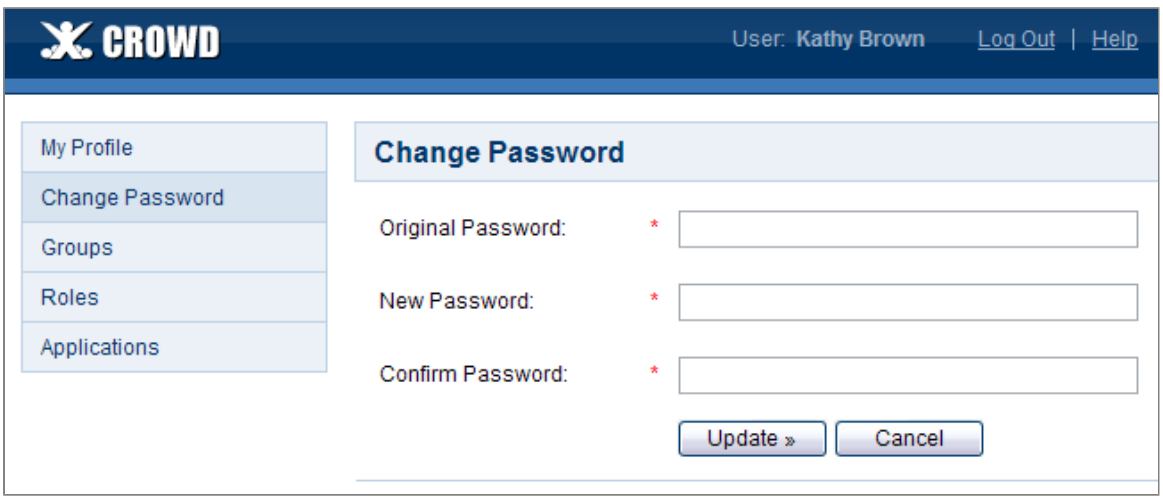
- In your LDAP directory, you can assign a group as a member of another group.
- In Crowd, you can map any group to an application, including a group which contains other groups. Currently, nested groups are supported for LDAP directory connectors only.
- For example, you might have two LDAP groups: 'engineering-group' and 'payroll-group'. Now you want to allow all members of those groups to access your Confluence wiki. You can create a group called 'confluence-users', mapped to the Confluence application, with members 'engineering-group', 'payroll-group' and any other groups and users. Crowd will allow members of those groups and sub-groups to log in to Confluence. When Confluence requests a list of the users in the 'confluence-users' group, Crowd will present all users in the group plus all users in its sub-groups.
- Good news for our Confluence, JIRA and other Atlassian customers — this feature satisfies your requests for nested groups in those products too.
- Take a look at our documentation.



2

Self-Service Console

- Crowd users, including non-administrators, can log in to Crowd.
- Change or reset your own password.
- Update your user profile.
- View your group and role membership.
- See a list of the applications you can log in to.
- The new Crowd User Guide explains the ins and outs.



Novell eDirectory Connector

- Crowd 1.4 provides a built-in directory connector for Novell eDirectory.
- Take a look at our documentation.

Posix Support for LDAP Directories

- Crowd supports read-only connections to an LDAP directory using the Posix/NIS schema.
- Initially, our support is targeted at OpenLDAP directories.
- This is useful if you have a Unix installation and want to integrate with an LDAP directory.
- Here's our documentation on connecting your LDAP directory using the Posix/NIS schema.

Plugin Framework

- For our development community, the new plugin framework supports customised event listeners and password encoders.
- For example, you might decide to write your own event listener to audit failed Crowd authentication requests. Within Crowd itself, the reset password listener uses the new event framework.
- You can create your own plugin to use a specific password encryption algorithm that Crowd does not support out of the box. Crowd's own password encoders provide examples of such plugins.

More than 30 Improvements and Bug-Fixes

JIRA Issues (35 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-1035	FATAL log messages produced when calling SecurityServerClientFactory.getSecurityServerClient().getClientProperties().updateProperties(properties)		Resolved
CWD-1032	Fix Upgrade Task 114 password encryption attribution		Resolved
CWD-1031	Fix XML importer parser		Resolved
CWD-1016	Update common modules		Resolved
CWD-1011	Atlassian Importer does not import passwords correctly		Closed
CWD-993	XML backup does not include delegated directory users and groups		Resolved
CWD-988	Provide read-only support for the POSIX schema		Resolved
CWD-979	Change created AD group type to Distribution		Resolved
CWD-978	Update Spring LDAP to 1.2.1		Resolved
CWD-976	Update directory importer documentation to better explain what's allowed and what isn't		Resolved

CWD-968	Users deleted from JIRA are not removed from the client side cache in CrowdCredentialsProvider.		Closed
CWD-959	Creation of Principals from a client application (JIRA/Confluence) will fail silently when there is multiple directories, one of those being an Internal Directory.		Resolved
CWD-952	Upgrade atlassian-user to be compatible with interface change for Confluence 2.8		Resolved
CWD-942	Problems when creating users from JIRA/Confluence in internal Crowd directories		Resolved
CWD-941	Allow client proxy and connection pool configuration in crowd.properties		Resolved
CWD-936	Provide the ability to choose an encryption type for a Generic Directory		Resolved
CWD-934	Online help links for new 1.4 features		Resolved
CWD-924	SSO failure when authenicating two users in two tabs (in one browser)		Resolved
CWD-920	OpenLDAP MD5 encrypted password stored as plain text		Resolved
CWD-903	Configure redirection of context-sensitive online help links for existing 1.3 release		Resolved
CWD-898	Crowd 1.3 UI is not compatible with IE 6		Resolved
CWD-870	CrowdCredentialsProvider exception handling improvements		Resolved
CWD-782	Textual changes on new directory importer screens		Resolved
CWD-684	Add Crowd Directory Information to the Crowd logs		Closed
CWD-680	Jive Forums 5.5.9 and above Support		Resolved
CWD-676	Event listener exception during startup		Resolved
CWD-614	Implement caching on Crowd client layer		Resolved
CWD-569	Unable to store group/role description		Resolved
CWD-547	crowd scans all Person objects in AD when it doesn't need to.		Resolved
CWD-486	Document configuring Novell eDirectory as an LDAP Directory Connector		Resolved
CWD-485	Officially support integration with Novell eDirectory		Resolved
CWD-306	Allow users to manage their accounts and view thier details in a 'self service' console.		Resolved
CWD-153	Fedora DS		Resolved
CWD-74	Support groups-within-groups		Resolved
CWD-25	Plugins System		Resolved

Crowd 1.3.3 Release Notes

14 October 2008
Crowd 1.3.3 is a recommended upgrade which fixes a parameter injection vulnerability, as described in the security advisory. Please refer to the advisory for details of the security vulnerability, risk assessment and mitigation strategies.

The latest version of Crowd, at the time of these release notes, is Crowd 1.5.1. We are supplying version 1.3.3 as an upgrade for versions 1.3.x, to fix the security vulnerability.

Don't have Crowd 1.5 yet?
Take a look at the new features and other highlights in the Crowd 1.5 Release Notes. And of course, Crowd 1.5.1 also includes the features of Crowd 1.4.

Download Latest Version

Crowd 1.3.2 Release Notes

3 April 2008
The Crowd development team presents **Crowd 1.3.2**. The main purpose of this release is to provide compatibility with the upcoming release of Confluence 2.8. We have updated Crowd's atlassian-user integration module to support an interface change in Confluence.

This release also fixes a problem occurring when an application attempts to add a user, where multiple directories are mapped to the application.

Don't have Crowd 1.3 yet?
Take a look at the new features and other highlights in the Crowd 1.3 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.3.2

JIRA Issues (5 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-959	Creation of Principals from a client application (JIRA/Confluence) will fail silently when there is multiple directories, one of those being an Internal Directory.		Resolved
CWD-952	Upgrade atlassian-user to be compatible with interface change for Confluence 2.8		Resolved
CWD-475	Crowds Import and Export seem to contain duplicate data and dies for Foreign Key violations.		Resolved
CWD-347	Crowd and direct LDAP conenction demanding different DNs (at least against ApacheDS)		Resolved
CWD-125	Provide Lotus Domino Support		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.3.1 Release Notes

20 March 2008
The Crowd development team has released **Crowd 1.3.1**. This is a bug-fix release, which solves some problems in Crowd 1.3.

Don't have Crowd 1.3 yet?
Take a look at the new features and other highlights in the Crowd 1.3 Release Notes.

Download Latest Version

Complete List of Fixes in Crowd 1.3.1

JIRA Issues (13 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-924	SSO failure when authenicating two users in two tabs (in one browser)		Resolved
CWD-920	OpenLDAP MD5 encrypted password stored as plain text		Resolved

CWD-916	View Principal/User sessions in the Crowd console directory links broken		Resolved
CWD-914	Viewing OpenLDAP Directoy Connector Info throws an exception		Resolved
CWD-909	User Name RDN Attribute field is not populated for Delegated Authentication directory screen		Resolved
CWD-900	Paged result size should not persist on directories that have not have "Use Paged Results" enabled.		Resolved
CWD-899	When creating an LDAP based directory a password algorithm attribute is being set for all directory types regardless if they use one or not.		Resolved
CWD-898	Crowd 1.3 UI is not compatible with IE 6		Resolved
CWD-875	User groups list in directory should sort alpha-numeric rather than natural.		Resolved
CWD-782	Textual changes on new directory importer screens		Resolved
CWD-561	Support the 'uid' and 'cn' attribute with the inetorgperson object at the same time		Resolved
CWD-527	IllegalDataException from active-directory authentication failure		Resolved
CWD-439	Errors in the Confluence logs about Crowd (XFire prolog EOF)		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.3 Release Notes

4 March 2008

The Atlassian Crowd team is delighted to present **Crowd 1.3**. This release includes innovative solutions for LDAP group administration, cross-directory user imports and a streamlined management interface.

A new directory type allows you to combine the features of a Crowd directory with authentication delegated to an LDAP directory. This means that you can use Crowd's flexible group management when the LDAP groups do not suit your requirements. For example, set up a simple group configuration for use with Confluence, JIRA and other Atlassian products.

Our new Directory Importer allows you to copy your users from one directory into another — from and to any type of directory. For example, you can copy users, groups and roles from an LDAP directory to a Crowd directory, or vice versa.

The Crowd Administration Console has a new menu structure with an enhanced look and feel. It's easier to find the functions that you perform most often and interaction is more intuitive.

Installing and setting up Crowd is simpler and faster. Database configuration is now part of the Setup Wizard. When upgrading, you have the option to import your data from an XML backup or point Crowd at your existing database, and so bypass most of the Setup Wizard.

To speed up troubleshooting, you can configure your logging levels and enable performance profiling via the Administration Console. There's a bucketful of improvements in performance and efficiency, and many other fixes and enhancements.

Highlights of this release:

- LDAP Authentication with Crowd Groups and Roles
- Cross-Directory User Importer
- Streamlined User Interface
- Simplified Installation, Setup and Integration
- Configuration of Logging and Profiling via Console
- Improved Performance and Efficiency
- Highlights for the Developers
- Plus Over 60 Improvements and Bug-Fixes

Responding to your feedback:

6 new feature requests implemented

36 votes satisfied

Your votes and issues help us keep improving our products. Keep 'em coming!

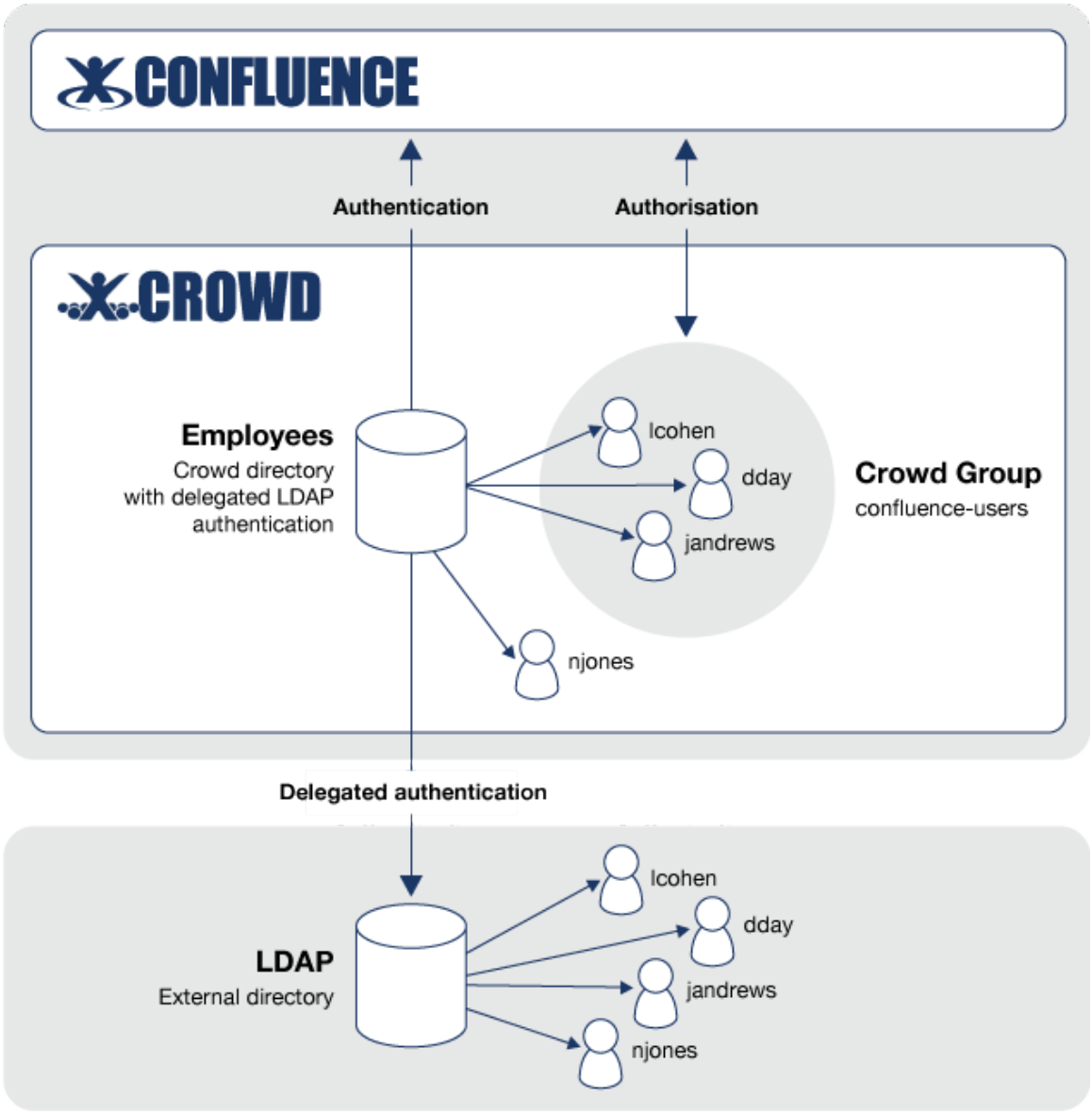
Upgrading to Crowd 1.3

You can download Crowd from the Atlassian website. If upgrading from a previous version, please read the Crowd 1.3 Upgrade Notes.

Highlights of Crowd 1.3

LDAP Authentication with Crowd Groups and Roles

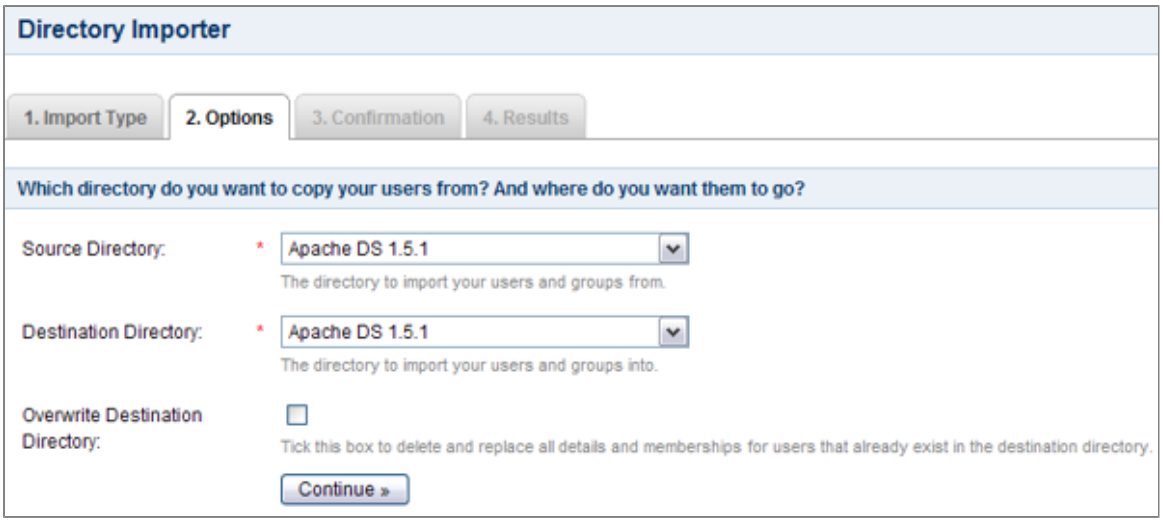
- Crowd 1.3 provides a new directory type, Delegated Authentication, combining the features of a Crowd internal directory with delegated LDAP authentication.
- This allows you to have your users authenticated via an external LDAP directory while managing the groups and roles in Crowd.
- Use Crowd's flexible and simple group management when the LDAP groups do not suit your requirements. For example, you can set up a group configuration in Crowd for use with Confluence, JIRA and other Atlassian products.
- Avoid the performance issues which might result from downloading large numbers of groups from LDAP.
- Use the new Directory Importer, described below, to synchronise your LDAP users with your Crowd directory.
- When a user logs in for the first time, Crowd automatically adds them to the Crowd directory if not already present.



2

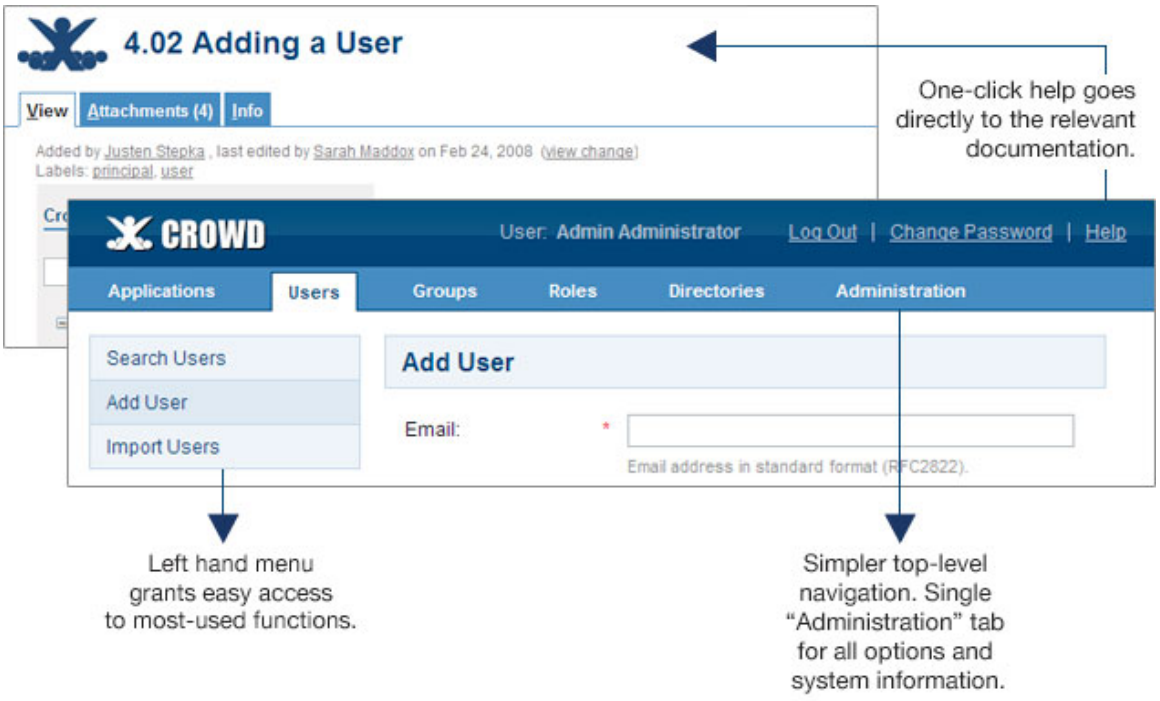
Cross-Directory User Importer

- Our new Directory Importer allows you to copy your users from one directory into another.
- Provided that the directory is defined in Crowd, you can copy from and to any directory type.
- For example, you might import users, groups, roles and memberships from an LDAP directory to a new Delegated Authentication directory (described above) so that you can manage the users, groups and roles in Crowd while allowing users to log in with their LDAP passwords.
- Read about the Directory Importer.



Streamlined User Interface

- The Crowd Administration Console has a new menu structure and an enhanced look and feel.
- A left-hand menu grants easy access to the functions you use most often, such as searching for a user or group.
- A single 'Administration' tab holds the configuration options, system information and backup/restore functions.
- In the interests of simplicity, we've changed the term 'principal' to 'user' throughout.
- When you click a 'Help' link, the relevant documentation page opens immediately.

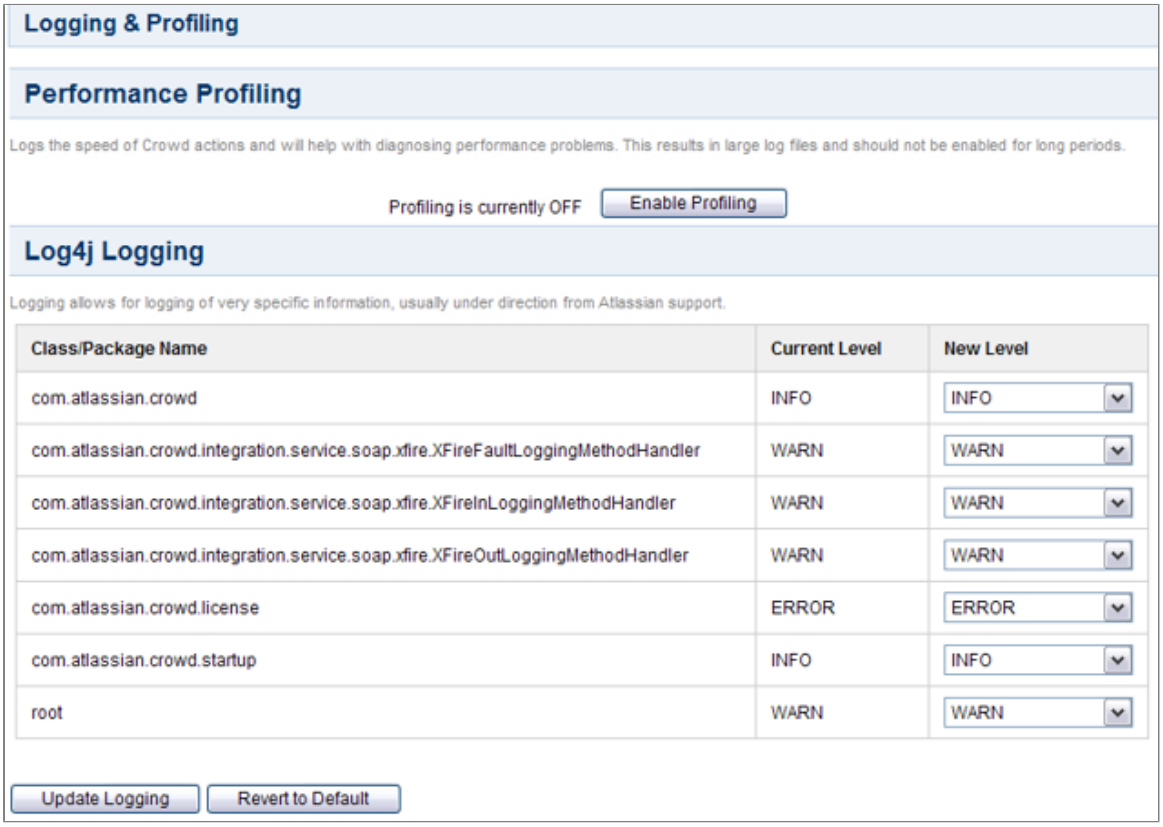


Simplified Installation, Setup and Integration

- Database configuration is now part of the Setup Wizard, which will update the configuration files based on the options you select.
- You can choose between a JNDI datasource (i.e. server-managed) or a simpler JDBC configuration.
- When upgrading, you can import an XML backup of your Crowd database or connect to an existing database via the Setup Wizard. This means that you don't have to go through the whole Setup Wizard, nor do a manual backup and restore of your Crowd database files.
- When integrating an application with Crowd, you'll notice that there's just one single JAR file to copy.

Configuration of Logging and Profiling via Console

- Enable and disable performance profiling.
- Configure your logging levels via the Crowd Administration Console, for quick and simple runtime troubleshooting.
- Edit the log configuration file for more advanced settings.
- Read the documentation.



Improved Performance and Efficiency

- You'll notice faster search results on the Administration Console screens, such as the Application Browser and User Browser.
- That annoying 'POSTDATA has expired' message no longer appears when you click the 'Back' button.
- Search results returned to a Crowd application are now sorted alphabetically — such as the list of groups shown in a Confluence group picker.
- We've fixed the Hibernate StaleStateException error that was causing occasional performance degradation and authentication failures.
- You can choose to store the login session tokens in the Crowd database (as done prior to Crowd 1.3) or in memory (new option as from Crowd 1.3). Depending upon your installation, in-memory storage could greatly improve response times during authentication. Read about configuring token storage.
- Gzip compression of Crowd Security Server output is now optional. You can turn it on or off via the Crowd Administration Console. Some reasons why you may want to turn Gzip compression off:
    - It may be easier to debug problems using uncompressed data.
    - Some agents, such as older versions of Internet Explorer, have problems with the Gzip format.

Highlights for the Developers

- The Java client library API has been upgraded. Read more about the API changes and the upgrade notes.
- You can pass the `crowd.properties` file to a client application as an environment variable.

Plus Over 60 Improvements and Bug-Fixes

JIRA Issues (69 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-897	Generic LDAP Directory type is displayed as OpenLDAP not Generic		Closed
CWD-882	Unalble to update the 'active' flag of an Application		Resolved
CWD-855	OGNL exceptions are thrown when remoing Groups and Roles in the Demo app		Resolved
CWD-849	Rationalise the path to crowd-init.properties that's displayed on startup		Resolved
CWD-847	Error message is confusing when no directories are mapped to an application		Resolved
CWD-838	Updating any directory type in Crowd has multiple validation problems.		Resolved
CWD-830	Change Crowd WAR deployment to zip archive.		Resolved
CWD-829	When updating a Delegated or Connector based directory, required fields are not marked as required.		Resolved
CWD-828	When updating an Internal Directory, there is no validation performed on the Configuration tab		Resolved
CWD-824	Session timeout during the installation should be larger than 5 minutes		Resolved
CWD-823	JDBC connection should default to MySQL		Resolved
CWD-822	crowd-init.properties value not set error message during startup is not useful		Resolved
CWD-818	Admin Console: Selected tab CSS needs tweaking for Windows compatability		Resolved
CWD-817	Default results per page to 100		Resolved
CWD-806	Fix log4j.properties so dates are displayed in log files.		Resolved
CWD-805	Crowd's Add Directory Screen indicates we support Open Directory.		Resolved
CWD-802	Allow to pass the contents of the crowd.properties programmatically to the crowd client		Resolved
CWD-800	When associating a Group/Role to a Principal in the Demo application, an error is displayed		Resolved
CWD-799	When creating a Group/Role to a Principal in the Demo application, an exception is thrown.		Resolved
CWD-798	When adding a Group or Role via the Demo app, the description field is not being persisted.		Resolved
CWD-790	Have you seen the client/lib directory lately? The current count is about 46 JAR files!		Resolved
CWD-775	Add Logging & Profiling functionality into Crowd Admin screen.		Resolved
CWD-768	Hibernate DAOs for Principals and Groups close the Hibernate Session when adding		Resolved

CWD-767	Crowd's Client libraries should be slimmed down to a single JAR file containing all required classes for a Crowd Client		Resolved
CWD-765	File missing in 1.2.2 release		Resolved
CWD-758	Hibernate StaleStateExceptions in Crowd		Resolved
CWD-757	Crowd with delegated LDAP auth - update documentation for Bamboo-Crowd integration		Resolved
CWD-739	Concurrency Issue in client libraries may result in multiple caches		Resolved
CWD-738	Allow configuring of request logs in the Crowd client libraries.		Resolved
CWD-731	OGNL Exception being thrown when updating a principal		Resolved
CWD-728	The Internal Directory is throwing a java.lang.IndexOutOfBoundsException: Index: 0, Size: 0 on requiresPasswordChange()		Resolved
CWD-727	Poor logging of a Token miss in the In-memory token cache.		Resolved
CWD-726	java.lang.IllegalStateException: Can't overwrite cause exception seen in Crowd		Resolved
CWD-724	Configuration classs for the LDAP importer		Resolved
CWD-723	LDAP Importer, to migrate data from one directory into another.		Resolved
CWD-720	Enable import from XML in the setup process		Closed
CWD-716	Error when attempting to remove a group		Resolved
CWD-711	The HTTPAuthenticator isAuthenticated method should initially check for a token		Resolved
CWD-706	Fix logging on startup for the OpenID Server. Stop the logging of Hibernate INFO.		Resolved
CWD-703	Crowd OpenID WAR file is missing commons-logging jar.		Resolved
CWD-700	The isMember call for groups can be slow for very large groups in an Internal Directory		Closed
CWD-699	Crowd SSO is incompatible with JIRA 3.12/Confluence 2.7 trusted application feature.		Resolved
CWD-694	ehcache-1.2.3.jar is missing from client/lib folder.		Resolved
CWD-688	Help links directly in the administration console		Closed
CWD-686	Sort groups, users and roles before returning results to the security server client		Resolved
CWD-685	Write System Info page to atlassian-crowd.log on Crowd startup		Resolved
CWD-675	remove "cache-control: no-store" on search results pages		Resolved
CWD-669	Adding group/role with prefixed space causes Hibernate error		Resolved
CWD-666	Persistence system should use c3p0 so hibernate's default pooling system is not used.		Resolved
CWD-654	Xalan is missing from the demo applications WEB-INF/lib folder.		Closed

CWD-650	Update the crowd distribution module parent POM version to version 10	Resolved
CWD-649	Update the atlassian-crowd module parent POM version to version 7	Closed
CWD-646	Move FishEye connector outside crowd-core	Resolved
CWD-645	Use Spring dependency injection for SecurityServerClient and HttpAuthenticator in Crowd applications	Resolved
CWD-639	Crowd hanging client applications, error with token manager	Resolved
CWD-633	Allow the crowd.properties file to be passed to a Client application as an environment variable	Resolved
CWD-622	Make SecurityServerClient not static	Closed
CWD-586	start_crowd.sh and build.sh fail on Solaris	Resolved
CWD-584	Adding a Principal to Sun DSEE 6.2 throws a NullPointerException	Resolved
CWD-570	First Name not being displayed from Apache DS	Resolved
CWD-499	Creating Groups and Principals fails on 2000	Resolved
CWD-481	Support CRYPT encryption in OpenLDAP connector	Resolved
CWD-466	Storing login tokens in an external DB is inefficient	Resolved
CWD-453	Crowd core jar breaks in Grails, need a new slimmed-down client jar	Resolved
CWD-427	OpenLDAP Connector should default to SSHA encryption.	Resolved
CWD-389	GZip compression is optional through the administration console.	Resolved
CWD-350	Tuckey rewrite filter dials home by doing a DNS lookup.	Resolved
CWD-208	Mixed authentication and authorization support for external directory connectors.	Resolved
CWD-149	Config Test doesn't appear to obey Directory and Group rules	Resolved

Known Issues in This Release

We have an enthusiastic and dedicated group of testers and customers who jump in there, try out the new Crowd release, and report any problems so that we can fix them quickly. Here's a list of known issues which will be fixed in our next point release.

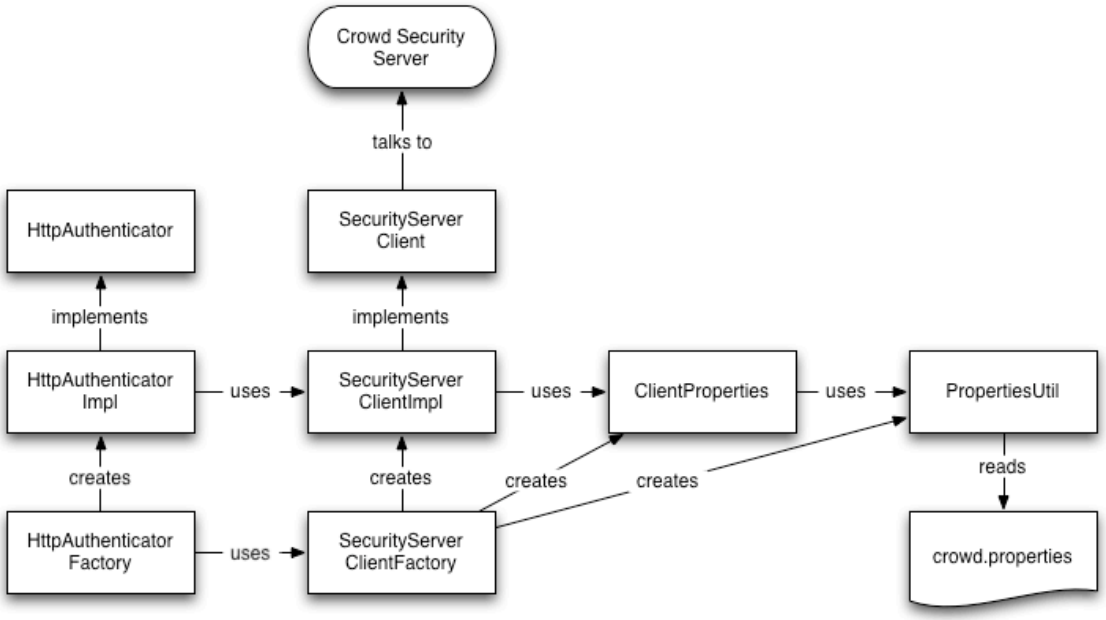
A big **thank you** to everyone who helps us ensure that Crowd keeps getting better and better.

Client API Changes

Crowd 1.3 brings a rework of the internals of the Crowd Client library — see CWD-622. This page gives a summary of the API changes.

Description of the changes

- The static implementations of HttpAuthenticator and SecurityServerClient have been removed. They have been replaced with instantiable objects.
- The GenericClient has been removed and its functions have been absorbed into the new SecurityServerClient and the ClientProperties objects.
- The relationships in the new class structure are represented below:



Why go to non-static?

- Makes it easier to unit test your applications. Simply mock out the SecurityServerClient or HttpAuthenticator interfaces to test business logic without being tied to the collaborators.
- Allows you to have multiple 'applications' in one classloader.

But I liked my static calls!

- SecurityServerClientFactory and HttpAuthenticatorFactory are provided to allow for a fast migration to the new API. The logical functionality of the client and authenticator are unchanged.
- So for example, instead of:

```
1.HttpAuthenticator.isAuthenticated(request);
```

  you could use:

```
1.HttpAuthenticatorFactory.getHttpAuthenticator().isAuthenticated(request);
```

What are my options?

1. Use the supplied factory methods to manage singleton instances, OR
2. Externally manage singleton instances, e.g. via an IoC container like Spring.

Using the factories

The factories, `HttpAuthenticatorFactory` and `SecurityServerClientFactory`, provide quick access to implementations of the `HttpAuthenticator` and `SecurityServerClient`. They manage singleton instances of the beans. This means that if you do opt to use the factories, then you should never instantiate `HttpAuthenticatorImpl` or `SecurityServerClientImpl` directly.

The factories naturally assume that there is one application client per classloader, i.e. one `SecurityServerClient` and one `HttpAuthenticator`.

Using an IoC container

Managing the singleton implementations externally may be a convenient approach for applications that use an IoC container. For example, Spring could be used to manage the instances of `SecurityServerClientImpl` and `HttpAuthenticatorImpl`. In Crowd, internally, we use this approach.

If you would like to use the standard Spring configuration, which loads the client properties from `crowd.properties`, simply add the `applicationContext-CrowdClient.xml` from the classpath to your Spring configuration:

```
1.<context-param>
2.     <param-name>contextConfigLocation</param-name>
3.     <param-value>
4.          classpath:/applicationContext-CrowdClient.xml
5.     </param-value>
6.</context-param>
```

This file is located in the `crowd-integration-client.jar`.

If you would like to customise your own configuration, modify the bean configuration to suit your needs:

```
01.<?xml version="1.0" encoding="UTF-8"?>
02.<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "
http://www.springframework.org/dtd/spring-beans.dtd">
03.
04.<beans>
05.
06.          <bean id="propertyUtils" class="com.atlassian.crowd.util.PropertyUtils"/>
07.
08.          <bean id="clientProperties" class=
"com.atlassian.crowd.integration.service.soap.client.ClientProperties">
09.                    <constructor-arg ref="propertyUtils"/>
10.          </bean>
11.
12.          <bean id="securityServerClient" class=
"com.atlassian.crowd.integration.service.soap.client.SecurityServerClientImpl">
13.                    <constructor-arg ref="clientProperties"/>
14.          </bean>
15.
16.          <bean id="httpAuthenticator" class=
"com.atlassian.crowd.integration.http.HttpAuthenticatorImpl">
17.                    <constructor-arg ref="securityServerClient"/>
18.          </bean>
19.
20.          <bean id="verifyTokenFilter" class=
"com.atlassian.crowd.integration.http.VerifyTokenFilter">
21.                    <constructor-arg ref="httpAuthenticator"/>
22.          </bean>
23.
24.          <bean id="crowdAuthenticationInterceptor" class=
"com.atlassian.crowd.integration.xwork.CrowdAuthenticationInterceptor">
25.                    <constructor-arg ref="httpAuthenticator"/>
26.          </bean>
27.
28.</beans>
```

Make sure that you do not use the factories (either directly or implicitly) when externally managing singletons.

If you would like to use the `VerifyTokenFilter`, you can use Spring to autowire the servlet filter by defining it in your `web.xml`:

```
01.<filter>
02.     <filter-name>verifyTokenFilter</filter-name>
03.     <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
04.</filter>
05.
06.<filter-mapping>
07.     <filter-name>verifyTokenFilter</filter-name>
08.     <url-pattern>/secure/*</url-pattern>
09.</filter-mapping>
```

This will protect all resources matching the `/secure/*` pattern.

Known Issues in Crowd 1.3

We have an enthusiastic and dedicated group of testers and customers who jump in there, try out the new Crowd release, and report any problems so that we can fix them quickly. Below is a list of known issues. We're working on them, and will have a point release out as soon as possible.

A big **thank you** to everyone who helps us ensure that Crowd keeps getting better and better.

While you're waiting, take a look at the great new features in Crowd 1.3.

You can also browse the Crowd project in our issue tracker to see what's fixed and what's not, for each release.

Issues to be Fixed in Crowd 1.3.1

JIRA Issues (13 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-924	SSO failure when authenicating two users in two tabs (in one browser)		Resolved
CWD-920	OpenLDAP MD5 encrypted password stored as plain text		Resolved
CWD-916	View Principal/User sessions in the Crowd console directory links broken		Resolved

CWD-914	Viewing OpenLDAP Directoy Connector Info throws an exception		Resolved
CWD-909	User Name RDN Attribute field is not populated for Delegated Authentication directory screen		Resolved
CWD-900	Paged result size should not persist on directories that have not have "Use Paged Results" enabled.		Resolved
CWD-899	When creating an LDAP based directory a password algorithm attribute is being set for all directory types regardless if they use one or not.		Resolved
CWD-898	Crowd 1.3 UI is not compatible with IE 6		Resolved
CWD-875	User groups list in directory should sort alpha-numeric rather than natural.		Resolved
CWD-782	Textual changes on new directory importer screens		Resolved
CWD-561	Support the 'uid' and 'cn' attribute with the inetorgperson object at the same time		Resolved
CWD-527	IllegalDataException from active-directory authentication failure		Resolved
CWD-439	Errors in the Confluence logs about Crowd (XFire prolog EOF)		Resolved

Crowd 1.3 Beta Release Notes

20 February 2008

Crowd 1.3 will be launched early in March 2008. A beta release is currently undergoing internal testing. These release notes apply to **Crowd 1.3 beta**. We'll publish the final release notes with the release of Crowd 1.3.0.

If you would like to participate in testing the beta release, please contact Crowd Support.

Upgrading to Crowd 1.3 Beta
If upgrading from a previous version, please read the Upgrade Notes.

What's Coming in Crowd 1.3

LDAP Authentication with Crowd Groups and Roles

- Crowd 1.3 provides a new directory type, Delegated Authentication, combining the features of a Crowd internal directory with delegated LDAP authentication.
- This allows you to have your users authenticated via an external LDAP directory while managing the groups and roles in Crowd.
- Use Crowd's flexible and simple group management when the LDAP groups do not suit your requirements. For example, you can set up a group configuration in Crowd for use with Confluence and other Atlassian products.
- Avoid the performance issues which might result from downloading large numbers of groups from LDAP.
- Use the new Directory Importer, described below, to synchronise your LDAP users with your Crowd directory.
- When a user logs in for the first time, Crowd automatically adds them to the Crowd directory if not already present.

Cross-Directory User Importer

- Our new Directory Importer allows you to copy your users from one directory into another.
- Provided that the directory is defined in Crowd, you can copy from and to any directory type.
- For example, you might import users, groups, roles and memberships from an LDAP directory to a new Delegated Authentication directory (described above) so that you can manage the users, groups and roles in Crowd while allowing users to log in with their LDAP passwords.
- Read about the Directory Importer.

Streamlined User Interface

- The Crowd Administration Console has a new menu structure and an enhanced look-and-feel.

sed functions, so that an administrator has fewer steps to perform and interaction is more intuitive.
- The 'Help' links on the Administration Console point directly to the relevant documentation pages.

4 Simplified Installation and Setup

- Database configuration is now part of the Setup Wizard, which will update the configuration files based on the options you select.
- You can choose between a JNDI datasource (i.e. server-managed) or a simpler JDBC configuration.
- When upgrading, you can import an XML backup of your Crowd database or connect to an existing database via the Setup Wizard. This means that you don't have to go through the whole Setup Wizard, nor do a manual backup and restore of your Crowd database files.

5 Logging and Profiling Configuration via Console

- Enable profiling and configure your logging levels via the Crowd Administration Console.

6 Improved Performance and Efficiency

- You'll notice faster search results on the Administration Console screens, such as the Application Browser and User Browser, etc.
- That annoying 'POSTDATA has expired' message no longer appears when you click the 'Back' button.
- Search results returned to a Crowd application are now sorted alphabetically — such as the list of groups shown in a Confluence group picker.
- We've fixed the Hibernate StaleStateException error that was causing occasional performance degradation and authentication failures.
- You can choose to store the login session tokens in the Crowd database (as done prior to Crowd 1.3) or in memory (new option as from Crowd 1.3). Depending upon your installation, in-memory storage could greatly improve response times during authentication. Read about configuring token storage.
- Gzip compression of Crowd Security Server output is now optional. You can turn it on or off via the Crowd Administration Console. Some reasons why you may want to turn Gzip compression off:
  - It may be easier to debug problems using uncompressed data.
  - Some agents, such as older versions of Internet Explorer, have problems with the Gzip format.

7 Highlights for the Developers

- The Java client library API has been upgraded. Read more about the API changes and the upgrade notes.
- You can pass the `crowd.properties` file to a client application as an environment variable.

Updates and Fixes in this Release

JIRA Issues (69 issues)

Key	Summary	Priority	Status
CWD-897	Generic LDAP Directory type is displayed as OpenLDAP not Generic		Closed
CWD-882	Unalble to update the 'active' flag of an Application		Resolved
CWD-855	OGNL exceptions are thrown when remoing Groups and Roles in the Demo app		Resolved
CWD-849	Rationalise the path to crowd-init.properties that's displayed on startup		Resolved
CWD-847	Error message is confusing when no directories are mapped to an application		Resolved
CWD-838	Updating any directory type in Crowd has multiple validation problems.		Resolved
CWD-830	Change Crowd WAR deployment to zip archive.		Resolved
CWD-829	When updating a Delegated or Connector based directory, required fields are not marked as required.		Resolved
CWD-828	When updating an Internal Directory, there is no validation performed on the Configuration tab		Resolved
CWD-824	Session timeout during the installation should be larger than 5 minutes		Resolved

CWD-823	JDBC connection should default to MySQL		Resolved
CWD-822	crowd-init.properties value not set error message during startup is not useful		Resolved
CWD-818	Admin Console: Selected tab CSS needs tweaking for Windows compatability		Resolved
CWD-817	Default results per page to 100		Resolved
CWD-806	Fix log4j.properties so dates are displayed in log files.		Resolved
CWD-805	Crowd's Add Directory Screen indicates we support Open Directory.		Resolved
CWD-802	Allow to pass the contents of the crowd.properties programmatically to the crowd client		Resolved
CWD-800	When associating a Group/Role to a Principal in the Demo application, an error is displayed		Resolved
CWD-799	When creating a Group/Role to a Principal in the Demo application, an exception is thrown.		Resolved
CWD-798	When adding a Group or Role via the Demo app, the description field is not being persisted.		Resolved
CWD-790	Have you seen the client/lib directory lately? The current count is about 46 JAR files!		Resolved
CWD-775	Add Logging & Profiling functionality into Crowd Admin screen.		Resolved
CWD-768	Hibernate DAOs for Principals and Groups close the Hibernate Session when adding		Resolved
CWD-767	Crowd's Client libraries should be slimmed down to a single JAR file containing all required classes for a Crowd Client		Resolved
CWD-765	File missing in 1.2.2 release		Resolved
CWD-758	Hibernate StaleStateExceptions in Crowd		Resolved
CWD-757	Crowd with delegated LDAP auth - update documentation for Bamboo-Crowd integration		Resolved
CWD-739	Concurrency Issue in client libraries may result in multiple caches		Resolved
CWD-738	Allow configuring of request logs in the Crowd client libraries.		Resolved
CWD-731	OGNL Exception being thrown when updating a principal		Resolved
CWD-728	The Internal Directory is throwing a java.lang.IndexOutOfBoundsException: Index: 0, Size: 0 on requiresPasswordChange()		Resolved
CWD-727	Poor logging of a Token miss in the In-memory token cache.		Resolved
CWD-726	java.lang.IllegalStateException: Can't overwrite cause exception seen in Crowd		Resolved
CWD-724	Configuration classs for the LDAP importer		Resolved
CWD-723	LDAP Importer, to migrate data from one directory into another.		Resolved
CWD-720	Enable import from XML in the setup process		Closed

CWD-716	Error when attempting to remove a group		Resolved
CWD-711	The HTTPAuthenticator isAuthenticated method should initially check for a token		Resolved
CWD-706	Fix logging on startup for the OpenID Server. Stop the logging of Hibernate INFO.		Resolved
CWD-703	Crowd OpenID WAR file is missing commons-logging jar.		Resolved
CWD-700	The isMember call for groups can be slow for very large groups in an Internal Directory		Closed
CWD-699	Crowd SSO is incompatible with JIRA 3.12/Confluence 2.7 trusted application feature.		Resolved
CWD-694	ehcache-1.2.3.jar is missing from client/lib folder.		Resolved
CWD-688	Help links directly in the administration console		Closed
CWD-686	Sort groups, users and roles before returning results to the security server client		Resolved
CWD-685	Write System Info page to atlassian-crowd.log on Crowd startup		Resolved
CWD-675	remove "cache-control: no-store" on search results pages		Resolved
CWD-669	Adding group/role with prefixed space causes Hibernate error		Resolved
CWD-666	Persistence system should use c3p0 so hibernate's default pooling system is not used.		Resolved
CWD-654	Xalan is missing from the demo applications WEB-INF/lib folder.		Closed
CWD-650	Update the crowd distribution module parent POM version to version 10		Resolved
CWD-649	Update the atlassian-crowd module parent POM version to version 7		Closed
CWD-646	Move FishEye connector outside crowd-core		Resolved
CWD-645	Use Spring dependency injection for SecurityServerClient and HttpAuthenticator in Crowd applications		Resolved
CWD-639	Crowd hanging client applications, error with token manager		Resolved
CWD-633	Allow the crowd.properties file to be passed to a Client application as an environment variable		Resolved
CWD-622	Make SecurityServerClient not static		Closed
CWD-586	start_crowd.sh and build.sh fail on Solaris		Resolved
CWD-584	Adding a Principal to Sun DSEE 6.2 throws a NullPointerException		Resolved
CWD-570	First Name not being displayed from Apache DS		Resolved
CWD-499	Creating Groups and Principals fails on 2000		Resolved
CWD-481	Support CRYPT encryption in OpenLDAP connector		Resolved

CWD-466	Storing login tokens in an external DB is inefficient		Resolved
CWD-453	Crowd core jar breaks in Grails, need a new slimmed-down client jar		Resolved
CWD-427	OpenLDAP Connector should default to SSHA encryption.		Resolved
CWD-389	GZip compression is optional through the administration console.		Resolved
CWD-350	Tuckey rewrite filter dials home by doing a DNS lookup.		Resolved
CWD-208	Mixed authentication and authorization support for external directory connectors.		Resolved
CWD-149	Config Test doesn't appear to obey Directory and Group rules		Resolved

Crowd 1.2.4 Release Notes

14 October 2008
Crowd 1.2.4 is a recommended upgrade which fixes a parameter injection vulnerability, as described in the security advisory. Please refer to the advisory for details of the security vulnerability, risk assessment and mitigation strategies.

The latest version of Crowd, at the time of these release notes, is Crowd 1.5.1. The previous public release of Crowd 1.2.x was version 1.2.2. Version 1.2.3 was an internal release. We are supplying version 1.2.4 as an upgrade for versions 1.2.x, to fix the security vulnerability.

Don't have Crowd 1.5 yet?
Take a look at the new features and other highlights in the Crowd 1.5 Release Notes. And of course, Crowd 1.5.1 also includes the features of Crowd 1.3 and Crowd 1.4.

Download Latest Version

Crowd 1.2.2 Release Notes

16 January 2008: The Crowd development team has released Crowd 1.2.2.

Crowd 1.2.2 upgrades its packaged version of Apache Tomcat to version 5.5.25, to fix some reported Apache Tomcat vulnerabilities. Tomcat is supplied as the application server in the Crowd Standalone distribution.

This release also solves some problems with the Crowd build and resolves the incompatibility between Crowd single sign-on and the new JIRA/Confluence trusted application feature.

Complete List of Fixes in Crowd 1.2.2

JIRA Issues (14 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-793	Receiving error when trying to build Crowd 1.2.2: taskdef class com.oopsconsultancy.xmltask.ant.XmlTask cannot be found		Resolved
CWD-739	Concurrency Issue in client libraries may result in multiple caches		Resolved
CWD-738	Allow configuring of request logs in the Crowd client libraries.		Resolved
CWD-728	The Internal Directory is throwing a java.lang.IndexOutOfBoundsException: Index: 0, Size: 0 on requiresPasswordChange()		Resolved
CWD-727	Poor logging of a Token miss in the In-memory token cache.		Resolved
CWD-711	The HTTPAuthenticator isAuthenticated method should initially check for a token		Resolved
CWD-710	Update Tomcat to 5.5.25 to fix reported vulns		Closed

CWD-706	Fix logging on startup for the OpenID Server. Stop the logging of Hibernate INFO.		Resolved
CWD-703	Crowd OpenID WAR file is missing commons-logging jar.		Resolved
CWD-699	Crowd SSO is incompatible with JIRA 3.12/Confluence 2.7 trusted application feature.		Resolved
CWD-667	Crowd user caching in JIRA delayed		Resolved
CWD-665	Create an XFire fault logging handler		Resolved
CWD-654	Xalan is missing from the demo applications WEB-INF/lib folder.		Closed
CWD-423	Upgrade to openid4java 0.9.3		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.2.1 Release Notes

10 December 2007: The Crowd development team has released Crowd 1.2.1.

Crowd 1.2.1 fixes some installation problems. Other improvements include the sorting of groups by directory name then group name in the Application Browser.

Fixes in Crowd 1.2.1

JIRA Issues (15 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-657	Acegi jar is missing from the client directory of the distribution		Resolved
CWD-653	ports in the crowd.properties files are incorrect for the demo and openidserver applications with the distribution		Resolved
CWD-651	Confluence importer error with MySQL		Resolved
CWD-650	Update the crowd distribution module parent POM version to version 10		Resolved
CWD-649	Update the atlassian-crowd module parent POM version to version 7		Closed
CWD-648	Xalan is missing from the demo applications WEB-INF/lib folder.		Resolved
CWD-644	Seraph library compatibility issues result in java.lang.NoSuchMethodError: com.atlassian.crowd.integration.seraph.CrowdAuthenticator.getAuthType()Ljava/lang/String;		Resolved
CWD-642	build.xml fails to correctly copy the openid crowd.properties file		Resolved
CWD-638	build.bat no longer properly runs , preventing the environmental changes such as database dialects from be changed automaticly		Resolved
CWD-629	Error found in Internal Directory when a user requires a password change		Resolved
CWD-584	Adding a Principal to Sun DSEE 6.2 throws a NullPointerException		Resolved
CWD-506	LDAP fitlering only supports one fitler.		Resolved
CWD-499	Creating Groups and Principals fails on 2000		Resolved

CWD-342	Sort groups alphabetically or provide a pop-up window to search and choose groups (like Confluence has)		Resolved
CWD-289	Sort groups by name when selecting groups that can access an application		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.2 Release Notes

The Atlassian Crowd team is delighted to present Crowd 1.2.

Crowd 1.2 is a major release that focuses on enhanced integration, security and usability. Crowd's directory permissions now allow finer-grained control, so that you can define the permissions per application. The Group and Role Browsers now display group/role membership. We have enhanced group management in the existing Jive Forums and Apache/Subversion connectors. Our NTLM plugin offers SSO (single sign-on) for JIRA and Confluence via NTLM desktop authentication. A new connector lets you integrate your Acegi security solution with Crowd. And you can import your Bamboo users directly into a Crowd directory.

We'd like to say a special thank you to CustomWare for their assistance with deployment and testing of the NTLM plugin.

Stop Press — 27 February 2008: We got a little bit ahead of ourselves with our announcement of **full** NTLM support in Crowd 1.2. The NTLM plugins for JIRA and for Confluence are provided and supported by a third party, not by Atlassian.

Highlights of this release:

- Directory Permissions per Application
- Group and Role Membership Browser
- Improved Browser for OpenID Login History
- NTLM Support
- Improved Integration with Jive Forums
- Acegi Application Connector
- Group-Based Authorisation Added for Subversion
- New Importer for Bamboo Users
- Plus Over 70 Improvements and Bug-Fixes

Responding to your feedback:

8 new feature requests implemented

68 votes satisfied

Your votes and issues help us keep improving our products, and are much appreciated.

DOWNLOAD latest version

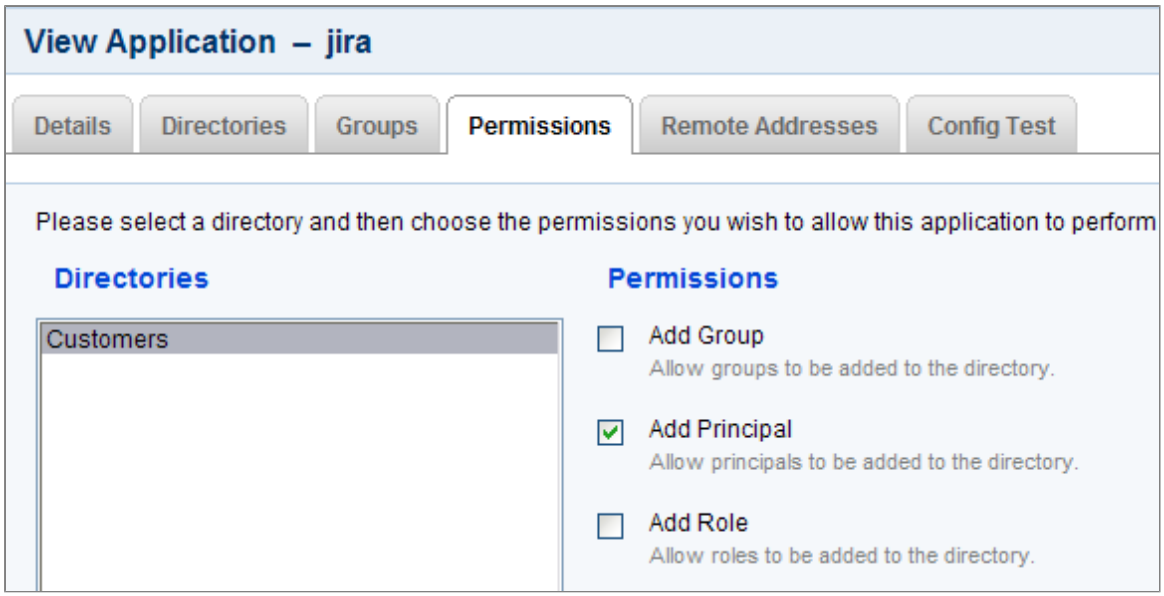
Upgrading to Crowd 1.2
You can download Crowd from the Atlassian website. If upgrading from a previous version, please read the Upgrade Notes.

Highlights of Crowd 1.2

1

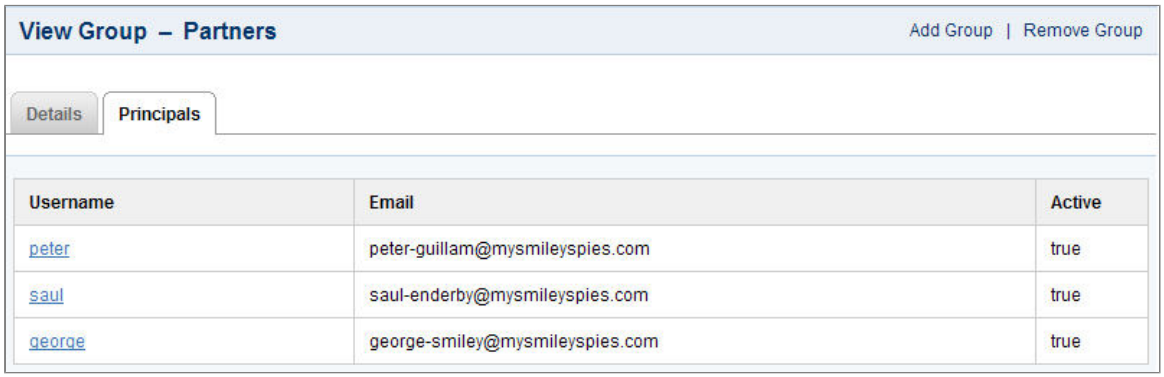
Directory Permissions per Application

- Directory permissions determine whether groups, principals and roles can be added, modified or deleted.
- Before this release, permissions were set at directory level only. Permissions therefore applied across all applications associated with the directory.
- With Crowd 1.2, directory permissions can be set for each application. For example, you could enable the 'Add Principal' permission on the 'Employees' directory for JIRA but disable the permission for Confluence.
- See the screenshot below, and take a look at an example.



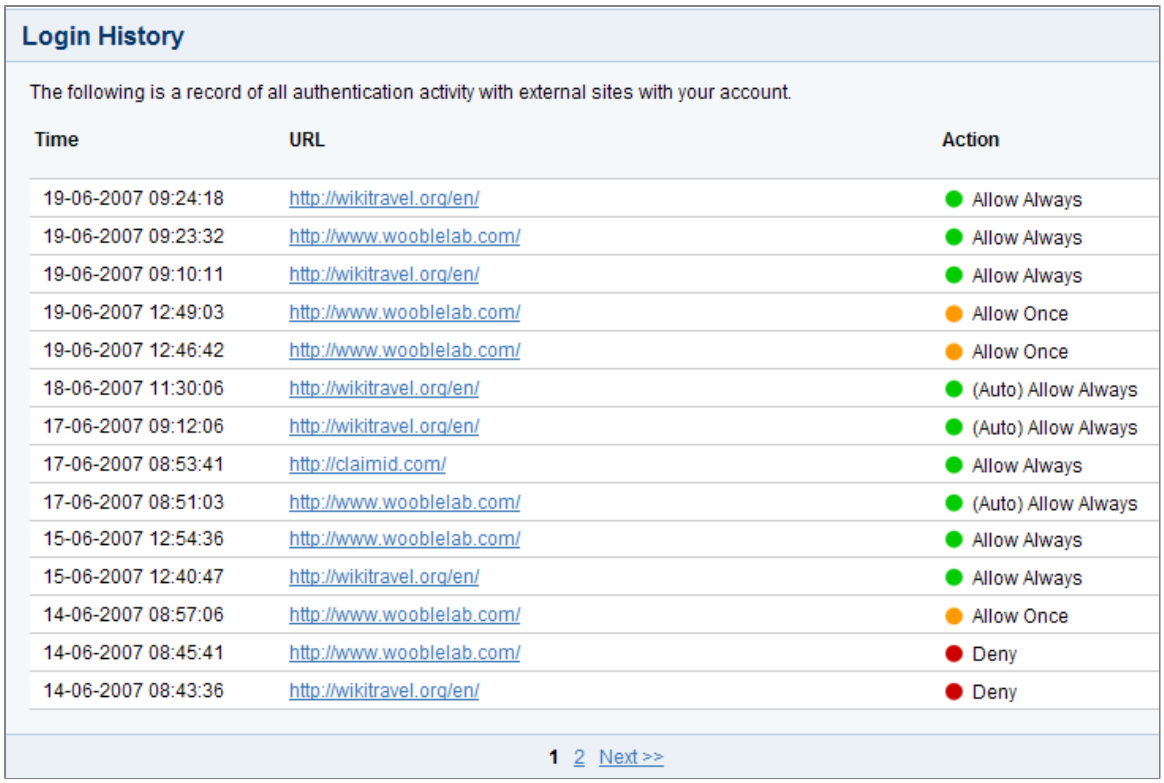
Group and Role Membership Browser

- A new 'Principals' tab in the Group Browser shows all principals belonging to a group.
- You can view membership in the Role Browser too.
- Read the documentation.



Improved Browser for OpenID Login History

- Instead of showing all login history on a single page, the Login History screen now divides the history into pages, for easier viewing.
- To move between pages, click '**Next**', '**Prev**' or a specific page number.
- In the 'Action' column, a new item '**(Auto) Allow Always**' tells you which logins were allowed automatically because of a previous 'Allow Always' instruction.



Login History

The following is a record of all authentication activity with external sites with your account.

Time	URL	Action
19-06-2007 09:24:18	http://wikitravel.org/en/	Allow Always
19-06-2007 09:23:32	http://www.wooblelab.com/	Allow Always
19-06-2007 09:10:11	http://wikitravel.org/en/	Allow Always
19-06-2007 12:49:03	http://www.wooblelab.com/	Allow Once
19-06-2007 12:46:42	http://www.wooblelab.com/	Allow Once
18-06-2007 11:30:06	http://wikitravel.org/en/	(Auto) Allow Always
17-06-2007 09:12:06	http://wikitravel.org/en/	(Auto) Allow Always
17-06-2007 08:53:41	http://claimid.com/	Allow Always
17-06-2007 08:51:03	http://www.wooblelab.com/	(Auto) Allow Always
15-06-2007 12:54:36	http://www.wooblelab.com/	Allow Always
15-06-2007 12:40:47	http://wikitravel.org/en/	Allow Always
14-06-2007 08:57:06	http://www.wooblelab.com/	Allow Once
14-06-2007 08:45:41	http://www.wooblelab.com/	Deny
14-06-2007 08:43:36	http://wikitravel.org/en/	Deny

1 2 Next >>

NTLM Support

- NTLM is a Microsoft authentication protocol that allows you to access a website using your desktop login. The protocol utilises an integration between Microsoft Internet Explorer and Active Directory. When using this feature, users will only need to log in to their desktop to access NTLM-integrated applications.
- JIRA and Confluence NTLM connectors are now supported with Crowd 1.2.
- Read the instructions on setting up Confluence and JIRA NTLM support in Crowd.

Improved Integration with Jive Forums

- Crowd 1.2 provides support for group management in Jive Forums.
- Groups and group memberships are now pulled from Crowd.
- You can use the Jive Forums admin console to define application permissions associated with groups.
- This allows Crowd to manage Jive Forums groups and memberships and Jive Forums to handle the permissions associated with the groups.
- Read the documentation.

Acegi Application Connector

- Crowd 1.2 provides a built-in application connector for Acegi, a security solution with a particular emphasis on Spring Java/JEE applications.
- Read the documentation.

Group-Based Authorisation Added for Subversion

- Crowd allows you to password-protect your SVN repository running under Apache.
- You can now also configure fine-grained access by group as well as by user.
- Read more about the Crowd Subversion connector.

New Importer for Bamboo Users

- Our new Bamboo importer allows you to copy your Bamboo users into a Crowd directory.
- Read the documentation.

Plus Over 70 Improvements and Bug-Fixes

JIRA Issues (77 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-637	NullPointerException in DefaultCookieHandler.setCookie		Resolved
CWD-625	If an allowed Principal Attribute is null it is not possible to update this in LDAP		Resolved
CWD-618	View Principal is throwing a RemoteException when trying to view the Roles of a Principal		Resolved
CWD-617	Browse Principal is not showing an email address for the principal returned.		Resolved
CWD-599	When creating and viewing an LDAP connector, we have been displaying the password as clear text, this should at least be a password field		Resolved
CWD-597	License user-limit check event should not execute for unlimited licenses		Resolved
CWD-593	Upgrade to Atlassian-Extras 1.9		Resolved
CWD-588	Jive Forums remote authentication is not working		Resolved
CWD-582	If the two core event listeners do not exist add them via an upgrade task.		Resolved
CWD-580	Events, EventType and Event Listeners are not being exported as part of the XML backup		Resolved
CWD-579	Role Tab shows the correct number of roles however they all show up as the principal name		Resolved
CWD-578	Allow a crowd administrator to recalculate the user total for a Crowd install		Resolved
CWD-577	Remove Group link on View Principal does not contain a valid directory ID		Resolved
CWD-576	Document Crowd installation on JBoss		Resolved
CWD-575	Document the 'config test' tab		Resolved
CWD-573	Multiple cookies are wrote back to the browser during an authentication.		Resolved
CWD-567	HSQL context path storage issues when not using start_crowd.bat/sh		Resolved
CWD-556	Atlassian applications hang and can not start when integrated with Crowd under the same VM.		Resolved

CWD-552	Data imports fail when no application-group associations are in place.		Resolved
CWD-540	CrowdID Install Documentation Mistake		Resolved
CWD-539	Need and EAR/WAR download to use other application servers		Resolved
CWD-537	Method to create a token for a principal without performing an authentication.		Resolved
CWD-534	Upgrade Crowd to Spring Framework 2.0.6 from 1.2.x		Resolved
CWD-526	Editing groups in Crowd has no effect in Bamboo		Resolved
CWD-525	Login to jira with an existing cookie (non-crowd) shows a nullpointer		Resolved
CWD-524	Full Name attribute (displayName/firstName+surname) used differently by atlassian-user and JiveForums		Closed
CWD-517	Documentation update for 'Upgrading Crowd' as per customer's comment		Resolved
CWD-516	JIRA breaks with retrieveUserMetaProperties NPE after adding user in Crowd		Resolved
CWD-514	Move Crowd to use Webwork 2.2.6		Resolved
CWD-513	Move Crowd to use Seraph 0.9		Resolved
CWD-508	Release Crowd EAR/WAR edition		Resolved
CWD-504	Crowd should be offered as a EAR/WAR package in addition to standalone		Resolved
CWD-503	Cannot modify user profile when using Crowd authentication, fails with NullPointer on RemotePrincipal.getEmail()		Resolved
CWD-502	Unauthenticated user causes session nuking in Crowdified JIRA		Resolved
CWD-501	OpenID history browser		Closed
CWD-500	Directory CRUD permissions on an Application-by-Appliction basis.		Resolved
CWD-497	Crowd integration of Extranet JIRA has authentication problems		Resolved
CWD-496	requiresPasswordChange gets reset to false during login for an InternalDirectory		Resolved
CWD-495	Principals are being added with whitepace in their usernames		Resolved
CWD-492	Concurrent modification exception in JIRAAuthenticator logout code		Resolved
CWD-489	change the Crowd Upgrade Guide to only copy the password from the crowd.properties files, not copy the entire files		Closed
CWD-488	The build.properties file and Ant associated ant task should not overwrite the password attribute in the crowd.properties file		Resolved
CWD-487	The upgrade manager should run after setup is complete		Resolved
CWD-484	When Confluence 2.6 releases we need to move the code from the bamboo-intergration module back into the atlassian-user module.		Resolved
CWD-465	Improve the current Jive integration to provide support for Group management		Resolved

CWD-464	Email address validation is not RFC-2822 compliant		Closed
CWD-462	Implement add user method of OSUser for JIRA		Resolved
CWD-459	Update the SecurityServer SOAP API to enable editing/updating groups		Resolved
CWD-452	JIRA user management should allow admins to update Crowd users		Resolved
CWD-442	View members of the group or role		Resolved
CWD-435	Exception using Seraph single-sign-on in Bamboo		Resolved
CWD-430	CrowdID Not Signing User Attributes Like Nickname or Email		Resolved
CWD-428	Change wording on the Atlassian importer		Resolved
CWD-425	Trim the application address when adding a valid application remote address.		Resolved
CWD-421	Client JARs in client/lib are incomplete		Resolved
CWD-419	displayName attribute is not used with the JIRA connector		Resolved
CWD-417	Libraries in client directory are not enough		Resolved
CWD-415	Tomcat doesn't start if it runs both Crowd and Confluence		Resolved
CWD-414	The CSV Importer needs to display user results for duplicate entries i.e. users that have been ignored since they already exist in Crowd.		Resolved
CWD-407	Textual changes to new CSV-importer screens		Resolved
CWD-398	;jsessionid added to all Crowd links		Resolved
CWD-392	No group integraton into Subversion		Resolved
CWD-390	Browser cookies cause NullPointerException when integrated with Confluence		Resolved
CWD-388	Paging principal sessions links are incorrect and do not function.		Resolved
CWD-380	Sources gets added to download archive		Resolved
CWD-373	Improve the build process for source releases		Resolved
CWD-349	Create a Bamboo to Crowd Principal and Group importer.		Resolved
CWD-348	When switching from internat authentication to Crowd authentication (using seraph?), exception is throw during login.		Resolved
CWD-336	No date sent in email headers for messages sent by Crowd		Closed
CWD-314	Not able to Retrieve Issues (RSS) if JIRA is Integrated with Crowd		Resolved
CWD-297	JIRA performance improvements		Resolved
CWD-281	Build script improvements		Resolved

CWD-249	Adjust build process to publish maven2 client poms.		Resolved
CWD-209	Maven 2 repository for Crowd client components.		Resolved
CWD-185	The import/export is confined to a given instance, we need to make it so the XML file can be used on any Crowd deployment.		Resolved
CWD-135	Support NTLM		Closed
CWD-19	Acegi Connector		Resolved

Crowd 1.1.2 Release Notes

The Crowd development team has released Crowd 1.1.2.

Crowd 1.1.2 is a **recommended upgrade** from Crowd 1.1.1 since it provides improved integration with JIRA and Confluence, and tidier functionality for multiple directories.

For cross product compatibility, you must upgrade to the following versions of each product:

- Crowd 1.1.2 or later
- Bamboo 1.2.2 or later
- Confluence 2.5.6 or later
- JIRA 3.7.4 or later

Integration with JIRA user management

With Crowd 1.1.2, you can now turn external user management **off** in JIRA. This means that you can allow signup via JIRA, and you can manage your users within JIRA. Changes will flow through to Crowd.

JIRA has an automatic group membership feature. This means that any new user added through JIRA will automatically be a member of all groups which have the **JIRA Users** permission. In this way, you can ensure that a new user is automatically added to several groups when they sign up with JIRA.

RSS feeds

Crowd 1.1.2 fixes the problem experienced accessing RSS feeds from JIRA including retrieving JIRA issues via Confluence macros (e.g. the JIRA portlet macro).

Improved LDAP Performance

Crowd 1.1.2 now queries LDAP repositories in a more efficient manner that will give a dramatic performance increase for those with large numbers of LDAP groups.

Other Fixes in Crowd 1.1.1

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (23 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-540	CrowdID Install Documentation Mistake		Resolved
CWD-503	Cannot modify user profile when using Crowd authentication, fails with NullPointer on RemotePrincipal.getEmail()		Resolved
CWD-497	Crowd integration of Extranet JIRA has authentication problems		Resolved
CWD-496	requiresPasswordChange gets reset to false during login for an InternalDirectory		Resolved
CWD-495	Principals are being added with whitepace in their usernames		Resolved
CWD-492	Concurrent modification exception in JIRAAuthenticator logout code		Resolved

CWD-487	The upgrade manager should run after setup is complete		Resolved
CWD-484	When Confluence 2.6 releases we need to move the code from the bamboo-intergration module back into the atlassian-user module.		Resolved
CWD-478	Update Confluence Integration Doc		Resolved
CWD-472	OpenID not working with LiveJournal		Resolved
CWD-462	Implement add user method of OSUser for JIRA		Resolved
CWD-452	JIRA user management should allow admins to update Crowd users		Resolved
CWD-448	Remote application's calls to removePrincipal(name) only removes the first principal it finds		Resolved
CWD-447	Remote application's calls to removeRole(name) only removes the first role it finds		Resolved
CWD-446	Remote application's calls to removeGroup(name) only removes the first group it finds		Resolved
CWD-421	Client JARs in client/lib are incomplete		Resolved
CWD-420	Configuring multiple repositories may result in duplicate users		Resolved
CWD-394	Full Name Search always returns all users		Closed
CWD-390	Browser cookies cause NullPointerException when integrated with Confluence		Resolved
CWD-383	misspelling in wsdl - encryptedCrednetial		Resolved
CWD-314	Not able to Retrieve Issues (RSS) if JIRA is Integrated with Crowd		Resolved
CWD-297	JIRA performance improvements		Resolved
CWD-132	Windows service registration feature.		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.1.1 Release Notes

The Crowd development team has released Crowd 1.1.1.

This release is a **highly recommended upgrade** from Crowd 1.1.0 since it provides a security fix to XWork, the technology underlying the web framework WebWork which is used by Crowd.

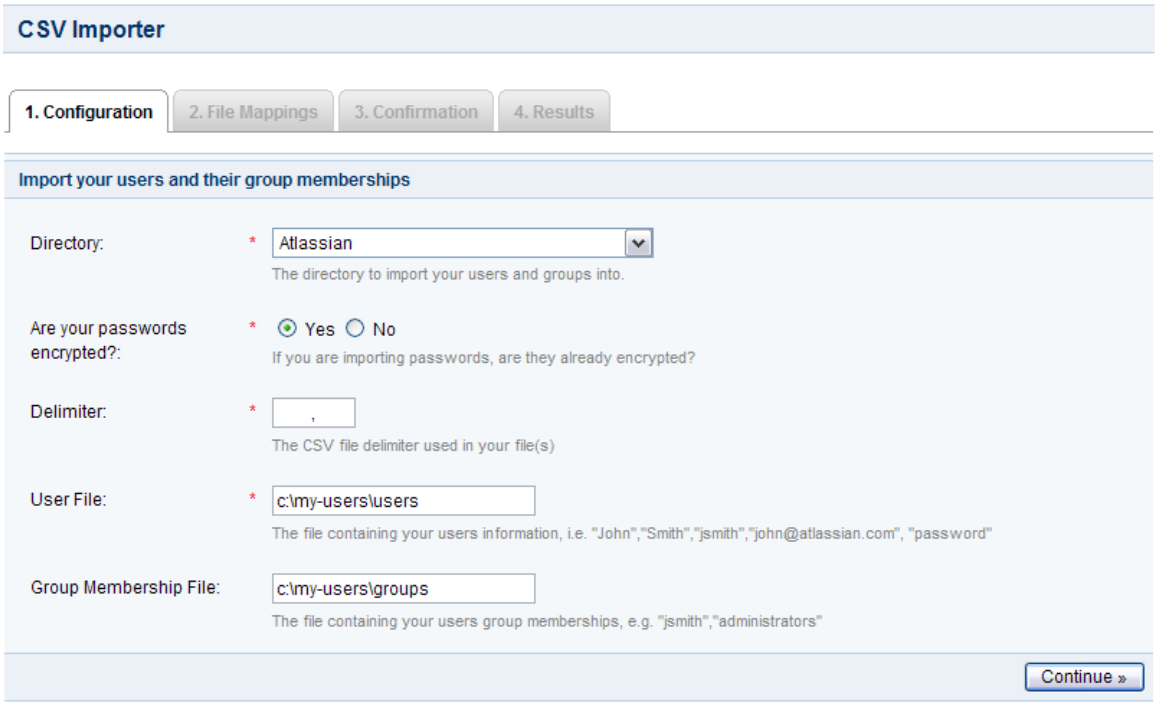
This release also contains a new CSV importer as well as fixes for some issues found in 1.1.0.

Importing Users and Groups from a CSV File

You can now copy users from an external directory or user base into Crowd via a CSV (comma-separated values) file.

The new CSV Importer allows you to specify a file containing user information, and optionally another file containing the groups to which the users belong. You can then map the CSV fields to the Crowd directory fields. After performing the import, Crowd sums up the results.

Screenshot: 'CSV Importer - Configuration'



Other Fixes in Crowd 1.1.1

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (20 issues)

Key	Summary	Priority	Status
CWD-445	Internal Directory search for Group by name is failing to aggregate the correct members		Closed
CWD-438	Users shown twice in JIRA		Closed
CWD-437	JIRA's logout via SSO does not clear it's session		Resolved
CWD-435	Exception using Seraph single-sign-on in Bamboo		Resolved
CWD-434	Searching for a group spanning multiple directories by its name will not amalgamate the principals		Resolved
CWD-428	Change wording on the Atlassian importer		Resolved
CWD-425	Trim the application address when adding a valid application remote address.		Resolved
CWD-419	displayName attribute is not used with the JIRA connector		Resolved
CWD-418	Chained directories are returning multiple groups/roles rather than aggregating group names.		Resolved
CWD-414	The CSV Importer needs to display user results for duplicate entries i.e. users that have been ignored since they already exist in Crowd.		Resolved
CWD-407	Textual changes to new CSV-importer screens		Resolved
CWD-400	JIRA attach screenshot does not write file to the filesystem when Crowdified.		Resolved
CWD-397	Document the CSV importer		Resolved

CWD-388	Paging principal sessions links are incorrect and do not function.		Resolved
CWD-385	Generated tokens have non-HTML escaped characters.		Resolved
CWD-382	Create custom add successful page does not display directort connector page.		Resolved
CWD-352	Configure the number of paged results for an LDAP connector		Resolved
CWD-309	The SunOne LDAP connector is not correctly authenticating users		Resolved
CWD-290	Upgrade webwork from 2.2.4 to 2.2.5		Resolved
CWD-53	CSV importer		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.1.0 Release Notes

The Atlassian Crowd team is proud to announce the release of Crowd 1.1.

This release contains a whole host of new features targeted at implementing OpenID, along with core updates to the Crowd Administration Console.

OpenID-enable your organisation with CrowdID

OpenID enables you to use a centralised identity to login to any website that supports OpenID. It opens up the possibilities of massive scale cross-domain SSO.

Think about all the accounts you have online: blogs, wikis, to-do lists, photo galleries. The list is endless. Even simple tasks such as leaving comments on someone else's blog may require you to register an account with that particular blogging system. This leaves you, as an end user, to set up and manage numerous accounts on each of these sites. With OpenID, rather than managing all these disparate accounts individually, users can manage their identity in one place via an authentication server.

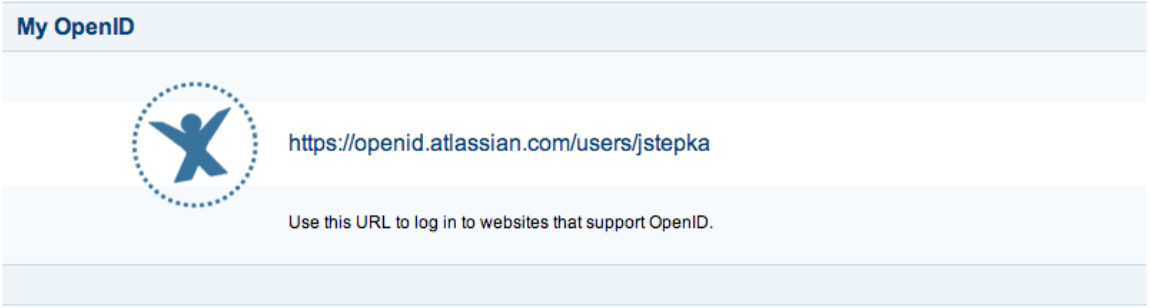
With the ever-increasing adoption of this open authentication framework, including names such as Microsoft, AOL, Sun, Verisign and Firefox, expect to see many applications enabled for OpenID authentication.

CrowdID offers OpenID to an organisation's user base, allowing users to manage their online identity. Everything from configuring different profiles, managing trusted sites to reviewing authentication activity, is accessible from CrowdID. Administrators can set up whitelists/blacklists so that only trusted hosts can request authentication and can set up secure communication via SSL. All of the users can be managed via Crowd's security server, utilizing LDAP services from products such as Microsoft Active Directory.

Included with CrowdID is a sample OpenID client application, providing a working example of an OpenID enabled application. This will help developers kick start OpenID-enabling their applications.

Using OpenID

Rather than registering and typing in your username and password on each site that you visit, OpenID allows you to type a URL similar to 'openid.mycompany.com/users/jstepka':

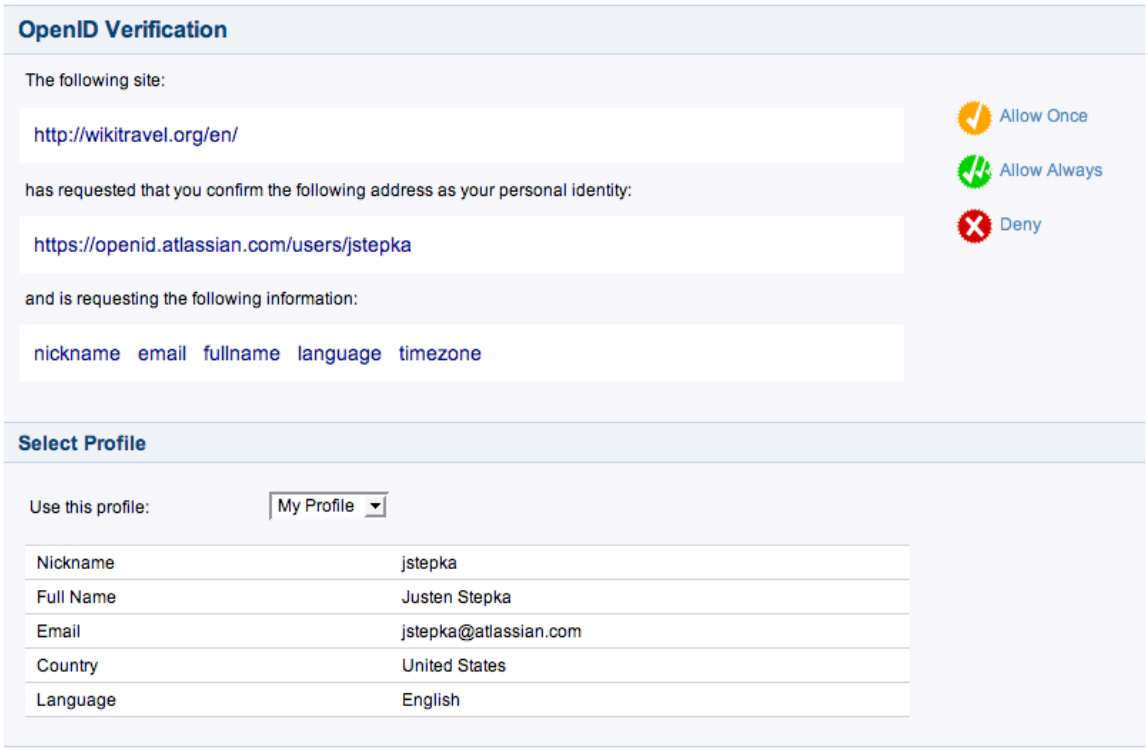


The OpenID website that you are logging in to will redirect you to CrowdID, which will ask you if you would like to allow authentication with the requesting site.

You can even choose to 'Always' allow authentication with particular OpenID sites, which allows pass-through authentication if you are already logged into your CrowdID server. If you do this, then when you visit the site later, simply provide your URL (e.g.

'openid.mycompany.com/users/jstepka') and you are in.

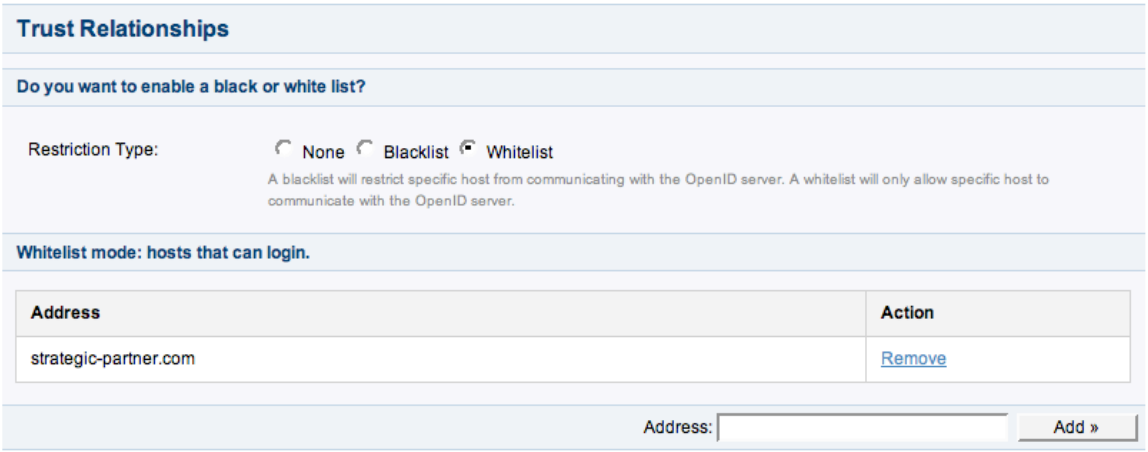
Think of it as 'Remember Me' for the whole internet!



'Blacklist' and 'Whitelist'

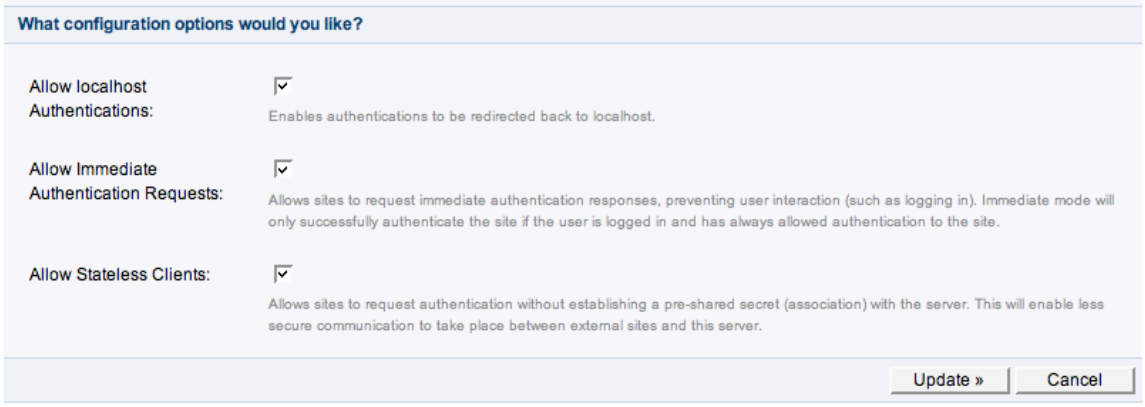
'Blacklists' and 'whitelists' allow administrators to lock down CrowdID their server so that, if necessary, it can only communicate with trusted hosts with which you have established relationships.

A blacklist will prevent specific hosts from communicating with the OpenID server. A whitelist will allow only specific hosts to communicate with the OpenID server.



OpenID Advanced Options

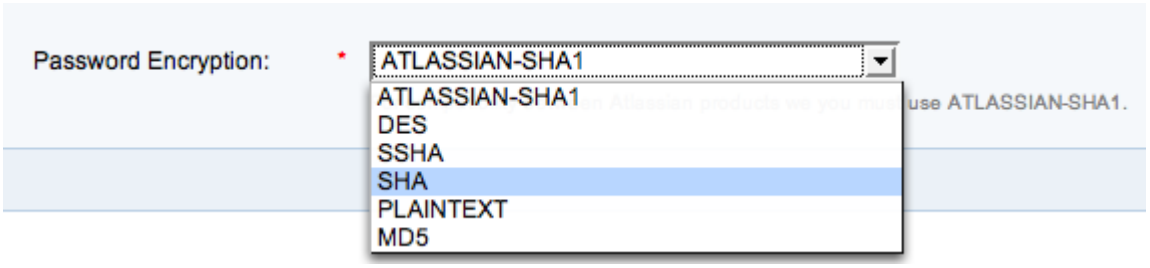
Some external sites implement security better than others. With CrowdID, you can pick how tough you want to be on OpenID sites that communicate with your Crowd OpenID server.



Crowd Console and Server Updates

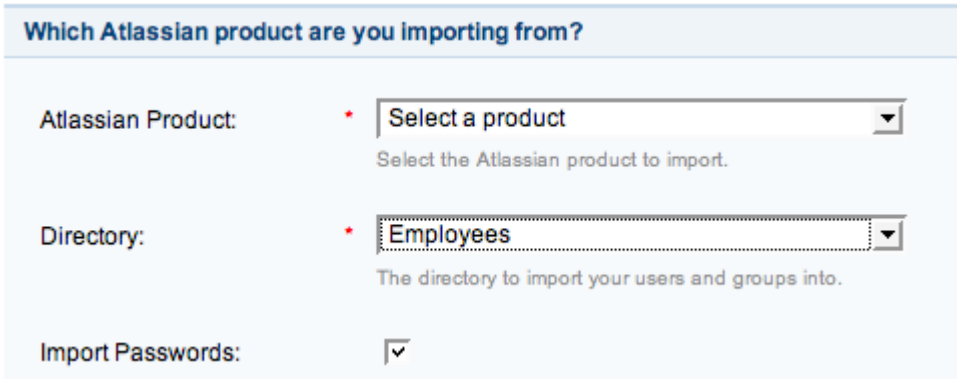
Choose Your Encryption Type

Every administrator has their own password policies. When using a Crowd Internal Directory you can now select the level of encryption you need.



Import Your JIRA and Confluence Passwords

Migration can be a pain. To ease your switch from existing Atlassian products, Crowd can now import your existing passwords!



Faster Web-Services

Crowd web-services now support GZip compression, improving the performance when downloading large amounts of data such as the all the members of a large group or when performing large search.

Improved Apache and Subversion Integration

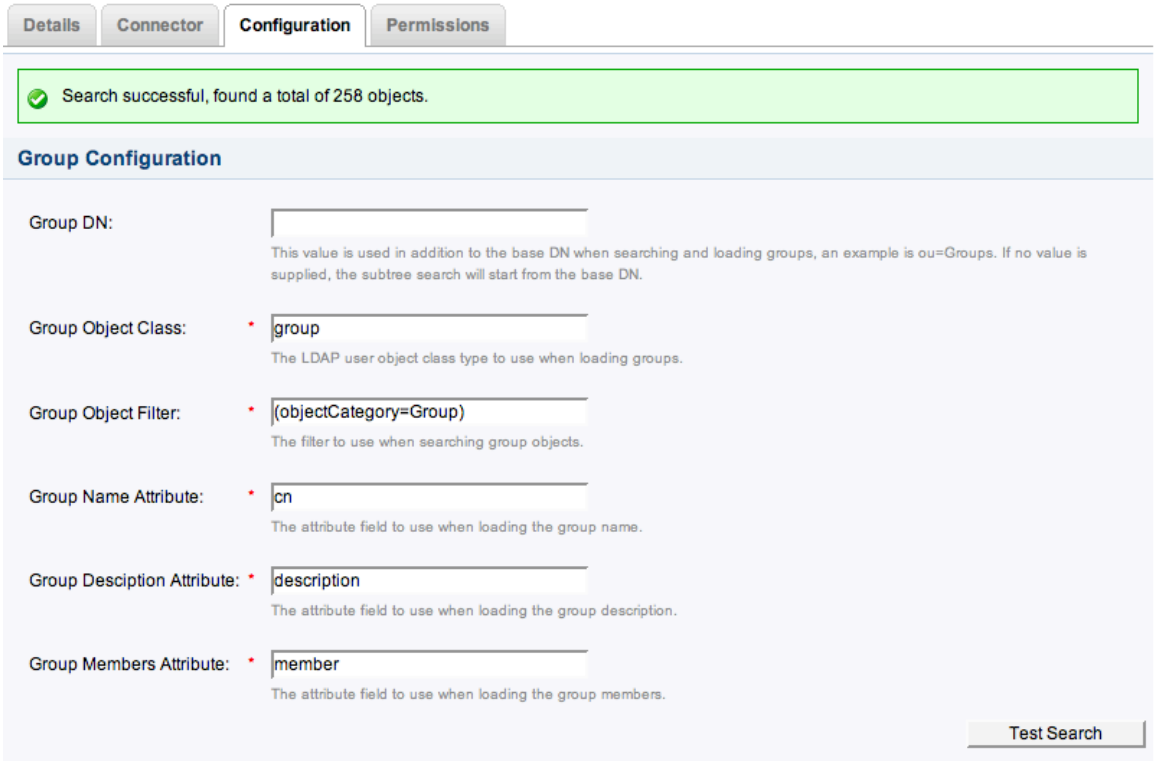
The Apache and Subversion library performance has been improved with the implementation of client-side caching of approved authentication requests.

Jive Forums 5.5 Support

The Jive Forums centralised authentication connector has been updated to support the new 5.5 major release of Jive Forums.

LDAP Configuration Tester

When setting up a Crowd LDAP connection you can now verify that your configuration connects as expected.



JIRA Issue Tracker

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (50 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-379	Change Password link on openid.atlassian.com throws 'No Action' error page		Resolved
CWD-377	Updating an Application will update the password for an application, even when you do not type in a new password		Closed
CWD-376	Export fails when an application does not have a description.		Resolved
CWD-368	Stray backslash on Groups administration screen		Resolved
CWD-365	Typo in hint for Password Encryption during initial directory setup		Resolved
CWD-360	ORA-01000: maximum open cursors exceeded		Resolved
CWD-359	'Blacklist' and 'Whitelist' options display intermittently in IE		Resolved
CWD-354	suggestions for the OpenID login page		Resolved
CWD-351	When logging out of Bamboo and anonymous mode is turned off, users still have the ability to create plans etc.		Resolved
CWD-343	Atlassian-user integration - get display name attribute from attributes if there rather than building display name adhoc.		Resolved
CWD-332	Test configuration buttons when creating an LDAP directory connector.		Resolved
CWD-325	Directory details tab shows empty pink error box		Resolved
CWD-323	Test connection utility for LDAP servers.		

			Resolved
CWD-320	Improve the importing of users from Confluence and JIRA so these users do not need to reset their passwords		Resolved
CWD-319	The export function of Crowd needs to have a flag to say don't export domain.		Resolved
CWD-318	ApacheDS crowd integration does not currently support the adding of groups		Resolved
CWD-313	The Apache module needs some kind of cache implemented similar to our other 'clients', to help improve performance around apache integration		Resolved
CWD-305	Add optional GZIP compression support for XFire SOAP services and client.		Resolved
CWD-304	Auto configure openid server as part of the setup process.		Resolved
CWD-302	Skin the OpenID Server		Closed
CWD-301	OpenID Client - Dummy Mode		Resolved
CWD-300	OpenID Server - dummy mode		Resolved
CWD-299	OpenID Client - Check Immediate		Resolved
CWD-298	OpenID Server - Check Immediate		Resolved
CWD-294	Test OpenIDClient Form Redirection		Resolved
CWD-292	OpenID Server Implementation		Resolved
CWD-291	Auto configure openid server as part of the setup process.		Closed
CWD-290	Upgrade webwork from 2.2.4 to 2.2.5		Resolved
CWD-288	Change application titles - not footers		Resolved
CWD-286	Skin Demo RP application		Resolved
CWD-285	Display attributes in the demo application upon login (store in session for display)		Resolved
CWD-284	Login and Logoff for OpenID demo relying party application.		Resolved
CWD-283	Configure request attributes for demo app		Resolved
CWD-280	Document OpenID server configuration		Closed
CWD-279	Attribute/Profile Management		Resolved
CWD-278	Authentication redirect from relying party.		Resolved
CWD-277	Skin Server		Resolved
CWD-276	Profile authentication history		Resolved
CWD-275	Enable/disable localhost relying parties.		

			Resolved
CWD-274	Whitelist and Blacklist Editor		Resolved
CWD-273	Force Association		Resolved
CWD-272	Reset password option.		Resolved
CWD-271	Login and Logoff for OpenID Server application.		Closed
CWD-269	document the management of the Crowd domain during setup and in the Console		Resolved
CWD-246	Update documentation with new information about installing connector for 5.5.X version of JIVE.		Resolved
CWD-245	Jive Forums 5.5 Support		Resolved
CWD-232	add 'SecurityServerClient'		Resolved
CWD-154	Apache DS connector		Resolved
CWD-144	Add 'green' success message to 'update' actions on Console.		Resolved
CWD-65	Explore OpenID support		Closed

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.7 Release Notes

The Crowd development team has released Crowd 1.0.7.

This release is a **highly recommended upgrade** from Crowd 1.0.6 and fixes 2 major issues found in 1.0.6:

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (5 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-316	Active Directory principals can signin with a blank password		Resolved
CWD-296	LDAP update password implementation.		Resolved
CWD-287	Reset password option for the Console		Resolved
CWD-233	javadoc SecurityServer		Resolved
CWD-181	Continually asked to re-auth with Apache		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.6 Release Notes

The Crowd development team has released Crowd 1.0.6.

This build is a quick fix for problems reported with the SSO integration for multi host environments:

You can now download Crowd from http://www.atlassian.com/Crowd

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (3 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-265	Confluence displays the users fullname instead of email when integrated with Crowd		Resolved
CWD-263	Fails with exception on Search		Resolved
CWD-262	Improve the management of the Crowd domain during setup and in the Console.		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.5 Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released Crowd 1.0.5.

If you are running Confluence version 2.4.4 or before, you will need to upgrade the confluence/WEB-INF/lib/atlassian-user-XXXX-XX-XX.jar Atlassian User library to version 2007-04-05. The original library file will need to be backed up, removed, and then replaced with the new version listed above.

This build is mix of bug fixes, documentation improvements, and feature enhancements:

You can now download Crowd from http://www.atlassian.com/Crowd

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (15 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-259	Username is not displayed in Confluence (2.4.X) when first logging in.		Closed
CWD-258	Domain for multihost single sign-on is not setting the cookie correctly.		Closed
CWD-257	VerifyTokenFilter missing from the Demo application.		Closed
CWD-256	Importer success screens display success even on an exception.		Resolved
CWD-254	review Installation documentation		Resolved
CWD-252	Active Directory filter does not exclude accounts which are no sAMAccountName type.		Closed
CWD-248	CLONE -The Sitemesh and Webwork cleanup filters are being wrapped around the XFire requests.		Closed
CWD-244	Set compile flags with maven build scripts to be vs. 1.4		Resolved
CWD-243	Document how you can not delete the Crowd console.		Resolved
CWD-242	You can delete the integrated Crowd application		Resolved
CWD-235	System error when no directory is selected when adding a group		Resolved
CWD-234	Add Websphere installation notes for Crowd.		Resolved
CWD-229	Transactions wrapping transactions. The transaction manager is not aware about the wrapping transaction.		Resolved
CWD-226	browser window title should say 'View Application'		Resolved
CWD-222	Crowd is not handling latin1 characters correctly		

			Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.4 Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released Crowd 1.0.4.

This build focused on bug fixes:

- Import export process was failing with Oracle DB.
- Implemented updating known attribute types on an LDAP object..
- Importing JIRA users is fixed for MySQL on a Unix like filesystem.

You can now download Crowd from http://www.atlassian.com/Crowd

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (6 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-225	Import and export of Crowd fails when the database is Oracle		Closed
CWD-221	Add documentation (marketing) section for Apache Directory Server		Closed
CWD-220	Implement RemoteDirectory updatePrincipal(RemotePrincipal) method for LDAP servers using InetOrgPerson as the Principal object.		Resolved
CWD-213	The Sitemesh and Webwork cleanup filters are being wrapped around the XFire requests.		Resolved
CWD-206	JIRA User Import Doesn't Set Groups on Principals		Resolved
CWD-172	Remove this error: SEVERE: No Store configured, persistence disabled		Resolved

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.3 Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released Crowd 1.0.3.

This build is a mix of new features, bugs fixes and feature improvements:

- Improved SSO integration with Seraph for JIRA, Confluence and Bamboo.
- First builds of Apache Directory Server connector.
- Now supports directory server version that do not have the paged ldap control.
- Documentation updates.

You can now download Crowd from http://www.atlassian.com/Crowd

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (9 issues)			
Key	**Summary**	**Priority**	**Status**

CWD-218	When an application is searching for its members from an LDAP repo AND an Internal Directory a HibernateException is thrown around trying to persist elements in a RemoteGroup.members		Closed
CWD-216	Crowd session token should be unique for each user, directory, machine		Closed
CWD-214	Login should logout any previous logged in users before a new login		Closed
CWD-179	Paged results control option for LDAP connectors.		Closed
CWD-177	Fisheye connector logs unnecessary exception.		Closed
CWD-175	Computers show up in the Principal list within Crowd from MSAD		Closed
CWD-169	NullPointerException on add OpenLDAP directory		Closed
CWD-163	Administration Console allows login of unauthorized users		Closed
CWD-121	Setting a "Remember Me" flag in Confluence, JIRA or Bamboo does not work, since the Token Reaper 'reaps' all session when the timeout is reached		Closed

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.2 Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released Crowd 1.0.2.

This addresses bugs and feature improvements which can be viewed through our JIRA issue tracker:

- Included missing libraries for build archive.
- Added logging for input and output operations on SOAP services.
- Improved Jira caching for Crowd data.
- Added support for SSO beyond centralised authentication for Jive Forums.

You can now download Crowd from http://www.atlassian.com/Crowd

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (6 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-199	Missing libraries from the Crowd distribution		Closed
CWD-198	I renamed the docs from "Documentation" to "Crowd Documentation" (sorry). Can you please fix the "Help link?		Closed
CWD-197	XFire service input and output logging.		Closed
CWD-196	Improve the ability to configure the internal cache's used by the Crowd client and the Crowd console		Closed
CWD-195	Implement SSO for Jive Forums		Closed
CWD-193	Download archive is missing wsdl4j-1.5.2.jar		Closed

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.1 Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released Crowd 1.0.1.

This addresses 3 critical bugs which can be viewed through our JIRA issue tracker:

- Create new group/role broken using OpenLDAP.
- XFireFault exception: "No write method for property".
- Single sign on Seraph authentication fails when the host on a domain is not the same.

You can now download Crowd from http://www.atlassian.com/Crowd

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (3 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-190	XFireFault exception: "No write method for property".		Closed
CWD-189	Create new group/role broken using OpenLDAP		Closed
CWD-82	Single sign on Seraph authentication fails when the host on a domain is not the same.		Closed

Cheers,

The Atlassian Crowd Development Team

Crowd 1.0.0 Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released Crowd 1.0.

This addresses bugs which can be viewed through our JIRA issue tracker:

- UI improvements with new screen layouts.
- Import and Export process for XML.
- LDAP Fixes for OpenLDAP and Microsoft Active Directory.
- Improved error reporting.
- Apache / Subversion support.

You can now download Crowd from http://www.atlassian.com/Crowd. If upgrading from a previous version, please follow the Upgrade Guide.

Errors were reported by the JIRA trusted connection.

- APP_UNKNOWN; Unknown Application: {0}; ["confluence:4557196"]

JIRA Issues (10 issues)			
Key	**Summary**	**Priority**	**Status**
CWD-188	License update (when invalid) page should detail current license details.		Closed
CWD-184	Make Crowd's internal exception extend NestableException from commons-lang		Closed
CWD-180	Schema violation with LDAP and Groups/Roles		Closed
CWD-178	LDAP flags are incorrect for Active Directory/LDAP (Win2k3 domain)		Closed
CWD-173	Implement an import and export function in Crowd		Closed
CWD-150	Build fails		Closed
CWD-101	Unable to upgrade from 0.2 to 0.3.3		Closed
CWD-97	Apache mod Crowd integration		Closed
CWD-90	sso support for fisheye		Closed

CWD-62	500 page.		Closed

Cheers,

The Atlassian Crowd Development Team

Crowd 0.4.5 Beta Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.4.5.

This addresses bugs which can be viewed through our JIRA issue tracker:

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12652

- Improved Active Directory LDAP attribute filtering.
- UI improvements with new screen layouts.
- Spring TX management.

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.4.4 Beta Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.4.4.

This addresses bugs which can be viewed through our JIRA issue tracker:

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12642

- Caching improvement for Confluence.
- Removed an additional attribute that was causing integration problems with SOAP services when using Active Directory.

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.4.3 Beta Release Notes

> Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.4.3.

This addresses bugs which can be viewed through our JIRA issue tracker:

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12267

- Support for AD when there are more than 999 records in a search result.
- Reduced the number of necessary libs for a client application.
- Improved the 'build.properties' file configuration.

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.4.2 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.4.2.

This addresses bugs which can be viewed through our JIRA issue tracker:

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12623

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.4.1 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.4.1.

This addresses bugs which can be viewed through our JIRA issue tracker:

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12600

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.4 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.4.

This release addresses several critical issues:

- Seraph Logout code fails to logout the user in Confluence, Bamboo and JIRA.
- Unable to search for a Principal by email address.
- Accept header authentication factor unreliable with Mozilla based browsers.
- Default 'localhost' configuration not added valid IP address of 127.0.0.1.

New features include:

- Allow all to authenticate.
- New LDAP connectors build off Spring LDAP Template with better performance enhancements.
- Support for LDAP filters

All Postgres DB will need to have the following command ran:

```
alter table "APPLICATIONDIRECTORIES" add column "ALLOWALLTOAUTHENTICATE" boolean;
```

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12266

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.3.3 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.3.3.

This release addresses the following:

- Upgrade from Webwork 1 to Webwork 2
- Workaround for Active Directory to support CN forwards.

CRITICAL POSTGRES UPGRADE NOTES: http://jira.atlassian.com/browse/CWD-71

We started testing on IE7 and have noticed the CSS bugs and will work to get this addressed for the next build.

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12544

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.3.2 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

The Crowd development team has released a new version of Crowd - 0.3.2.

This release addresses a Seraph SSO issue when integrating JIRA, Confluence and Bamboo.

http://jira.atlassian.com/secure/IssueNavigator.jspa?reset=true&pid=11291&fixfor=12540

You can now download Crowd from http://www.atlassian.com/Crowd

Cheers,

The Atlassian Crowd Development Team

Crowd 0.3 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

Crowd 0.3

- Standalone version - Tomcat 5.5 with HSQL - .zip (65.3 Mbs)
- Standalone version - Tomcat 5.5 with HSQL - .tar.gz (64.7 Mbs)

Points of Interest

- The focus of this distribution is on performance for a large number of users and groups when integrating JIRA, Confluence and Bamboo integration.

Crowd 0.2 Beta Release Notes

Crowd 2.0.3 has now been released — see the Crowd 2.0.3 Release Notes.

Crowd 0.2

- Standalone version - Tomcat 5.5 with HSQL - .zip (59.5Mbs)
- Standalone version - Tomcat 5.5 with HSQL - .tar.gz (59.7Mbs)

Points of Interest

- There is an error when unzipping on the Windows platform, the archive integrity is fine and this will be fixed for the 0.3 release.
- The focus of this distribution is for JIRA and Confluence integration. Performance enhancements will be added for the 0.3 release which will allow large user-databases to be integrated.

Installing Crowd

Installing Crowd

You can download Crowd here.

> **Warning: Some unzip programs cause errors**
> Some archive-extract programs cause errors when unzipping the Crowd archive file.
>
> - **Linux** or **Unix** users can use any unzip program.
> - **Solaris** users must use GNU Tar instead of Solaris Tar.
> - **Windows** users should use a third-party unzip program like 7Zip or Winzip. If you do not have one, please download and install one before continuing:
>   - 7Zip — Recommended. If in doubt, download the '32-bit .exe' version
>   - Winzip

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

RELATED TOPICS

- Crowd Release Notes
- Installing Crowd
- Upgrading Crowd
- Migrating Crowd between Servers

Supported Platforms

This page describes the supported platforms and hardware requirements for **Crowd 2.0.x**.

Key: ✓ = Supported. ✗ = Not Supported

Java Version	
JDK (1)	✓ 1.6, 1.5 ✗ 1.4
Operating Systems	
Microsoft Windows (2)	✓
Linux / Solaris (2)	✓
Apple Mac OS X (2)	✓
Application Servers	
Apache Tomcat	✓ 6.0.x (Crowd ships with Apache Tomcat 6.0.20) ✓ 5.5.x (Tested on 5.5.26)

Databases	
MySQL (3)	✅ 5.0.37 and later
Oracle	✅ 10g (Tested on 10.2.0.1.)
PostgreSQL	✅ 8.x, 7.x
Microsoft SQL Server	✅ 2008, 2005
HSQLDB (4)	✅ (For evaluation only.)
Web Browsers	
Microsoft Internet Explorer (Windows)	✅ 8, 7 ❌ 6
Mozilla Firefox (all platforms)	✅ 3.x ❌ 2.x
Safari	✅ 4.x
Opera	❌

Notes:
1. JDK:

- It is not enough to have the JRE only. Please ensure that you have the full JDK. You can download the Java SE Development Kit (JDK) from the Sun website.
- Once the JDK is installed, you will need to set the **JAVA_HOME** environment variable, pointing to the root directory of the JDK. Some JDK installers set this automatically (check by typing 'echo %JAVA_HOME%' in a DOS prompt, or 'echo $JAVA_HOME' in a shell). If it is not set, please see Setting JAVA_HOME.

2. Operating systems: Crowd is a pure Java application and should run on any platform provided the Java runtime platform requirements are satisfied.

3. MySQL: Please ensure that you set transaction isolation to 'read-committed' instead of the default 'repeatable-read', as described in the database configuration guide.

4. HSQLDB: Crowd ships with a built-in HSQL database, which is fine for evaluation purposes but is somewhat susceptible to data loss during system crashes. For production environments we recommend that you configure Crowd to use an external database.

Vote for more supported application servers
If you are interested in support for other application servers, please make your requests via our issue tracker. In particular, you can vote for the following existing requests:

- CWD-1192 — Provide support for versions of Resin newer than 3.0.26.
- CWD-950 — Provide official support for Websphere.

Hardware Requirements

The hardware required to run Crowd depends significantly on the number of applications and users that your installation will have, as well as the maximum number of concurrent requests that the system will experience during peak hours.

During evaluation Crowd will run well on any reasonably fast workstation computer (eg. 1.5+Ghz processor). Memory requirements depend on how many applications and users you will store, but 256MB is enough for most evaluation purposes.

Most users start by downloading Crowd, and running it on their local computer. It is easy to migrate Crowd to your enterprise infrastructure later.

We would appreciate if you let us know what hardware configuration works for you. Please create a support request in JIRA with your hardware specification and mention the number of applications and users in your Crowd installation.

While some of our customers run Crowd on SPARC-based hardware, Atlassian only officially supports Crowd running on x86 hardware and 64-bit derivatives of x86 hardware.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
  - Connecting CrowdID to a Database
  - Installing Crowd and CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

Setting JAVA_HOME

Once you have installed the JDK (see Supported Platforms), you need to set the JAVA_HOME environment variable.

To set the JAVA_HOME environment variable on Windows

1. Right click on the '**My Computer**' icon on your desktop and select '**Properties**'.
2. Click the '**Advanced**' tab.
3. Click the '**Environment Variables**' button.
4. Click '**New**'.
5. In the '**Variable name**' field, enter 'JAVA_HOME'.
6. In the '**Variable value**' field, enter the directory (including its full path) where you installed the JDK.
7. Restart the computer.

To set the JAVA_HOME environment variable on 'nix based systems

There are many ways you can do it on 'nix based systems (including Mac OS X). Here are two:

For your current user,

1. Open up a shell / terminal window
2. `vi ~/.profile` (replace vi with your favourite text editor)
3. Add `export JAVA_HOME=/path/to/java/home/dir` on its own line at the end of the file
4. Add `export PATH=$JAVA_HOME/bin:$PATH` on its own line immediately after
5. Save, and restart your shell
6. Running `java -version` should give you the desired results

For all users in the system,

1. Open up a shell / terminal window
2. `vi /etc/profile` (replace vi with your favourite text editor)
3. Add `export JAVA_HOME=/path/to/java/home/dir` on its own line at the end of the file
4. Add `export PATH=$JAVA_HOME/bin:$PATH` on its own line immediately after
5. Save, and restart your shell
6. Running `java -version` should give you the desired results

If you are using a GUI, you may not need to open up the shell. Instead, you might be able to open the file directly in a graphical text editor.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
    - HSQLDB
    - MS SQL Server
    - MySQL
    - Oracle
    - PostgreSQL
  - Connecting CrowdID to a Database
    - HSQLDB for CrowdID

        - MS SQL Server for CrowdID
        - MySQL for CrowdID
        - Oracle for CrowdID
        - PostgreSQL for CrowdID
    - Installing Crowd and CrowdID WAR Distribution
        - Installing Crowd WAR Distribution
            - Configuring Crowd and CrowdID on Tomcat
            - Installing Crowd WAR on JBoss
        - Installing CrowdID WAR Distribution
    - Specifying your Crowd Home Directory
- Running the Setup Wizard
    - Troubleshooting your Configuration on Setup
- Configuring Crowd
    - Important Directories and Files
        - The crowd.properties File
    - Changing the Port that Crowd uses
    - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
    - Specifying Startup Order of Windows Services
    - Changing the User for the Crowd Windows Service
    - Removing the Crowd Windows Service
    - Troubleshooting Crowd as a Windows Service

Installing Crowd and CrowdID

The instructions below tell you how to install the **standalone distribution** of Crowd, which includes Apache Tomcat. If you wish to deploy a WAR distribution of Crowd or CrowdID on your own existing application server instead, read the instructions on the Crowd WAR distribution.

Crowd versions 1.1 and later include **CrowdID**. Installing Crowd, as described below, will also install CrowdID.

Hint: If you are evaluating Crowd or you are unsure which version to install, just follow the simple instructions on this page.

On this page:

- 1. Install Crowd (Standalone Distribution)
- 2. *Optional* Prepare your Database
- 3. Start Crowd and Complete the Setup Wizard
- Next Steps

1. Install Crowd (Standalone Distribution)

1. Download Crowd.
2. Please check your unzip program before extracting the downloaded archive – see the note on the Crowd installation front page.
3. Unzip the download archive into a directory of your choice. Note: Do not specify directory names that contain spaces.
   We'll refer to this installation directory as **{CROWD_INSTALL}**.
4. Specify your Crowd Home directory by editing the configuration file at: {`CROWD_INSTALL}\crowd-webapp\WEB-INF\classes\crowd-init.properties`.

   The **Crowd Home** directory is where Crowd will store its configuration information. If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will also store its database in this directory. To specify the directory:

   - Open the `crowd-init.properties` file.
   - Choose the appropriate line in the file, depending upon your operating system (see below).
   - Remove the # at the beginning of the line.
   - Enter the name of the directory you want Crowd to use as its Home directory. For example,
     - On Windows:

       ```
       crowd.home=c:/data/crowd-home
       ```

       Note: On Windows, make sure you use forward slashes as shown above, not backward slashes.

     - On Mac and UNIX-based systems:

       ```
       crowd.home=/var/crowd-home
       ```

- Save the `crowd-init.properties` file.

2. *Optional* Prepare your Database

Hint: If you are evaluating Crowd and are happy to use the database supplied, you can skip this step.

If you wish to set up Crowd and/or CrowdID with an external database, see:

- Connecting Crowd to a Database
- Connecting CrowdID to a Database

3. Start Crowd and Complete the Setup Wizard

1. Run the start-up script, found in your {CROWD_INSTALL} directory:
   - `start_crowd.bat` for Windows.
   - `start_crowd.sh` for Mac and Unix-based systems.
2. Point a web browser at http://localhost:8095/crowd where you will see the **Crowd Setup Wizard**. Follow the instructions in the Wizard. You can also read more information about the Setup Wizard.

Next Steps

- If you are running Crowd on UNIX/Linux, consider setting Crowd to run automatically on startup and use an unprivileged system user .
- If you are running Crowd on Windows, consider setting Crowd to run automatically on startup.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Connecting Crowd to a Database

You can configure your database connection as part of the Crowd Setup Wizard. It will make things easier if you have created the database and deployed the database driver before you start.

HSQLDB database is supplied for evaluation purposes
The Standalone distribution of Crowd is shipped with an embedded HSQLDB database. You can choose this embedded database during the Crowd setup process. The embedded database is fine for evaluation purposes, but for production installations you should connect Crowd to an enterprise database. This also lets you take advantage of existing database backup and recovery procedures.

Select the page corresponding to your database, for help on setting up an external database:

- HSQLDB
- MS SQL Server
- MySQL
- Oracle
- PostgreSQL

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

HSQLDB

The Standalone distribution of Crowd is shipped with an embedded HSQLDB database. When you run the Crowd Setup Wizard, you will be asked to choose a database. If you choose the embedded database, the data files will be stored in the Crowd Home directory, as configured during installation.

Also see http://hsqldb.sourceforge.net/doc/guide/ch01.html#N101C2.

HSQLDB should not be used as a production database. It is included for evaluation purposes only.

HSQLDB periodically must update its files to represent changes made in the database. In doing so, it must delete the current `crowddb.data` file on the file system (beneath the `/database` folder in your Crowd home directory) and replace it with a new one.

If an administrator issues a shutdown on Crowd while this update is happening, data can be lost and typically all configuration data for your Crowd server will be lost.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
  - Connecting CrowdID to a Database
  - Installing Crowd and CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

MS SQL Server

Supported Versions
Crowd supports MS SQL Server 2005 and 2008 versions.

When you run the Crowd Setup Wizard, you will be asked to choose a database and provide configuration settings for that database. It will make things easier if you have created the database and deployed the database driver before you start the Setup Wizard.

Follow the instructions below to set up MS SQL Server for Crowd.

1. Configure SQL Server

1. Create a database user which Crowd will connect as (e.g. **crowduser**).

In SQL Server, the database user (**crowduser** above) should not be the database owner, but should be in the `db_owner role`.

2. Create a database for Crowd to store data in (e.g. **crowddb**).
3. Ensure that the user has permission to connect to the database, and create and populate tables

2. Copy the SQL Server Driver to your Application Server

1. Download the SQL Server JDBC driver from JTDS (recommended), or I-net software (commercial).

Microsoft have their own JDBC driver but we strongly recommend avoiding it after our JIRA customers have reported various connection errors (JRA-5760, JRA-6872), workflow problems (JRA-8443) and Chinese character problems (JRA-5054).

2. Add the SQL Server JDBC driver JAR (`jtds-[version].jar`) to the following directory:
   - For Crowd standalone distribution:
     - Crowd 2.0.2 or later: `{CROWD_INSTALL}/apache-tomcat/lib/`.
     - Crowd 2.0.1 or earlier: `{CROWD_INSTALL}/apache-tomcat/common/lib/`.
   - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
     - Tomcat 5.5.x: `common/lib/`.
     - Tomcat 6.x: `lib/`.

Next Steps

Complete the Crowd installation, then start Crowd and run the Setup Wizard as described in the Installation Guide.

Configuring Unicode Support in MS SQL Server

To configure Crowd to support Unicode in **MS SQL Server 2005 and 2008**, enter the following in the '**Hibernate Dialect**' field on the Crowd Setup Wizard's Database Configuration screen:
`com.atlassian.crowd.util.persistence.hibernate.SQLServerIntlDialect`

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
  - Connecting CrowdID to a Database
  - Installing Crowd and CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

MySQL

When you run the Crowd Setup Wizard, you will be asked to choose a database and provide configuration settings for that database. It will make things easier if you have created the database and deployed the database driver before you start the Setup Wizard.

Crowd supports MySQL **5.0.37 and later**. Follow the instructions below to set up MySQL for Crowd.

1. Configure MySQL

1. Create a database user which Crowd will connect as (e.g. **crowduser**).
2. Create a database for Crowd to store data in (e.g. **crowd**). For a UTF-8 encoded database:

```
create database crowd character set utf8;
```

3. Ensure that the user has permission to connect to the database, and create and populate tables.
4. Modify MySQL startup options in the configuration file **my.cnf** (often named **my.ini** on Windows), so the transaction level is set to `transaction-isolation = READ-COMMITTED`. (Refer to MySQL Option Files for detailed instructions on editing **my.cnf** and **my.ini**.)

```
[mysqld]
transaction-isolation = READ-COMMITTED
```

   Notes:

   - On Windows, the **my.cnf** file is often named **my.ini**. Windows can handle both file names.
   - The above configuration will prevent errors when you import directory information into Crowd. See CWD-1505.
5. Restart your MySQL server for the configuration change to take effect.

2. Copy the MySQL Driver to your Application Server

1. Download the MySQL Connector/J JDBC driver driver.
2. Add the MySQL JDBC driver jar (`mysql-connector-java-3.x.x-bin.jar`) to the following directory:
   - For Crowd standalone distribution:
     - Crowd 2.0.2 or later: `{CROWD_INSTALL}/apache-tomcat/lib/`.
     - Crowd 2.0.1 or earlier: `{CROWD_INSTALL}/apache-tomcat/common/lib/`.
   - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
     - Tomcat 5.5.x: `common/lib/`.
     - Tomcat 6.x: `lib/`.

Do not place Debug Driver on CLASSPATH
Do not place the Debug Driver (`mysql-connector-java-3.x.x-bin-g.jar`) on the `CLASSPATH` as this can cause issues. See (JRA-8674).

Next Steps

Complete the Crowd installation, then start Crowd and run the Setup Wizard as described in the Installation Guide.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
  - Connecting CrowdID to a Database
  - Installing Crowd and CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

Oracle

When you run the Crowd Setup Wizard, you will be asked to choose a database and provide configuration settings for that database. It will make things easier if you have created the database and deployed the database driver before you start the Setup Wizard.

Follow the instructions below to set up Oracle for Crowd.

1. Configure Oracle

1. Create a database user which Crowd will connect as (e.g. **crowduser**).
2. Create a database for Crowd to store data in (e.g. **crowddb**).
3. Ensure that the user has permission to connect to the database, and create and populate tables

2. Copy the Oracle driver to your application server

1. Download the Oracle JDBC driver from http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html.
2. Add the Oracle JDBC driver jar to the following directory:
   - For Crowd standalone distribution:
     - Crowd 2.0.2 or later: `{CROWD_INSTALL}/apache-tomcat/lib/`.
     - Crowd 2.0.1 or earlier: `{CROWD_INSTALL}/apache-tomcat/common/lib/`.
   - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
     - Tomcat 5.5.x: `common/lib/`.
     - Tomcat 6.x: `lib/`.

Next Steps

Complete the Crowd installation, then start Crowd and run the Setup Wizard as described in the Installation Guide.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
  - Connecting CrowdID to a Database
  - Installing Crowd and CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

PostgreSQL

When you run the Crowd Setup Wizard, you will be asked to choose a database and provide configuration settings for that database. It will

make things easier if you have created the database and deployed the database driver before you start the Setup Wizard.

Follow the instructions below to set up PostgreSQL for Crowd.

1. Configure PostgreSQL

1. Create a database user which Crowd will connect as (e.g. **crowduser**).
2. Create a database for Crowd to store data in (e.g. **crowddb**).
3. Ensure that the user has permission to connect to the database, and create and populate tables

2. Copy the PostgreSQL driver to your application server

1. Download the PostgreSQL JDBC driver from http://jdbc.postgresql.org/download.html. Get the JDBC 3 driver specific to your Postgres version, eg. `postgresql-8.x-xxx.jdbc3.jar`.
2. Add the PostgreSQL JDBC driver jar to the following directory:
    - For Crowd standalone distribution:
        - Crowd 2.0.2 or later: `{CROWD_INSTALL}/apache-tomcat/lib/`.
        - Crowd 2.0.1 or earlier: `{CROWD_INSTALL}/apache-tomcat/common/lib/`.
    - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
        - Tomcat 5.5.x: `common/lib/`.
        - Tomcat 6.x: `lib/`.

Next Steps

Complete the Crowd installation, then start Crowd and run the Setup Wizard as described in the Installation Guide.

RELATED TOPICS

- Supported Platforms
    - Setting JAVA_HOME
- Installing Crowd and CrowdID
    - Connecting Crowd to a Database
    - Connecting CrowdID to a Database
    - Installing Crowd and CrowdID WAR Distribution
    - Specifying your Crowd Home Directory
- Running the Setup Wizard
    - Troubleshooting your Configuration on Setup
- Configuring Crowd
    - Important Directories and Files
    - Changing the Port that Crowd uses
    - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
    - Specifying Startup Order of Windows Services
    - Changing the User for the Crowd Windows Service
    - Removing the Crowd Windows Service
    - Troubleshooting Crowd as a Windows Service

Connecting CrowdID to a Database

CrowdID is a free add-on that ships with Crowd versions 1.1 and later.

By default, CrowdID in the Crowd 'Standalone' distribution is shipped preconfigured with HSQL. This is fine for evaluation purposes, but for production installations, you should connect CrowdID to an enterprise database. This also lets you take advantage of existing database backup and recovery procedures.

CrowdID database connection is not yet part of Setup Wizard
This page describes the procedure for connecting **CrowdID** to an external database. You'll notice that the procedure for connecting **Crowd itself** to a database is simpler, because the Crowd database connection is configured by the Crowd Setup Wizard. The CrowdID database configuration cannot be done as part of the Setup Wizard. We hope to improve the CrowdID integration soon. In the meantime, please follow the steps below.

The following instructions will allow you to configure CrowdID to an external database:

- HSQLDB for CrowdID
- MS SQL Server for CrowdID
- MySQL for CrowdID
- Oracle for CrowdID
- PostgreSQL for CrowdID

Database Overview

CrowdID in the Crowd 'Standalone' distribution includes the Apache Tomcat application server and an in-memory HSQL database engine. This JNDI reference (`CrowdIDDS`) can be adjusted to use your custom database and driver by editing the `crowd.xml` deployment description.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
    - HSQLDB
    - MS SQL Server
    - MySQL
    - Oracle
    - PostgreSQL
  - Connecting CrowdID to a Database
    - HSQLDB for CrowdID
    - MS SQL Server for CrowdID
    - MySQL for CrowdID
    - Oracle for CrowdID
    - PostgreSQL for CrowdID
  - Installing Crowd and CrowdID WAR Distribution
    - Installing Crowd WAR Distribution
      - Configuring Crowd and CrowdID on Tomcat
      - Installing Crowd WAR on JBoss
    - Installing CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
    - The crowd.properties File
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

HSQLDB for CrowdID

The default version of CrowdID uses an embedded HSQLDB database.

Also see http://hsqldb.sourceforge.net/doc/guide/ch01.html#N101C2 .

HSQLDB periodically must update its files to represent changes made in the database. In doing so, it must delete the current `crowddb.data` file on the filesystem (beneath the `/database` folder) and replace it with a new one.

If an administrator issues a shutdown on CrowdID in this period, data can be lost, and typically all configuration data for your CrowdID server will be lost.

HSQLDB should not be used as a production database. It is included for evaluation purposes only.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
    - HSQLDB
    - MS SQL Server
    - MySQL
    - Oracle
    - PostgreSQL
  - Connecting CrowdID to a Database
    - HSQLDB for CrowdID
    - MS SQL Server for CrowdID
    - MySQL for CrowdID
    - Oracle for CrowdID
    - PostgreSQL for CrowdID
  - Installing Crowd and CrowdID WAR Distribution
    - Installing Crowd WAR Distribution
      - Configuring Crowd and CrowdID on Tomcat
      - Installing Crowd WAR on JBoss
    - Installing CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files

        - The crowd.properties File
    - Changing the Port that Crowd uses
    - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
    - Specifying Startup Order of Windows Services
    - Changing the User for the Crowd Windows Service
    - Removing the Crowd Windows Service
    - Troubleshooting Crowd as a Windows Service

MS SQL Server for CrowdID

Follow the steps below to connect CrowdID to MS SQL Server.

1. Configure SQL Server

1. Create a database user which CrowdID will connect as (e.g. **crowduser**).

   > ⚠ In SQL Server, the database user (**crowduser** above) should not be the database owner, but should be in the `db_owner role`.

2. Create a database for CrowdID to store data in (e.g. **crowdiddb**). ⚠ This must be a different database to the one used by Crowd.
3. Ensure that the user has permission to connect to the database, and create and populate tables.

2. Copy the SQL Server Driver to your Application Server

1. Download the SQL Server JDBC driver from JTDS (recommended, assumed below), or I-net software (commercial).

   > ⚠ Microsoft have their own JDBC driver but we strongly recommend avoiding it after our JIRA customers have reported various connection errors (JRA-5760, [JRA-6872|http://jira.atlassian.com/browse/JRA-6872), workflow problems (JRA-8443) and Chinese character problems (JRA-5054).

2. Add the SQL Server JDBC driver JAR (`jtds-[version].jar`) to the following directory:
    - For Crowd standalone distribution:
        - Crowd 2.0.2 or later: {`CROWD_INSTALL}/apache-tomcat/lib/`.
        - Crowd 2.0.1 or earlier: {`CROWD_INSTALL}/apache-tomcat/common/lib/`.
    - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
        - Tomcat 5.5.x: `common/lib/`.
        - Tomcat 6.x: `lib/`.

3. Configure your Application Server to Connect to SQL Server

1. Edit the `conf/Catalina/localhost/openidserver.xml` file and customise the **username**, **password**, **driverClassName** and **url** parameters for the Datasource.

```
01.<Context path="/openidserver" docBase="../../crowd-openidserver-webapp" debug="0">
02.
03.<Resource name="jdbc/CrowdIDDS" auth="Container" type="javax.sql.DataSource"
04.username="[enter db username here]"
05.password="[enter db password here]"
06.driverClassName="net.sourceforge.jtds.jdbc.Driver"
07.url="jdbc:jtds:sqlserver://localhost:1433/crowdiddb"
08.[ delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and maxActive
params here ]
09./>
10.
11.<Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>
12.
13.</Context>
```

2. Delete the **minEvictableIdleTimeMillis**, **timeBetweenEvictionRunsMillis** and **maxActive** attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure CrowdID to use MS SQL Server

1. Edit the `build.properties` file (located in the root of the Standalone distribution) and modify the **hibernate.dialect** to the following:

```
hibernate.dialect=org.hibernate.dialect.SQLServerDialect
```

2. Then run `./build.sh` or `build.bat`. This will configure CrowdID to use the MS SQL Server dialect.

If you do not wish to edit this file and run the build script, you can edit the `jdbc.properties` file (which the above script modifies) directly.

The `jdbc.properties` file is located here: `crowd-openidserver-webapp\WEB-INF\classes\jdbc.properties`. Modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdIDDS
hibernate.dialect=org.hibernate.dialect.SQLServerDialect
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next Steps

You should now have an application server configured to connect to a database, and CrowdID configured to use the correct database. Now start up CrowdID and watch the logs for any errors.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
    - HSQLDB
    - MS SQL Server
    - MySQL
    - Oracle
    - PostgreSQL
  - Connecting CrowdID to a Database
    - HSQLDB for CrowdID
    - MS SQL Server for CrowdID
    - MySQL for CrowdID
    - Oracle for CrowdID
    - PostgreSQL for CrowdID
  - Installing Crowd and CrowdID WAR Distribution
    - Installing Crowd WAR Distribution
      - Configuring Crowd and CrowdID on Tomcat
      - Installing Crowd WAR on JBoss
    - Installing CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
    - The crowd.properties File
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

MySQL for CrowdID

Follow the steps below to connect CrowdID to MySQL.

1. Configure MySQL

1. Create a database user which CrowdID will connect as (e.g. **crowduser**).
2. Create a database for CrowdID to store data in (e.g. **crowdiddb**).
   This must be a different database from the one used by Crowd.
   For a UTF-8 encoded database:

   ```
   create database crowdiddb character set utf8;
   ```

3. Ensure that the user has permission to connect to the database, and create and populate tables.

2. Copy the MySQL Driver to your Application Server

1. Download the latest MySQL Connector/J JDBC driver.
2. Add the MySQL JDBC driver jar (`mysql-connector-java-3.x.x-bin.jar`) to the following directory:
   - For Crowd standalone distribution:
     - Crowd 2.0.2 or later: {`CROWD_INSTALL`}`/apache-tomcat/lib/`.
     - Crowd 2.0.1 or earlier: {`CROWD_INSTALL`}`/apache-tomcat/common/lib/`.

- For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
  - Tomcat 5.5.x: `common/lib/`.
  - Tomcat 6.x: `lib/`.

Do not place the Debug Driver (**mysql-connector-java-3.x.x-bin-g.jar**) on the `CLASSPATH` as this can cause issues. (JRA-8674).

3. Configure your Application Server to Connect to MySQL

1. Edit the file `apache-tomcat-X.X.XX/conf/Catalina/localhost/openidserver.xml` and customise the **username**, **password**, **driverClassName** and **url** parameters for the Datasource.

```
<Context path="/openidserver" docBase="../../crowd-openidserver-webapp" debug="0">

<Resource name="jdbc/CrowdIDDS" auth="Container" type="javax.sql.DataSource"
username="[enter db username here]"
password="[enter db password here]"
driverClassName="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost/crowddiddb?autoReconnect=true&amp;useUnicode=true&amp;characterEnco
delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and maxActive params
here ]
/>

<Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>

</Context>
```

   The URL above assumes a UTF-8 database — i.e. created with `create database crowddiddb character set utf8;`.

MySQL closes idle connections after 8 hours, so the `autoReconnect=true` is necessary to tell the driver to reconnect.

2. Delete the **minEvictableIdleTimeMillis**, **timeBetweenEvictionRunsMillis** and **maxActive** attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure CrowdID to use MySQL

1. Edit the `build.properties` file (located in the root of the Standalone distribution) and modify the **hibernate.dialect** to the following. **Please choose only one of the 3 available options depending on how you have configured your database server**.

```
*For MySQL set:*
hibernate.dialect=org.hibernate.dialect.MySQLDialect
*For MySQL with InnoDB set:*
hibernate.dialect=org.hibernate.dialect.MySQLInnoDBDialect
*For MySQL with MyISAM set:*
hibernate.dialect=org.hibernate.dialect.MySQLMyISAMDialect
```

2. Then run `./build.sh` or `build.bat`. This will configure CrowdID to use the MySQL dialect.

If you do not wish to edit this file and run the build script, you can edit the **jdbc.properties** (which the above script modifies) directly. The **jdbc.properties** file is located here: `crowd-openidserver-webapp\WEB-INF\classes\jdbc.properties`. Modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdIDDS
hibernate.dialect=org.hibernate.dialect.MySQLDialect
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next steps

You should now have an application server configured to connect to a database, and CrowdID configured to use the correct database. Now start up CrowdID and watch the logs for any errors.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
    - HSQLDB
    - MS SQL Server
    - MySQL
    - Oracle
    - PostgreSQL
  - Connecting CrowdID to a Database
    - HSQLDB for CrowdID
    - MS SQL Server for CrowdID
    - MySQL for CrowdID
    - Oracle for CrowdID
    - PostgreSQL for CrowdID
  - Installing Crowd and CrowdID WAR Distribution
    - Installing Crowd WAR Distribution
      - Configuring Crowd and CrowdID on Tomcat
      - Installing Crowd WAR on JBoss
    - Installing CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
    - The crowd.properties File
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

Oracle for CrowdID

Follow the steps below to connect CrowdID to Oracle.

1. Configure Oracle

1. Create a database user which CrowdID will connect as (e.g. **crowduser**).
2. Create a database for CrowdID to store data in (e.g. **crowdiddb**). This must be a different database to the one used by Crowd.
3. Ensure that the user has permission to connect to the database, and create and populate tables.

2. Copy the Oracle Driver to your Application Server

1. Download the Oracle JDBC driver from http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html.
2. Add the Oracle JDBC driver jar to the following directory:
   - For Crowd standalone distribution:
     - Crowd 2.0.2 or later: {`CROWD_INSTALL`}`/apache-tomcat/lib/`.
     - Crowd 2.0.1 or earlier: {`CROWD_INSTALL`}`/apache-tomcat/common/lib/`.
   - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
     - Tomcat 5.5.x: `common/lib/`.
     - Tomcat 6.x: `lib/`.

3. Configure your Application Server to Connect to Oracle

1. Edit the file `apache-tomcat-X.X.XX/conf/Catalina/localhost/openidserver.xml` and customise the **username**, **password**, **driverClassName** and **url** parameters for the Datasource.

```
01.<Context path="/openidserver" docBase="../../crowd-openidserver-webapp" debug="0">
02.
03.<Resource name="jdbc/CrowdIDDS" auth="Container" type="javax.sql.DataSource"
04.username="[enter db username here]"
05.password="[enter db password here]"
06.driverClassName="oracle.jdbc.driver.OracleDriver"
07.url="jdbc:oracle:thin:@localhost:1521:crowdiddb"
08.[ delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and maxActive
params here ]
09./>
10.
11.<Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>
12.
13.</Context>
```

2. Delete the **minEvictableIdleTimeMillis**, **timeBetweenEvictionRunsMillis** and **maxActive** attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure CrowdID to use Oracle

1. Edit the `build.properties file` (located in the root of the standalone release) and modify the **hibernate.dialect** to the following

```
hibernate.dialect=org.hibernate.dialect.OracleDialect
```

2. Then run `./build.sh` or `build.bat`. This will configure CrowdID to use the Oracle dialect. ⚠ There is a problem with `build.bat` in Crowd version 1.2.0. To fix the problem, please apply the patch described in CWD-638.

If you do not wish to edit this file and run the build script, you can edit the **jdbc.properties** (which the above script modifies) directly. The **jdbc.properties** file is located here: `crowd-openidserver-webapp\WEB-INF\classes\jdbc.properties`. Modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdIDDS
hibernate.dialect=org.hibernate.dialect.Oracle
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next Steps

You should now have an application server configured to connect to a database, and CrowdID configured to use the correct database. Now start up CrowdID and watch the logs for any errors.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database
    - HSQLDB
    - MS SQL Server
    - MySQL
    - Oracle
    - PostgreSQL
  - Connecting CrowdID to a Database
    - HSQLDB for CrowdID
    - MS SQL Server for CrowdID
    - MySQL for CrowdID
    - Oracle for CrowdID
    - PostgreSQL for CrowdID
  - Installing Crowd and CrowdID WAR Distribution
    - Installing Crowd WAR Distribution
      - Configuring Crowd and CrowdID on Tomcat
      - Installing Crowd WAR on JBoss
    - Installing CrowdID WAR Distribution
  - Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
    - The crowd.properties File
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

PostgreSQL for CrowdID

Follow the steps below to connect CrowdID to PostgreSQL.

1. Configure PostgreSQL

1. Create a database user which CrowdID will connect as (e.g. **crowduser**).
2. Create a database for CrowdID to store data in (e.g. **crowdiddb**). ⚠ This must be a different database to the one used by Crowd.
3. Ensure that the user has permission to connect to the database, and create and populate tables.

2. Copy the PostgreSQL Driver to your Application Server

1. Download the PostgreSQL JDBC driver from http://jdbc.postgresql.org/download.html. Get the JDBC 3 driver specific to your Postgres version, eg. + postgresql-8.x-xxx.jdbc3.jar.
2. Add the PostgreSQL JDBC driver JAR to the following directory:
    - For Crowd standalone distribution:
        - Crowd 2.0.2 or later: {CROWD_INSTALL}/apache-tomcat/lib/.
        - Crowd 2.0.1 or earlier: {CROWD_INSTALL}/apache-tomcat/common/lib/.
    - For Crowd WAR distribution, copy the driver JAR to your application server. For example, on Tomcat:
        - Tomcat 5.5.x: common/lib/.
        - Tomcat 6.x: lib/.

3. Configure your Application Server to Connect to PostgreSQL

1. Edit the file apache-tomcat-X.X.XX/conf/Catalina/localhost/openidserver.xml and customise the **username**, **password**, **driverClassName** and **url** parameters for the Datasource.

```
01.<Context path="/openidserver" docBase="../../crowd-openidserver-webapp" debug="0">
02.
03.<Resource name="jdbc/CrowdIDDS" auth="Container" type="javax.sql.DataSource"
04.username="[enter db username here]"
05.password="[enter db password here]"
06.driverClassName="org.postgresql.Driver"
07.url="jdbc:postgresql://host:port/crowdiddb" [ see also
http://jdbc.postgresql.org/doc.html) ]"
08.[ delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and maxActive
params here ]
09./>
10.
11.<Manager className="org.apache.catalina.session.PersistentManager" saveOnRestart="false"/>
12.
13.</Context>
```

2. Delete the **minEvictableIdleTimeMillis**, **timeBetweenEvictionRunsMillis** and **maxActive** attributes (which are only needed for HSQL, and degrade performance otherwise).

4. Configure CrowdID to use PostgreSQL

1. Edit the build.properties file located in the root of the standalone release and modify the **hibernate.dialect** to the following

```
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
```

2. Then run ./build.sh or build.bat. This will configure crowd to use the PostgreSQL dialect. There is a problem with build.bat in Crowd version 1.2.0. To fix the problem, please apply the patch described in CWD-638.

If you do not wish to edit this file and run the build script, you can edit the **jdbc.properties** (which the above script modifies) directly. The **jdbc.properties** file is located here: crowd-openidserver-webapp\WEB-INF\classes\jdbc.properties. Modify the file to the following:

```
# - Crowd Configuration Options

hibernate.connection.datasource=java\:comp/env/jdbc/CrowdIDDS
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

...
```

Next Steps

You should now have an application server configured to connect to a database, and CrowdID configured to use the correct database. Now start up CrowdID and watch the logs for any errors.

RELATED TOPICS

- Supported Platforms
    - Setting JAVA_HOME
- Installing Crowd and CrowdID
    - Connecting Crowd to a Database
        - HSQLDB
        - MS SQL Server
        - MySQL
        - Oracle
        - PostgreSQL
    - Connecting CrowdID to a Database
        - HSQLDB for CrowdID
        - MS SQL Server for CrowdID
        - MySQL for CrowdID

- Oracle for CrowdID
- PostgreSQL for CrowdID
- Installing Crowd and CrowdID WAR Distribution
  - Installing Crowd WAR Distribution
    - Configuring Crowd and CrowdID on Tomcat
    - Installing Crowd WAR on JBoss
  - Installing CrowdID WAR Distribution
- Specifying your Crowd Home Directory
- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
    - The crowd.properties File
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

Installing Crowd and CrowdID WAR Distribution

> ⚠ The **Crowd and CrowdID WAR distributions** are intended for deployment onto an existing J2EE application server. It is assumed that you already know how to deploy a web application onto your chosen application server. If not, please contact your system administrator to assist you, or consider installing the Crowd Standalone distribution instead.

The standard Crowd installation guide tells you how to install the Standalone distribution of Crowd, which includes Apache Tomcat. Instead, you may wish to deploy Crowd or CrowdID onto your own existing application server. For this purpose, we provide WAR (Webapp ARchive) distributions of the Crowd and CrowdID server applications.

Crowd supports all the application servers listed in System Requirements.

The procedures for connecting **Crowd** and **CrowdID** are slightly different. The **Crowd** setup process provides the option of JDBC or JNDI datasource connections via the Crowd Setup Wizard. **CrowdID** requires a JNDI datasource configuration. Detailed instructions are on the following pages:

- Installing Crowd WAR Distribution
- Installing CrowdID WAR Distribution

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Installing Crowd WAR Distribution

> ⚠ The **Crowd and CrowdID WAR distributions** are intended for deployment onto an existing J2EE application server. It is assumed that you already know how to deploy a web application onto your chosen application server. If not, please contact your system administrator to assist you, or consider installing the Crowd Standalone distribution instead.

The standard Crowd installation guide tells you how to install the Standalone distribution of Crowd, which includes Apache Tomcat. Instead, you may wish to deploy Crowd or CrowdID onto your own existing application server. For this purpose, we provide WAR (Webapp ARchive) distributions of the Crowd and CrowdID server applications.

Below is a generic overview of the steps required to install the **Crowd** WAR distribution. You will need to perform specific configuration steps depending upon your application server. As well as the generic instructions below, we also provide specific instructions on the following pages:

- Configuring Crowd and CrowdID on Tomcat
- Installing Crowd WAR on JBoss

Step 1. Check the System Requirements

⚠ Please check that your database and server are supported and make sure that all dependencies are installed as described below, otherwise Crowd will not run properly.

Supported Platforms

Key: ✓ = Supported. ✗ = Not Supported

Java Version	
JDK (1)	✓ 1.6, 1.5 ✗ 1.4
Operating Systems	
Microsoft Windows (2)	✓
Linux / Solaris (2)	✓
Apple Mac OS X (2)	✓
Application Servers	
Apache Tomcat	✓ 6.0.x (Crowd ships with Apache Tomcat 6.0.20) ✓ 5.5.x (Tested on 5.5.26)
Databases	
MySQL (3)	✓ 5.0.37 and later
Oracle	✓ 10g (Tested on 10.2.0.1.)
PostgreSQL	✓ 8.x, 7.x
Microsoft SQL Server	✓ 2008, 2005
HSQLDB (4)	✓ (For evaluation only.)
Web Browsers	
Microsoft Internet Explorer (Windows)	✓ 8, 7 ✗ 6
Mozilla Firefox (all platforms)	✓ 3.x ✗ 2.x
Safari	✓ 4.x
Opera	✗

Notes:
1. JDK:

- It is not enough to have the JRE only. Please ensure that you have the full JDK. You can download the Java SE Development Kit (JDK) from the Sun website.
- Once the JDK is installed, you will need to set the **JAVA_HOME** environment variable, pointing to the root directory of the JDK. Some JDK installers set this automatically (check by typing 'echo %JAVA_HOME%' in a DOS prompt, or 'echo $JAVA_HOME' in a shell). If it is not set, please see Setting JAVA_HOME.

2. Operating systems: Crowd is a pure Java application and should run on any platform provided the Java runtime platform requirements are satisfied.

3. MySQL: Please ensure that you set transaction isolation to 'read-committed' instead of the default 'repeatable-read', as described in the database configuration guide.

4. HSQLDB: Crowd ships with a built-in HSQL database, which is fine for evaluation purposes but is somewhat susceptible to data loss during system crashes. For production environments we recommend that you configure Crowd to use an external database.
Dependencies

Ensure that the following JAR files are deployed in the shared `lib` folder on the application server:

- JTA (Java Transaction API)

The JTA specifies standard Java interfaces between a transaction manager and the parties involved in a distributed transaction system: the resource manager, the application server and the transactional applications. Refer to the Sun documentation for more information.

- JavaMail classes
- Java Beans Activation Framework (for those using Sun JDK 1.5.x only, this is included in JDK 1.6)

All of these JAR files are available in the Crowd Standalone Distribution zip file, available on the Crowd download centre. The files are: `activation-1.1.jar`, `jta-1.0.1B.jar` and `mail-1.4.jar`. You will find them in {`CROWD_INSTALL`}`\apache-tomcat\lib` (for Crowd **2.0.2 or later**) or in {`CROWD_INSTALL`}`\apache-tomcat\common\lib` (for Crowd **2.0.1 or earlier**).

Step 2. Install Crowd WAR

Below is a summary of the Crowd WAR installation steps:

1. Download the Crowd WAR distribution from the Crowd download centre.

   You will find the WAR archives for the Crowd and the CrowdID applications by clicking the '**Show all**' link. You will need to deploy each application separately. For the rest of these instructions, we assume you are deploying Crowd WAR.
2. Please check your unzip program before extracting the downloaded archive, as some unzip programs can cause errors — see the note on the Crowd installation front page.
3. Unzip the download archive into a directory of your choice. We'll call it `CROWD` in the rest of these instructions.
4. Specify your Crowd Home directory by editing the configuration file at `server/default/deploy/crowd.war/WEB-INF/classes/crowd-init.properties`.

   The **Crowd Home** directory is where Crowd will store its configuration information. If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will also store its database in this directory. To specify the directory:

   - Open the `crowd-init.properties` file.
   - Choose the appropriate line in the file, depending upon your operating system (see below).
   - Remove the # at the beginning of the line.
   - Enter the name of the directory you want Crowd to use as its Home directory. For example,
     - On Windows:

       ```
       crowd.home=c:/data/crowd-home
       ```

       Note: On Windows, make sure you use forward slashes as shown above, not backward slashes.

     - On Mac and UNIX-based systems:

       ```
       crowd.home=/var/crowd-home
       ```

   - Save the `crowd-init.properties` file.
5. Create a database in your chosen database server.
6. Copy the JDBC driver into your application server's classpath.
7. Depending upon your application server, you may need to zip up the WAR file again before deploying it. Place the `CROWD` directory or the WAR file into your application server's deployment directory. Please consult the server-specific documentation on how to do this. A few Atlassian best practice guides are listed here:
   - Configuring Crowd and CrowdID on Tomcat
   - Installing Crowd WAR on JBoss
8. Restart your application server.
9. Point a web browser at the IP address and port that your application server is running on (typically `http://localhost:8080`). The Crowd Setup Wizard will start.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Configuring Crowd and CrowdID on Tomcat

The standard Crowd installation guide tells you how to install the Standalone distribution of Crowd, which includes Apache Tomcat. You may wish to deploy Crowd on your own existing application server instead. For this purpose, we provide WAR (Webapp ARchive) distributions of the Crowd and CrowdID server applications.

This page contains specific configuration steps for deploying Crowd or CrowdID on an existing Apache Tomcat server.

Please refer also to the generic WAR setup guide.

On this page:

- Configuring a Crowd 'context' in Tomcat
- Configuring a CrowdID 'context' in Tomcat
- Modifying Tomcat's server.xml to allow for Unicode Character Set

Configuring a Crowd 'context' in Tomcat

1. Create a file called `crowd.xml` that contains the following context:

```
1.<Context path="/crowd" docBase="/path/to/atlassian-crowd-war-directory" reloadable="false"
/>
```

2. Place the file in your Tomcat's `conf/Catalina/localhost/` directory.

3. Modify the '`/path/to/atlassian-crowd-war-directory`' in the above file to reflect the actual path to your Crowd WAR distribution. To avoid problems with your deployment, this should **NOT** be Tomcat's `webapps` directory.
If you are installing Crowd or CrowdID on Windows, make sure that the paths you specify for the location of the WAR file and database are full paths including drive letters.

Configuring a CrowdID 'context' in Tomcat

If you are deploying CrowdID, you will need to specify a JNDI datasource.

1. Create a file called `openidserver.xml` that resembles the following example. Our example is for a MySQL database:

```
1.<Context path="/openidserver" docBase="/path/to/atlassian-crowd-openid-war-directory"
reloadable="false">
2.<Resource name="jdbc/CrowdIDDS" auth="Container" type="javax.sql.DataSource"
3.username="crowdiduser"
4.password="mypassword"
5.driverClassName="com.mysql.jdbc.Driver"
6.url="jdbc:mysql://localhost/crowdopenidserverdb?autoReconnect=true"/>
7.</Context>
```

2. Place the file in your Tomcat's `conf/Catalina/localhost/` directory.

3. Modify the '`/path/to/atlassian-crowd-openid-war-directory`' in the above file to reflect the actual path to your CrowdID WAR distribution. To avoid problems with your deployment, this should **NOT** be Tomcat's `webapps` directory.
If you are installing Crowd or CrowdID on Windows, make sure that the paths you specify for the location of the WAR file and database are full paths including drive letters.

4. Modify the content of the above file appropriately for your database (e.g. Oracle, PostgreSQL, etc). In particular, please remember to update the `driveClassName`.

5. Copy the JDBC driver JAR to your Tomcat's directory:
   - For Tomcat 5.5.x: `common/lib/`.
   - For Tomcat 6.x: `lib/`.

Modifying Tomcat's server.xml to allow for Unicode Character Set

If your user directory contains usernames or group names with Unicode characters, you need to modify your Tomcat distribution's `conf/server.xml` file. For example, you need to do this if your user directory allows for internationalised characters in usernames.

The additional setting described below will ensure that the HTTP query string is encoded in the same way as the body, instead of using the URI character encoding.

1. In your Tomcat distribution's `conf/server.xml` file, find the connector definition for your HTTP protocol. The connector definition looks very much like this:

```
1.<Connector port="8080" maxHttpHeaderSize="8192"
2.maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
3.enableLookups="false" redirectPort="8443" acceptCount="100"
4.connectionTimeout="20000" disableUploadTimeout="true"/>
```

2. Modify the block to include the `useBodyEncodingForURI` property with value set to 'true':

```
<Connector port="8080" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" redirectPort="8443" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true" useBodyEncodingForURI="true"/>
```

This setting affects all web applications
Because you must define this property at the connector level, this setting will affect all web applications you have deployed under the connector. This should not adversely affect the other web applications, but please be aware of this fact. Crowd and CrowdID will run fine without this property set, but you will run into issues if a username or group contains internationalised characters. It is best to set this property to 'true'.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Installing Crowd WAR on JBoss

The standard Crowd installation guide tells you how to install the Standalone distribution of Crowd, which includes Apache Tomcat. You may wish to deploy Crowd on your own existing application server instead. For this purpose, we provide WAR (Webapp ARchive) distributions of the Crowd and CrowdID server applications.

This page shows one example - use it as a basis for other installations

- This page tells you how to deploy Crowd onto a JBoss Application Server. For other application servers, refer to the generic WAR setup guide.
- On this page, we have used PostgreSQL as an example of a database connected via a JNDI datasource. Crowd supports all the databases listed under supported platforms. Refer to Connecting Crowd to a Database for instructions on connecting Crowd to your database.

Step 1. Check the System Requirements

Please check that your database and server are supported and make sure that all dependencies are installed as described below, otherwise Crowd will not run properly.

Supported Platforms

Key: = Supported. = Not Supported

Java Version	
JDK (1)	1.6, 1.5 1.4
Operating Systems	
Microsoft Windows (2)	
Linux / Solaris (2)	
Apple Mac OS X (2)	
Application Servers	
Apache Tomcat	6.0.x (Crowd ships with Apache Tomcat 6.0.20) 5.5.x (Tested on 5.5.26)
Databases	
MySQL (3)	5.0.37 and later
Oracle	10g (Tested on 10.2.0.1.)
PostgreSQL	8.x, 7.x

Microsoft SQL Server	✓ 2008, 2005
HSQLDB [4]	✓ (For evaluation only.)
Web Browsers	
Microsoft Internet Explorer (Windows)	✓ 8, 7 ✗ 6
Mozilla Firefox (all platforms)	✓ 3.x ✗ 2.x
Safari	✓ 4.x
Opera	✗

Notes:
1. JDK:

- It is not enough to have the JRE only. Please ensure that you have the full JDK. You can download the Java SE Development Kit (JDK) from the Sun website.
- Once the JDK is installed, you will need to set the **JAVA_HOME** environment variable, pointing to the root directory of the JDK. Some JDK installers set this automatically (check by typing 'echo %JAVA_HOME%' in a DOS prompt, or 'echo $JAVA_HOME' in a shell). If it is not set, please see Setting JAVA_HOME.

2. Operating systems: Crowd is a pure Java application and should run on any platform provided the Java runtime platform requirements are satisfied.

3. MySQL: Please ensure that you set transaction isolation to 'read-committed' instead of the default 'repeatable-read', as described in the database configuration guide.

4. HSQLDB: Crowd ships with a built-in HSQL database, which is fine for evaluation purposes but is somewhat susceptible to data loss during system crashes. For production environments we recommend that you configure Crowd to use an external database.
Dependencies

Ensure that the following JAR files are deployed in the shared `lib` folder on the application server:

- JTA (Java Transaction API)

  > The JTA specifies standard Java interfaces between a transaction manager and the parties involved in a distributed transaction system: the resource manager, the application server and the transactional applications. Refer to the Sun documentation for more information.

- JavaMail classes
- Java Beans Activation Framework (for those using Sun JDK 1.5.x only, this is included in JDK 1.6)

  All of these JAR files are available in the Crowd Standalone Distribution zip file, available on the Crowd download centre. The files are: `activation-1.1.jar`, `jta-1.0.1B.jar` and `mail-1.4.jar`. You will find them in {`CROWD_INSTALL`}`\apache-tomcat\lib` (for Crowd **2.0.2 or later**) or in {`CROWD_INSTALL`}`\apache-tomcat\common\lib` (for Crowd **2.0.1 or earlier**).

Step 2. Install Crowd WAR

Follow the steps below to install Crowd on JBoss 4.2.2 GA using a PostgreSQL database:

1. Download the WAR distribution from the Crowd download centre.
   You will find the WAR archives for the Crowd and the CrowdID applications. You will need to deploy Crowd and CrowdID WARs separately. For the rest of these instructions, we assume you are deploying Crowd WAR.

2. Please check your unzip program before extracting the downloaded archive, as some unzip programs can cause errors – see the note on the Crowd installation front page.

3. Unzip the download archive a directory named `crowd.war` inside `JBOSS_INSTALL/server/default/deploy`. In the rest of these instruction we will call this directory `server/default/deploy/crowd.war`.

4. Specify your Crowd Home directory by editing the configuration file at `server/default/deploy/crowd.war/WEB-INF/classes/crowd-init.properties`.

   The **Crowd Home** directory is where Crowd will store its configuration information. If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will also store its database in this directory. To specify the directory:

- Open the `crowd-init.properties` file.
- Choose the appropriate line in the file, depending upon your operating system (see below).
- Remove the # at the beginning of the line.
- Enter the name of the directory you want Crowd to use as its Home directory. For example,
  - On Windows:

    ```
    crowd.home=c:/data/crowd-home
    ```

    Note: On Windows, make sure you use forward slashes as shown above, not backward slashes.

  - On Mac and UNIX-based systems:

    ```
    crowd.home=/var/crowd-home
    ```

- Save the `crowd-init.properties` file.

5. Add a file `server/default/deploy/crowd.war/WEB-INF/jboss-web.xml` with the following content:

```
1.<jboss-web>
2.     <resource-ref>
3.          <res-ref-name>jdbc/CrowdDS</res-ref-name>
4.          <res-type>javax.sql.DataSource</res-type>
5.          <jndi-name>java:CrowdDS</jndi-name>
6.     </resource-ref>
7.</jboss-web>
```

6. Create a database `crowd_db` in your database server. In our example, we assume you are using PostgreSQL.
7. Add a datasource definition file `server/default/deploy/postgres-ds.xml` that resembles the following example. Our example is for a PostgreSQL database:

```
01.<datasources>
02.     <local-tx-datasource>
03.          <jndi-name>CrowdDS</jndi-name>
04.          <connection-url>jdbc:postgresql://localhost:5432/crowd_db</connection-url>
05.          <driver-class>org.postgresql.Driver</driver-class>
06.          <user-name>postgres</user-name>
07.          <password>postgres</password>
08.     </local-tx-datasource>
09.</datasources>
```

8. Modify your `{CROWD_HOME}/crowd.properties` file to point to the port of the JBoss server. This file is located at the root of your Crowd Home directory (specified above). Port `8080` is the default port number, and is shown in the example below:

```
1.crowd.server.url=http://localhost:8080/crowd/services/
2.application.login.url=http://localhost:8080/crowd/console/
```

9. Start JBoss with `run.sh` (Unix-based systems) or `run.bat` (Windows).
10. Point a web browser at http://localhost:8080/ where you will see the Crowd Setup Wizard.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Installing CrowdID WAR Distribution

> ⚠ The **Crowd and CrowdID WAR distributions** are intended for deployment onto an existing J2EE application server. It is assumed that you already know how to deploy a web application onto your chosen application server. If not, please contact your system administrator to assist you, or consider installing the Crowd Standalone distribution instead.

The standard Crowd installation guide tells you how to install the Standalone distribution of Crowd, which includes Apache Tomcat. Instead, you may wish to deploy Crowd or CrowdID onto your own existing application server. For this purpose, we provide WAR (Webapp ARchive) distributions of the Crowd and CrowdID server applications.

Below is a generic overview of the steps required to install the **CrowdID** WAR distribution. You will need to perform specific configuration steps, depending upon your application server. As well as the generic instructions below, we also provide server-specific instructions on the following pages:

- Configuring Crowd and CrowdID on Tomcat

Step 1. Check the System Requirements

⚠ Please check that your database and server are supported and all dependencies are installed as described below, otherwise Crowd will not run properly.

Supported Platforms

Key: ✓ = Supported. ✗ = Not Supported

Java Version	
JDK (1)	✓ 1.6, 1.5 ✗ 1.4
Operating Systems	
Microsoft Windows (2)	✓
Linux / Solaris (2)	✓
Apple Mac OS X (2)	✓
Application Servers	
Apache Tomcat	✓ 6.0.x (Crowd ships with Apache Tomcat 6.0.20) ✓ 5.5.x (Tested on 5.5.26)
Databases	
MySQL (3)	✓ 5.0.37 and later
Oracle	✓ 10g (Tested on 10.2.0.1.)
PostgreSQL	✓ 8.x, 7.x
Microsoft SQL Server	✓ 2008, 2005
HSQLDB (4)	✓ (For evaluation only.)
Web Browsers	
Microsoft Internet Explorer (Windows)	✓ 8, 7 ✗ 6
Mozilla Firefox (all platforms)	✓ 3.x ✗ 2.x
Safari	✓ 4.x
Opera	✗

Notes:
1. JDK:

- It is not enough to have the JRE only. Please ensure that you have the full JDK. You can download the Java SE Development Kit (JDK) from the Sun website.
- Once the JDK is installed, you will need to set the **JAVA_HOME** environment variable, pointing to the root directory of the JDK. Some JDK installers set this automatically (check by typing 'echo %JAVA_HOME%' in a DOS prompt, or 'echo $JAVA_HOME' in a shell). If it is not set, please see Setting JAVA_HOME.

2. Operating systems: Crowd is a pure Java application and should run on any platform provided the Java runtime platform requirements are satisfied.

3. MySQL: Please ensure that you set transaction isolation to 'read-committed' instead of the default 'repeatable-read', as described in the database configuration guide.

4. HSQLDB: Crowd ships with a built-in HSQL database, which is fine for evaluation purposes but is somewhat susceptible to data loss during system crashes. For production environments we recommend that you configure Crowd to use an external database.
Dependencies

Ensure that the following JAR files are deployed in the shared `lib` folder on the application server:

- JTA (Java Transaction API)

  > The JTA specifies standard Java interfaces between a transaction manager and the parties involved in a distributed transaction system: the resource manager, the application server and the transactional applications. Refer to the Sun documentation for more information.

- JavaMail classes
- Java Beans Activation Framework (for those using Sun JDK 1.5.x only, this is included in JDK 1.6)

  All of these JAR files are available in the Crowd Standalone Distribution zip file, available on the Crowd download centre. The files are: `activation-1.1.jar`, `jta-1.0.1B.jar` and `mail-1.4.jar`. You will find them in {`CROWD_INSTALL`}`\apache-tomcat\lib` (for Crowd **2.0.2 or later**) or in {`CROWD_INSTALL`}`\apache-tomcat\common\lib` (for Crowd **2.0.1 or earlier**).

Step 2. Install CrowdID WAR

Below is a summary of the CrowdID WAR installation steps:

1. Download the CrowdID WAR distribution from the Crowd download centre.
   You will find the WAR archives for the Crowd and the CrowdID applications. You will need to deploy each application separately. For the rest of these instructions, we assume you are deploying CrowdID WAR.
2. Please check your unzip program before extracting the downloaded archive – see the note on the Crowd installation front page.
3. Unzip the download archive into a directory of your choice. We'll call it `CROWDID` in the rest of these instructions.
4. Create a database in your chosen database server and add the required datasource definition file to your application server.
5. Modify file `CROWDID/WEB-INF/classes/jdbc.properties` to use your chosen Hibernate database dialect, as explained in the previous step.
6. Modify file `CROWDID/WEB-INF/classes/crowd.properties` to point to the port of your application server. `8080` is the default, and is shown in the example below:

   ```
   1.crowd.server.url=http://localhost:8080/crowd/services/
   2.application.login.url=http://localhost:8080/crowd/console/
   ```

7. Depending upon your application server, you may need to zip up the WAR file again before deploying it. Place the `CROWDID` directory or the WAR file into your application server's deployment directory. Please consult the server-specific documentation on how to do this.
8. Restart your application server.
9. Point a web browser at the IP address and port that your application server is running on (typically `http://localhost:8080`). The Crowd Setup Wizard will start.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Specifying your Crowd Home Directory

The **Crowd Home** directory is where Crowd will store its configuration information. If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will also store its database in this directory. To specify the directory:

- Open the `crowd-init.properties` file.
- Choose the appropriate line in the file, depending upon your operating system (see below).
- Remove the # at the beginning of the line.
- Enter the name of the directory you want Crowd to use as its Home directory. For example,

- On Windows:

```
1.crowd.home=c:/data/crowd-home
```

Note: On Windows, make sure you use forward slashes as shown above, not backward slashes.

- On Mac and UNIX-based systems:

```
1.crowd.home=/var/crowd-home
```

- Save the `crowd-init.properties` file.

Advanced Usage

It is also possible to define the `crowd.home` property as a Java system or Servlet Context parameter.

Java System Parameter

Use the following format for your Java parameter:

```
1.-Dcrowd.home=/var/crowd-home
```

Where should you put this value?

You could add it to the `setenv.sh` or `setenv.bat` file supplied with the standalone release of Crowd.

Servlet Context Parameter

The following configuration XML can be added to the `crowd-standalone-install/apache-tomcat/conf/Catalina/localhost/crowd.xml` context file to set the `crowd.home` property:

```
1.<Parameter name="crowd.home" value="/var/crowd-home" override="false"/>
```

Running the Setup Wizard

Before running the Setup Wizard described below, please follow the instructions on installing Crowd.

When you access the Crowd Administration Console for the first time, you will see the **Crowd Setup Wizard**. This is a series of screens which will prompt you to configure your database connection and to supply some default values (which you can change later if necessary).

On this page:

- Step 1. Starting the Setup Wizard
- Step 2. Licensing
- Step 3. Installation Type
- Step 4. Database Configuration
- Step 5. (*Optional*) Import Existing Crowd Data
- Step 6. Options
- Step 7. Mail Server
- Step 8. Default Directory
- Step 9. Default Administrator
- Step 10. Integrated Applications
- Step 11. Setup Complete

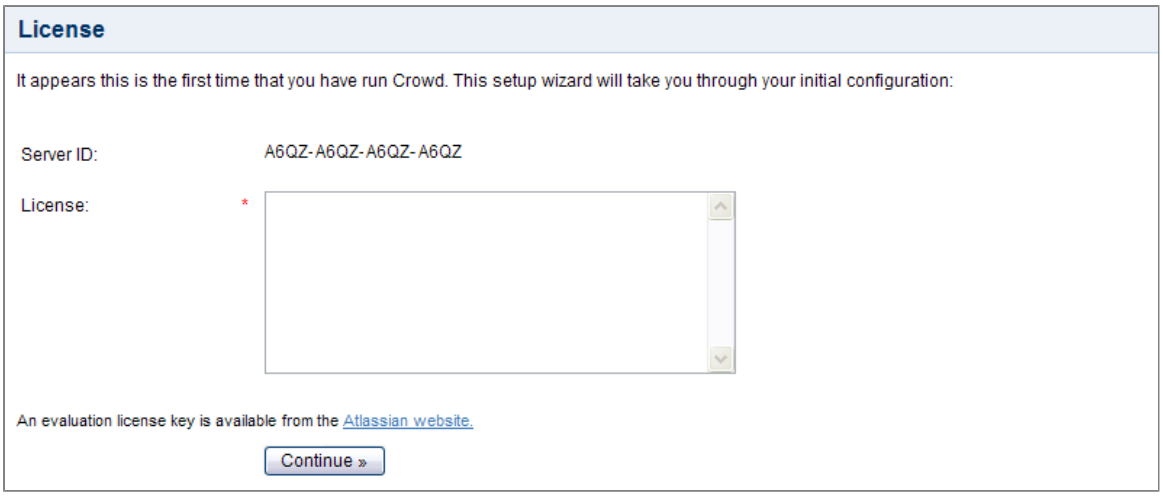
Do you need to restart the Setup Wizard from the beginning?
Read this hint in the Crowd Knowledge Base.

Step 1. Starting the Setup Wizard

Go to the following URL in your web browser: http://localhost:8095/crowd or http://localhost:8095/crowd/console.

- If there are no errors, you should see the '**License**' screen described below.
- If there is an error in your configuration, you will see the '**Crowd Checklist**' screen. Read more about troubleshooting your installation.

Step 2. Licensing

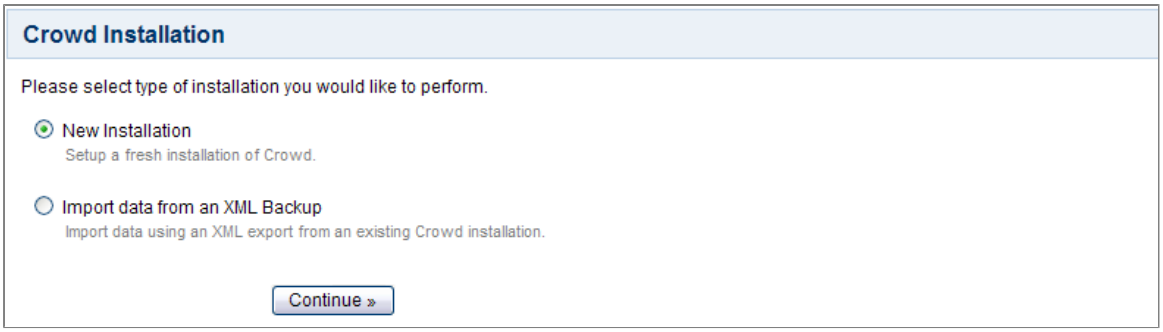


Crowd licenses are based on the number of end-users who will log in to the applications that are integrated with Crowd.

You can obtain an evaluation license from the Atlassian website. When you obtain an evaluation license — or purchase, renew or upgrade your license — you will receive a license key via email or on the Atlassian website.

Type or paste your license key into the '**License**' field, shown on the screenshot above.

Step 3. Installation Type



In this step, you will choose whether to set up a new Crowd database or restore an existing database. Choose an option as follows:

- '**New Installation**' — Set up a new Crowd database.
  Hint: Choose this option if you are evaluating Crowd.
- '**Import data from an XML Backup**' — Import your Crowd data from an XML backup file, which has been exported from your existing Crowd installation.

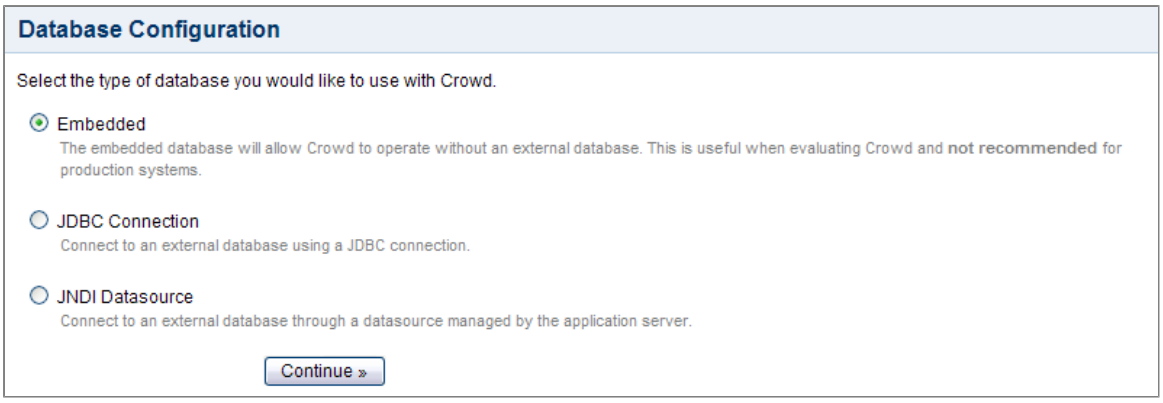
Step 4. Database Configuration

The '**Database Configuration**' screen allows you to choose the type of database connection, as described below.

> **If in any doubt, choose the default 'Embedded' option for evaluation purposes.**

When you click 'Continue' after choosing your database options, there may be a short wait while Crowd writes the information to the database tables. Please be patient.

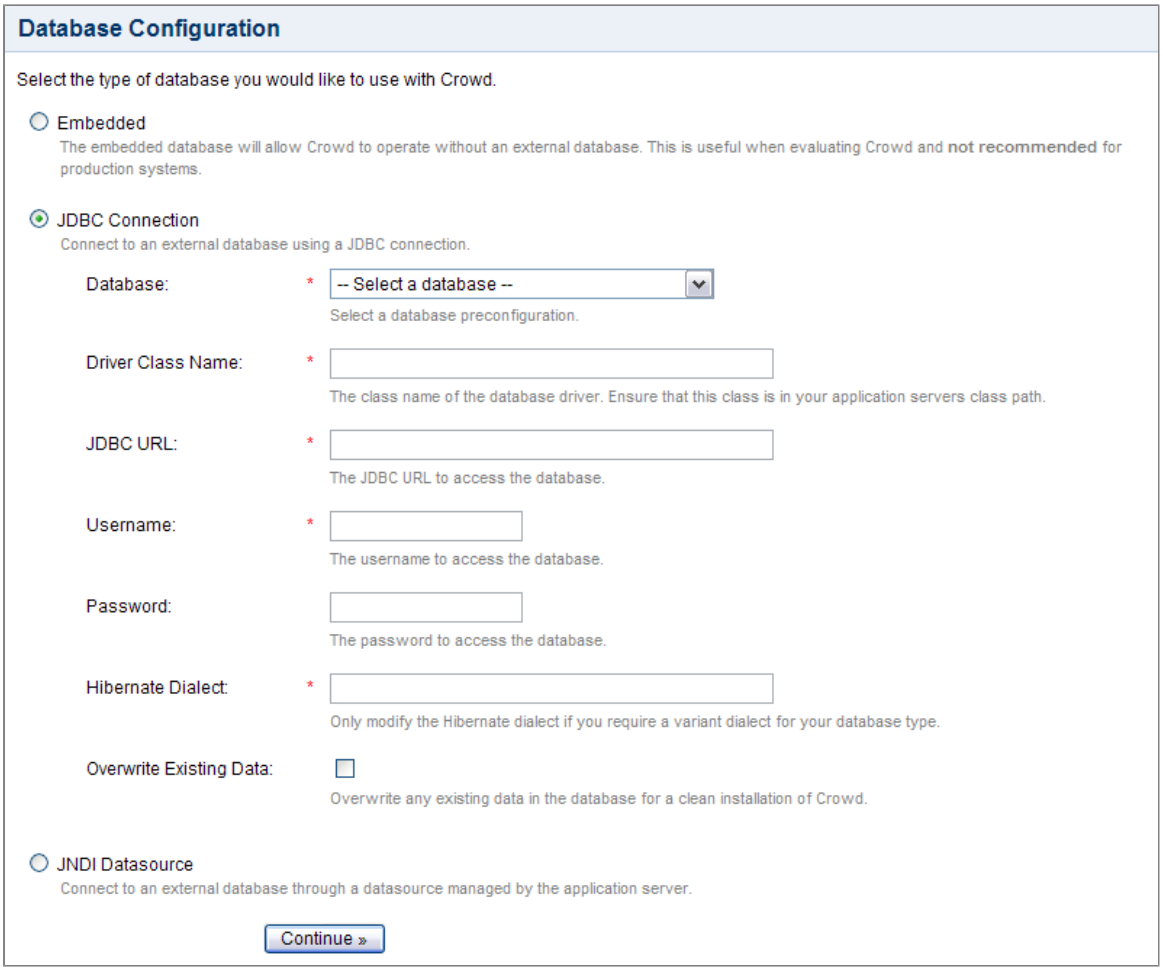
Database Option 1: Embedded HSQLDB (For Evaluation Purposes Only)



Crowd 'Standalone' is shipped with an embedded HSQLDB database. If you choose the '**Embedded**' option, the data files are stored in the Crowd Home directory, as configured on installation.

The HSQLDB database is fine for evaluation purposes, but for production installations you should connect Crowd to an enterprise database using the JDBC or JNDI datasource connections described below. This also lets you take advantage of your existing database backup and recovery procedures.

Database Option 2: JDBC Connection



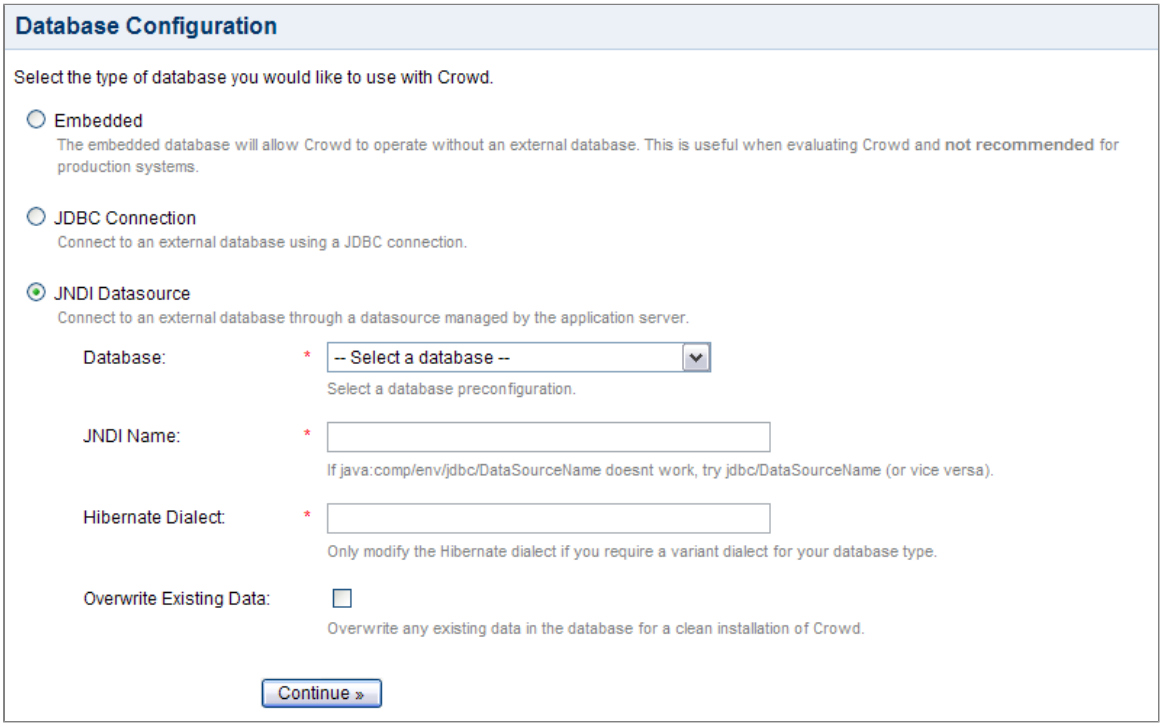
Select the '**JDBC Connection**' if you want to connect to an external database via a JDBC connection. (If you have not yet created your database for Crowd, follow the database setup instructions.)

Supply the details for your database:

Field	Description
Database	Select your database server type.
Driver Class Name	Enter the class name for your database driver. Make sure that the class is in the class path on your application server. See guidelines on creating your specific database.

JDBC URL	Enter the URL at which Crowd can access the database JDBC connection.
Username	Enter the username which Crowd will use to access the database.
Password	Enter the password corresponding to the above username.
Hibernate Dialect	This is the Hibernate configuration for the selected database type. The Crowd installation will supply a default dialect for the database type you have chosen. You should only alter this dialect if you need an alternative for the database type or are using an unsupported database type. • To configure Crowd to support Unicode in **MS SQL Server 2005 and 2008**, enter the following in the '**Hibernate Dialect**' field on the Crowd Setup Wizard's Database Configuration screen: `com.atlassian.crowd.util.persistence.hibernate.SQLServerIntlDialect`
Overwrite Existing Data	Crowd will ask you to confirm that existing data should be overwritten, if both of the following are true: • You chose '**New Installation**' or '**Import data from an XML Backup**' in *Step 3* above, and • The database configured on the above screen already exists and contains Crowd data.

Database Option 3: JNDI Datasource



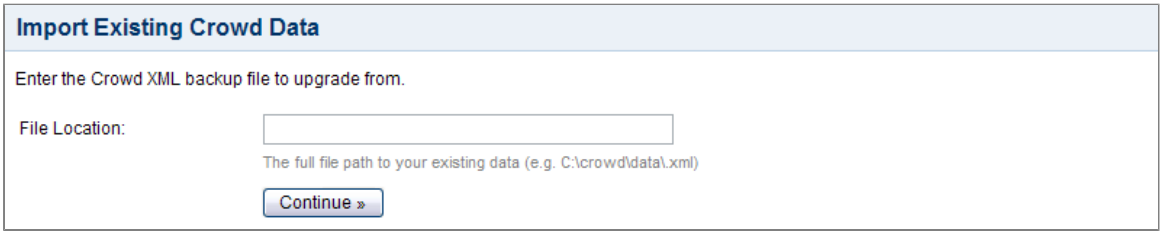
Select the '**JNDI Datasource**' if you want to connect to an external database via a datasource managed by your application server.

Supply the details for your database:

Field	Description
Database	Select your database server type.
JNDI Name	Enter the datasource name, e.g. `jdbc/CrowdDS` or `java:comp/env/jdbc/CrowdDS`.
Hibernate Dialect	This is the Hibernate configuration for the selected database type. The Crowd installation will supply a default dialect for the database type you have chosen. You should only alter this dialect if you need an alternative for the database type or you have selected an unsupported database type. • To configure Crowd to support Unicode in **MS SQL Server 2005 and 2008**, enter the following in the '**Hibernate Dialect**' field on the Crowd Setup Wizard's Database Configuration screen: `com.atlassian.crowd.util.persistence.hibernate.SQLServerIntlDialect`

Overwrite Existing Data	Crowd will prompt you to confirm that existing data should be overwritten, if both of the following are true: • You chose '**New Installation**' or '**Import data from an XML Backup**' in *Step 3* above, and • The database configured on the above screen already exists and contains Crowd data.

Step 5. (*Optional*) Import Existing Crowd Data

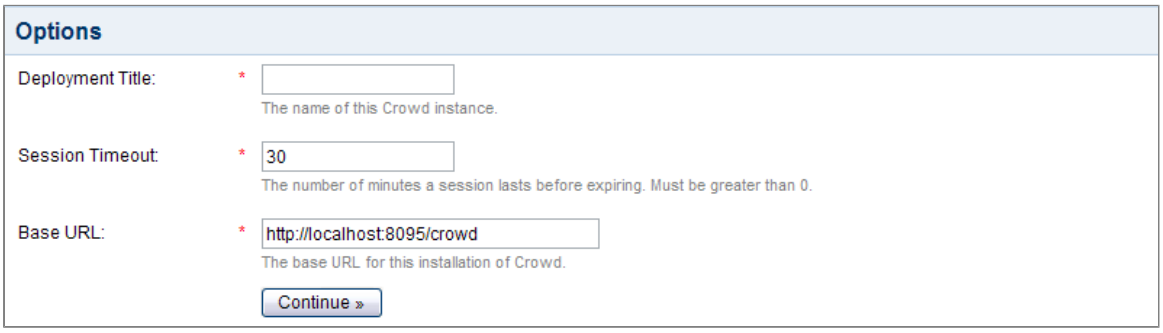


This screen will appear only if you selected '**Import data from an XML Backup**' in *Step 3* above.

In '**File Location**', enter the full path to your XML backup file including the name of the XML file.

> **Upgrading from an existing Crowd installation?**
> If you have connected to an existing database or imported your data from XML, the setup will be complete once you have clicked 'Continue' on the above screen. See *Step 11* below and read more about upgrading Crowd.

Step 6. Options



This part of the setup process allows you to specify general options for the Crowd server.

- The deployment title is a unique name for your Crowd instance. The deployment title is used by default in the subject line of email notifications.

  You can change this value later, via the Crowd Administration Console.

- The session timeout determines how long a session will be considered valid during any period of inactivity. This value is specified in minutes and must be greater than `0`.

  You can change this value later, via the Crowd Administration Console.

- The base URL is the website address of the Crowd server. This value is used during startup to put the correct values into the `crowd.properties` file for the Crowd Administration Console.
  There is no option to change this value via the Crowd Administration Console, because the URL must not be changed while Crowd is running.

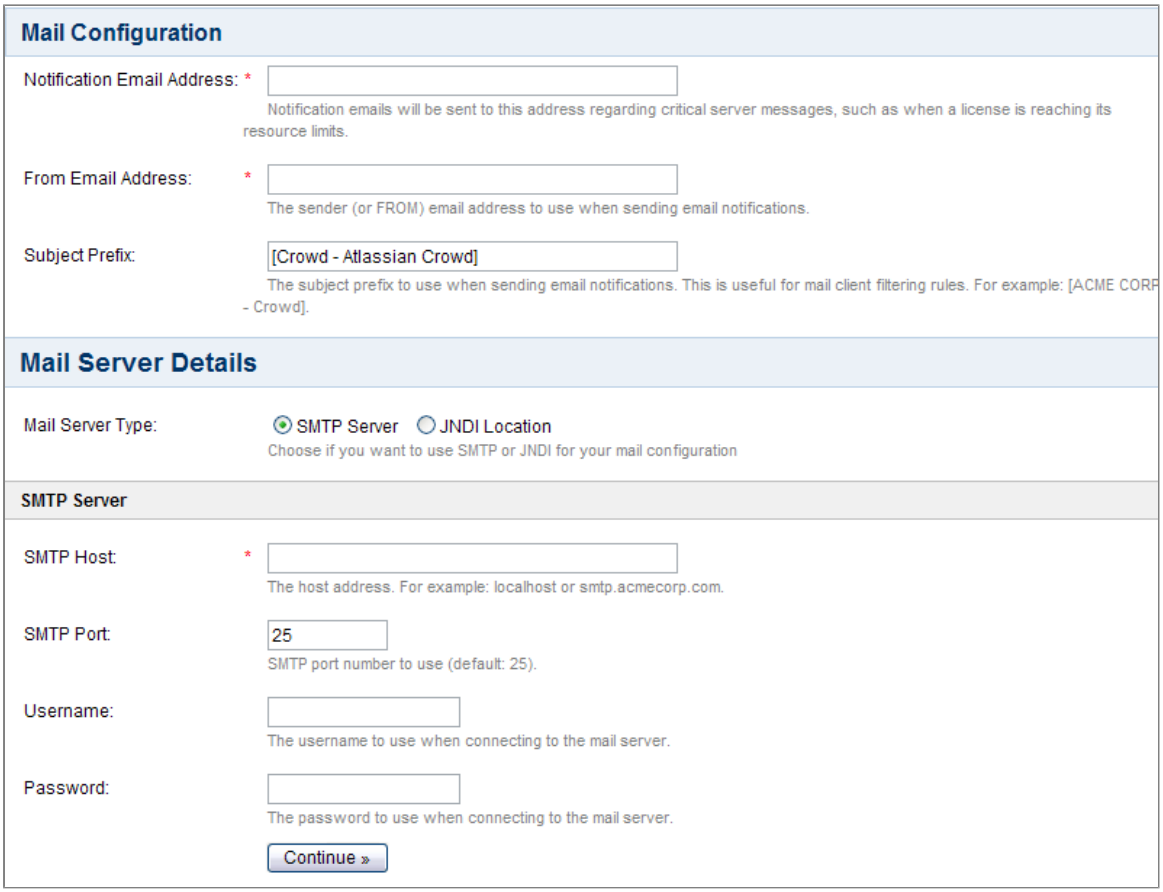
Step 7. Mail Server

Crowd can send email notifications to users for specific events, such as when a password is reset.

The '**Mail Configuration**' screen allows you to choose between an SMTP and a JNDI mail server, as described below.

> **If in any doubt, choose the 'SMTP Server' option for evaluation purposes.**

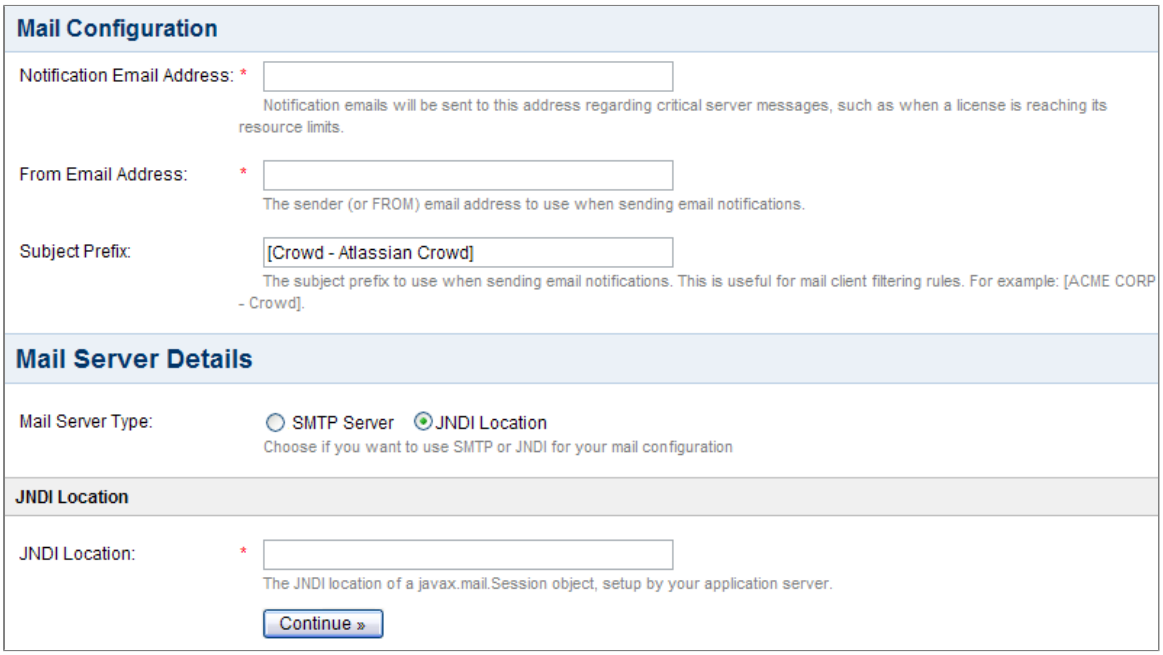
Mail Server Option 1: SMTP



Enter the details as follows:

- **Notification Email Address** — The email address which will receive notifications about server events.
- **From Email Address** — Crowd will add this email address as the 'sender' on the emails generated by Crowd and sent to users.
- **Subject Prefix** — The prefix which will appear at the start of the email subject, for all emails generated by Crowd. This can be useful for email client programs that offer filtering rules.
- **Mail Server Type** — Select the '**SMTP Server**' radio button.
- **SMTP Host** — The hostname of the SMTP mail server, e.g. 'localhost' or 'smtp.acme.com'.
- **SMTP Port** — The port on which the SMTP mail server listens. The default is '25'.
- **Username** — The username that your Crowd server will use when it logs in to your mail server.
- **Password** — The password that your Crowd server will use when it logs in to your mail server.

Mail Server Option 2: JNDI Location

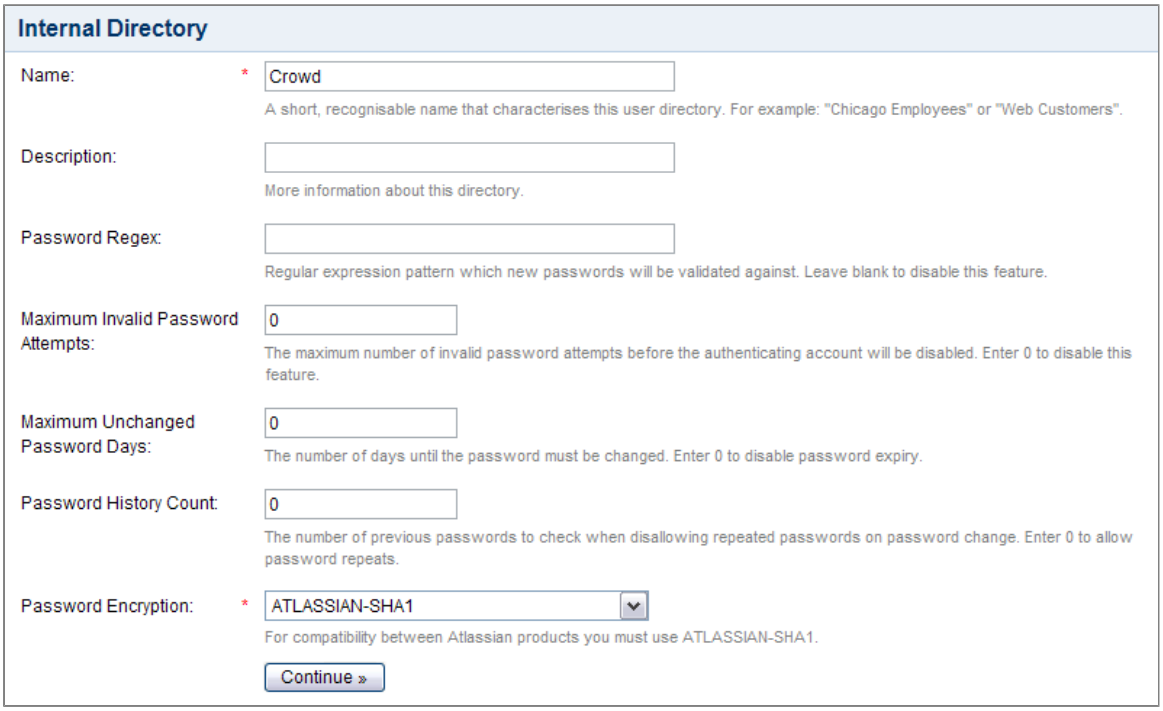


Select the '**JNDI Location**' if you want to connect to a mail server via a datasource managed by your application server.

Enter the details as follows:

- **Notification Email Address** — The email address which will receive notifications about server events.
- **From Email Address** — Crowd will add this email address as the 'sender' on the emails generated by Crowd and sent to users.
- **Subject Prefix** — The prefix which will appear at the start of the email subject, for all emails generated by Crowd. This can be useful for email client programs that offer filtering rules.
- **Mail Server Type** — Select the '**JNDI Location**' radio button.
- **JNDI Location** — The datasource name of a `javax.mail.Session` object which has been set up by your application server.

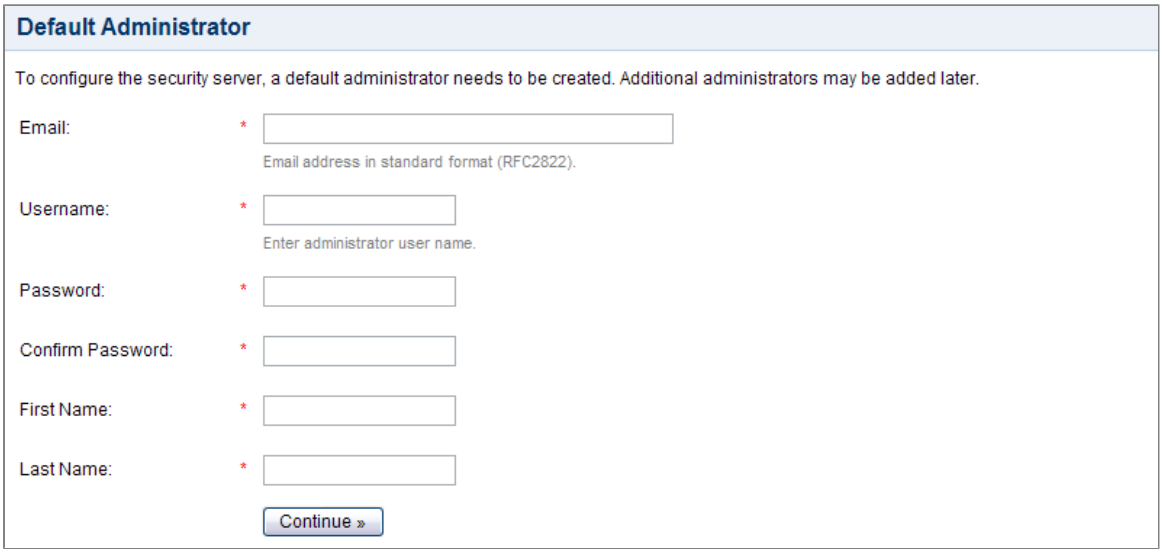
Step 8. Default Directory



Please configure a default user directory. For information about configuring different types of directories (Internal, LDAP, Delegated Authentication or Custom) refer to Adding a Directory.

Crowd administrators group is in default directory
The default group `crowd-administrators` will be automatically created in the default directory. Members of this group have rights to administer Crowd.

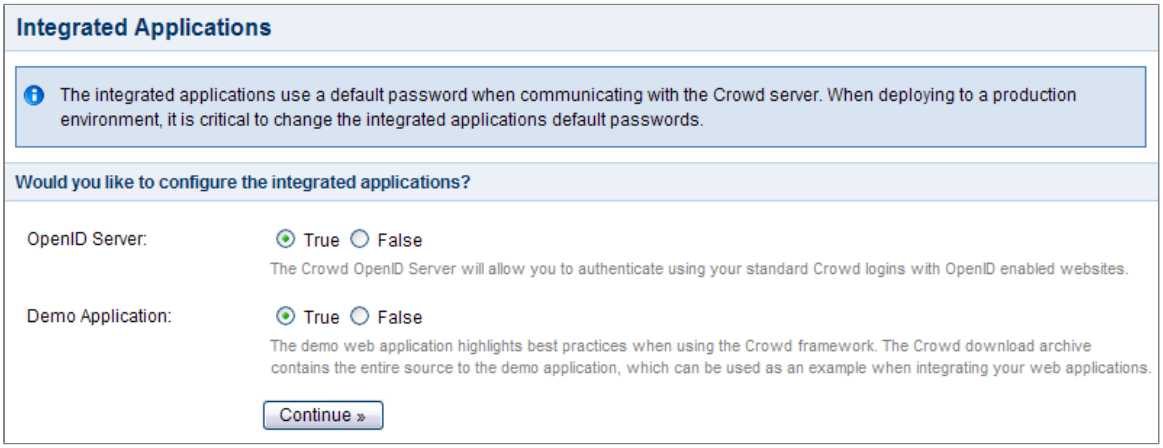
Step 9. Default Administrator



Please specify a default Crowd administrator. The default administrator will be automatically added to the default group

`crowd-administrators`, thereby giving them rights to access the Crowd Administration Console.

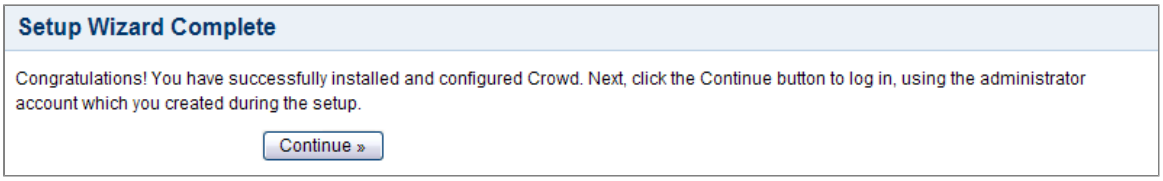
Step 10. Integrated Applications



You have the option to auto-configure two applications. We recommend that you select '**True**' for both:

- **OpenID Server** — This is the **CrowdID** application, which allows you to provide OpenID services for your end-users. For details please see the CrowdID Administration Guide and the CrowdID User Guide.
- **Demo Application** — The 'demo' application is an example of an application integrated with Crowd. It highlights best practices for using the Crowd framework, and is provided to assist you with quickly setting up and configuring Crowd. The Crowd download zip file (archive) contains the entire source for the 'demo' application, which you can use as an example when integrating your custom web applications.

Step 11. Setup Complete



You are now ready to use the Crowd Administration Console. For details, please see the Crowd Administration Guide.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Troubleshooting your Configuration on Setup

This page describes the '**Crowd Checklist**' screen and tells you how to use the screen to troubleshoot your initial Crowd configuration. The '**Crowd Checklist**' screen may appear when you start the Setup Wizard after installing Crowd.

The 'Crowd Checklist' appears only if there is an error in your environment configuration, preventing you from completing the Setup Wizard.

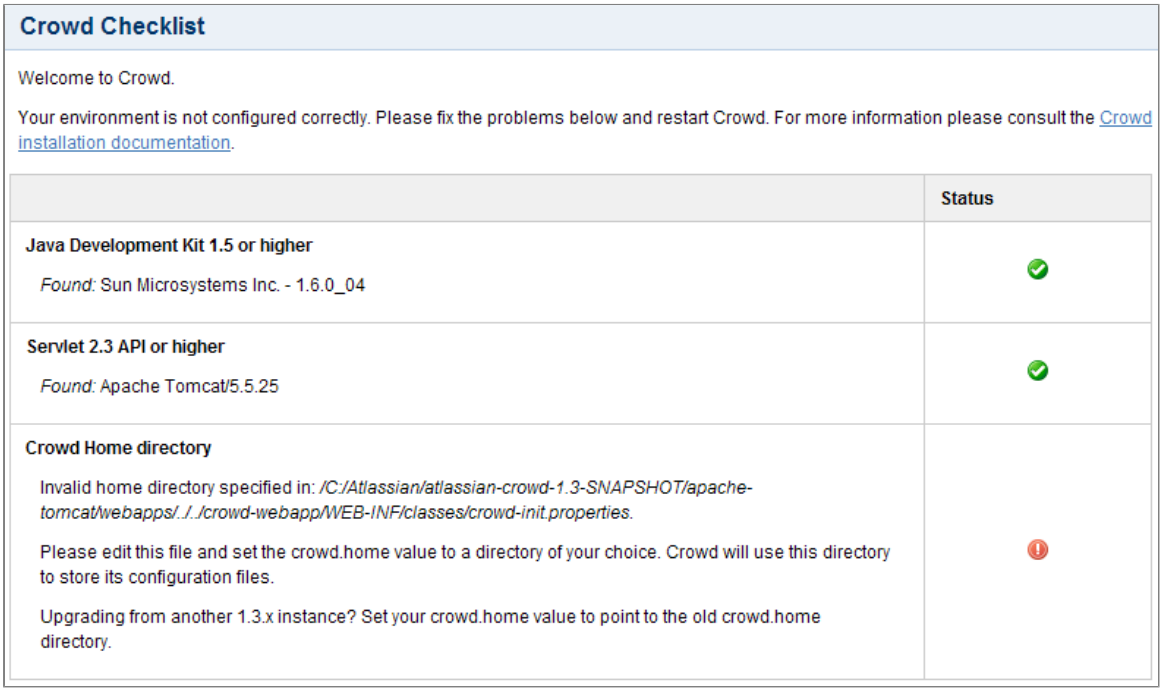
Troubleshooting your Configuration Problems

The 'Crowd Checklist' shows a list of environmental requirements on the left and a 'Status' for each setting on the right. A red exclamation mark ( ) in the '**Status**' column indicates a problem with one of the settings.

Environmental Requirement	Possible Error Message	Solution
Java Development Kit 1.5 or higher	(*The screen will show the version of JDK detected in your system, with a red exclamation mark in the 'Status' column if insufficient.*)	Refer to the System Requirements page for information about the JDK required and where you can get it.

Servlet 2.3 API or higher	(*The screen will show the application server and version detected in your system, with a red exclamation mark in the 'Status' column if insufficient.*)	Make sure that the servlet container on your application server supports the Servlet 2.3 specification. Note: Crowd ships with Apache Tomcat (5.5.x) which is compliant.
Crowd Home directory	Invalid home directory specified in {CROWD-INSTALL}/crowd-webapp/WEB-INF/classes/crowd-init.properties. Please edit this file and set the crowd.home value to a directory of your choice. Crowd will use this directory to store its configuration files.	Define the directory which you want Crowd to use as its '**home**'. Read all about it in the installation guide.

Screenshot: 'Crowd Checklist'



The above screenshot shows a problem with the setting of the Crowd home directory.

RELATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Configuring Crowd

You can configure Crowd to suit your environment, as described on the following pages:

- Important Directories and Files
- Changing the Port that Crowd uses
- Configuring Crowd to Work with SSL

RELATED TOPICS

- Specifying your Crowd Home Directory
- Configuring an SSL Certificate for Microsoft Active Directory
- Troubleshooting your Configuration on Setup

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

Important Directories and Files

This page contains information about the important directories and files to be aware of when configuring Crowd.

On this page:

- The Crowd Home Directory
  - The `crowd.properties` File
  - The `crowd.cfg.xml` File
  - The `bundled-plugins` Directory in Crowd Home
  - The `caches` Directory in Crowd Home
  - The `database` Directory in Crowd Home
  - The `plugin-data` Directory in Crowd Home
  - The `plugins` Directory in Crowd Home
- The Crowd Installation Directory
  - The `crowd-init.properties` File
  - The `build.properties` File
  - The `build.xml` File

When configuring an application to work with Crowd, you will be interested in the crowd.properties file.

The Crowd Home Directory

The Crowd Home directory is where Crowd will store its configuration information. If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will also store its database in this directory.

The location of this directory is specified in the `crowd-init.properties` file described below. You can set the location during installation.

Crowd's System Information screen shows the location of your Crowd Home directory.

Important files and directories in the Crowd Home directory, listed here and described below:

- The `crowd.properties` File
- The `crowd.cfg.xml` File
- The `bundled-plugins` Directory in Crowd Home
- The `caches` Directory in Crowd Home
- The `database` Directory in Crowd Home
- The `plugin-data` Directory in Crowd Home
- The `plugins` Directory in Crowd Home

The `crowd.properties` File

The `crowd.properties` file, containing application configuration settings for the Crowd Administration Console application, is located at the root of your Crowd Home directory.

For more information, refer to the page about the crowd.properties File.

The `crowd.cfg.xml` File

This file stores configuration information for the Crowd Administration Console application, including:

- License information
- Server ID
- Database configuration properties
- Setup phase reached.

The contents of this file is automatically generated when you run the Crowd Setup Wizard.

The file is located at the root of your Crowd Home directory.

Here's an example of the content of `crowd.cfg.xml`, when the embedded HSQL database was specified at setup:

```
<?xml version="1.0" encoding="UTF-8"?>

<application-configuration>
    <setupStep>complete</setupStep>
    <setupType>install.new</setupType>
    <buildNumber>320</buildNumber>
    <properties>
        <property name="crowd.server.id">B9AN-B9AN-B9AN-B9AN</property>
        <property name="hibernate.c3p0.acquire_increment">1</property>
        <property name="hibernate.c3p0.idle_test_period">100</property>
        <property name="hibernate.c3p0.max_size">15</property>
        <property name="hibernate.c3p0.max_statements">0</property>
        <property name="hibernate.c3p0.min_size">0</property>
        <property name="hibernate.c3p0.timeout">30</property>
        <property name="hibernate.connection.driver_class">org.hsqldb.jdbcDriver</property>
        <property name="hibernate.connection.password"></property>
        <property name="hibernate.connection.url"
>jdbc:hsqldb:C:/data/crowd-home-15/database/defaultdb</property>
        <property name="hibernate.connection.username">sa</property>
        <property name="hibernate.dialect">org.hibernate.dialect.HSQLDialect</property>
        <property name="hibernate.setup">true</property>
        <property name="license">AAABGQ0ODAoPeNpdkF1LwzAUhu/plus-some-more-stuff</property>
    </properties>
</application-configuration>
```

The `bundled-plugins` Directory in Crowd Home

The `bundled-plugins` directory is a sub-directory of your Crowd Home directory. It contains plugins which are shipped with your Crowd installation, such as:

- The SAML integration plugin which provides the Google Apps SSO feature.
- The Shared Access Layer (SAL) plugins.
- The REST module plugin.
- And more.

The plugins are a collection of jars generated when you install the Crowd web application. The jars are obtained by unzipping `atlassian-bundled-plugins.zip` from `{CROWD_INSTALL}\crowd-webapp\WEB-INF\classes`.

The `caches` Directory in Crowd Home

The `caches` directory is a sub-directory of your Crowd Home directory. It contains various files that Crowd caches to improve performance. The files in sub-directories of this directory are either created or updated generated when you install or restart the Crowd web application.

Do not modify or remove these files while Crowd is running. It should be safe for you to delete these files between application restarts.

It may improve Crowd's performance if you link this sub-directory to a fast disk.

The `database` Directory in Crowd Home

If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will store its database in this directory.

The `plugin-data` Directory in Crowd Home

The `plugin-data` directory is a sub-directory of your Crowd Home directory. This is a place for plugins to store their data. The directory will be created the first time a plugin needs it. For example, if you configure the Google Apps Connector, then the connector's SSO Keys will be stored in the `plugin-data` directory.

The `plugins` Directory in Crowd Home

The `plugins` directory is a sub-directory of your Crowd Home directory. This directory will contain plugins that are not shipped with Crowd and that you have installed separately onto your Crowd instance.

The Crowd Installation Directory

This is the directory into which the downloaded Crowd application has been unzipped during installation.

Important files in the Crowd Installation directory, listed here and described below:

- The `crowd-init.properties` File
- The `build.properties` File
- The `build.xml` File

The `crowd-init.properties` File

This is where you specify your Crowd Home directory (described above). You can set the location during installation.

The `crowd-init.properties` file is located in the Crowd Installation directory at {`CROWD_INSTALL}\crowd-webapp\WEB-INF\classes\crowd-init.properties`

The file content looks something like this before it has been customised:

```
## You can specify your crowd.home property here or in your system environment variables.

# On Windows-based operating systems, uncomment the following
# line and set crowd.home to a directory Crowd should use to
# store its configuration.
# NOTE: use forward slashes instead of backward slashes

#crowd.home=c:/data/crowd-home

# On Unix-based operating systems, uncomment the following
# line and set crowd.home to a directory Crowd should use to
# store its configuration.

#crowd.home=/var/crowd-home
```

The `build.properties` File

This configuration file stores various deployment properties of Crowd and the 'demo' application.

The file is located at the root of your Crowd Installation directory (described above).

The default `build.properties` file will look similar to the following:

```
# Modify the attributes of this file to quickly adjust the deployment values of Crowd.

# The Hibernate database dialect to use.
hibernate.dialect=org.hibernate.dialect.HSQLDialect

# The Hibernate transaction factory to use.
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory

# The http port you wish to run crowd from, ie: http://localhost:8095/crowd
crowd.tomcat.connector.port=8095

# Tomcat requires a unique port for shutdown
crowd.tomcat.shutdown.port=8020

# Crowd context root
crowd.url=http://localhost:8095/crowd

# Demo context root
demo.url=http://localhost:8095/demo

# OpenID server context root
openidserver.url=http://localhost:8095/openidserver
```

Parameter	Description
hibernate.dialect	This parameter controls the database dialect the Hibernate persistence system will use when executing commands versus your database server.
hibernate.transaction.factory_class	This parameter controls the transaction factory to use when executing transactions at run-time: Hibernate provides two generic options, additional application server specific options are available: • `org.hibernate.transaction.JDBCTransactionFactory` delegates to database (JDBC) transactions (default). • `org.hibernate.transaction.JTATransactionFactory` delegates to JTA (if an existing transaction is under way, the work performed is done in that context. Otherwise a new transaction is started).
crowd.url	The path and port for the root of the Crowd Administration Console web-application.
demo.url	The path and port for the root of the Crowd demo web-application
openidserver.url	The path and port for the root of the CrowdID web-application

The `build.xml` File

This is an Ant script that loads properties from the `build.properties` configuration file.

The file is located at the root of your Crowd Installation directory (described above).

If configuring Crowd and/or the demo application to run on a port and context path other than the default, you will need to run the command `build.sh` (or `build.bat`) against the `build.xml` configuration file. This process will then edit all of the necessary Crowd configuration files for your deployment.

The sample output from running `build.xml` will look similar to the following:

```
shamid@mocha:~/atlassian-crowd-1.1.0$ ./build.sh
Buildfile: build.xml

init:

assistant:
 Changing Tomcat's connector port to 8095
 Changing Tomcat's shutdown port to 8020
Configuring the Crowd Console
Copying crowd.properties to: crowd-webapp/WEB-INF/classes
Copying 1 file to /home/shamid/atlassian-crowd-1.1.0/crowd-webapp/WEB-INF/classes
Configuring the Crowd hibernate configuration
Updating the HibernateDialect and TransactionFactory in
crowd-webapp/WEB-INF/classes/jdbc.properties
Updating property file:
/home/shamid/atlassian-crowd-1.1.0/crowd-webapp/WEB-INF/classes/jdbc.properties
Configuring the demo application
Renaming and copying demo.properties to: demo-webapp/WEB-INF/classes/crowd.properties
Copying 1 file to /home/shamid/atlassian-crowd-1.1.0/demo-webapp/WEB-INF/classes
Configuring the OpenID server application
Renaming and copying openidserver.properties to:
crowd-openidserver-webapp/WEB-INF/classes/crowd.properties
Copying 1 file to /home/shamid/atlassian-crowd-1.1.0/crowd-openidserver-webapp/WEB-INF/classes
Configuring the OpenID hibernate configuration
Updating the HibernateDialect and TransactionFactory in
crowd-openidserver-webapp/WEB-INF/classes/jdbc.properties
Updating property file:
/home/shamid/atlassian-crowd-1.1.0/crowd-openidserver-webapp/WEB-INF/classes/jdbc.properties

BUILD SUCCESSFUL
Total time: 2 seconds
```

RELATED TOPICS

- Finding the atlassian-crowd.log File

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
- Installing Crowd as a Windows Service

The crowd.properties File

When integrating an application with Crowd, you will copy Crowd's client library and the `crowd.properties` configuration file into the application's library. For details of the procedure, refer to Adding an Application.

The Crowd Administration Console application also has its own `crowd.properties` file, which is located at the root of your Crowd Home directory. (See Important Directories and Files for more about the Crowd Home directory.)

Attributes of the crowd.properties File

Attribute	Description
application.name	The name that the application will use when authenticating with the Crowd server. This needs to match the name you specified in Adding an Application.
application.password	The password that the application will use when authenticating with the Crowd server. This needs to match the password you specified in Adding an Application.
application.login.url	Crowd will redirect the user to this URL if their authentication token expires or is invalid due to security restrictions.

crowd.server.url	The URL to use when connecting with the integration libraries to communicate with the Crowd server.
session.isauthenticated	The session key to use when storing a `Boolean` value indicating whether the user is authenticated or not.
session.tokenkey	The session key to use when storing a `String` value of the user's authentication token.
session.validationinterval	The session key to use when storing an `Integer` value of the number of minutes between authentication validation. If this value is set to `0`, each HTTP request will be authenticated.
session.lastvalidation	The session key to use when storing a `Date` value of the user's last authentication.

The following **optional** attributes in the `crowd.properties` file allow further customisation of the client:

Attribute	Description	Default Value
http.proxy.host	The name of the proxy server used to transport SOAP traffic to the Crowd server.	(none)
http.proxy.port	The connection port of the proxy server (must be specified if a proxy host is specified).	(none)
http.proxy.username	The username used to authenticate with the proxy server (if the proxy server requires authentication).	(none)
http.proxy.password	The password used to authenticate with the proxy server (if the proxy server requires authentication).	(none)
http.max.connections	The maximum number of HTTP connections in the connection pool for communication with the Crowd server.	20
http.timeout	The HTTP connection timeout (milliseconds) used for communication with the Crowd server. A value of zero indicates that there is no connection timeout.	0
cookie.tokenkey	When using Crowd for single sign-on (SSO), you can specify the SSO cookie name for each application. Under the standard configuration, Crowd will use a single, default cookie name for all Crowd-connected applications. You can override the default with your own cookie name. As well as allowing you to define the SSO cookie name, this feature also allows you to divide your applications into different SSO groups. For example, you might use one SSO token for your public websites and another for your internal websites.	crowd.token_key

Passing crowd.properties as an Environment Variable

You can pass the location of a client application's `crowd.properties` file to the client application as an environment variable when starting the client application. This means that you can choose a suitable location for the `crowd.properties` file, instead of putting it in the client application's `WEB-INF/classes` directory.

This applies to the Crowd Administration Console's `crowd.properties` file too. You may find this particularly useful when integrating with a WAR deployment of an integrated application.

Example:

```
-Dcrowd.properties={FILE-PATH}/crowd.properties
```

RELATED TOPICS

Passing the crowd.properties File as an Environment Variable
Important Directories and Files
Adding an Application

Changing the Port that Crowd uses

By default, Crowd is configured to use port 8095. If this port is already in use within your network, you will need to change the port that Crowd uses.

Follow these steps:

1. Edit the `build.properties` file, as described in Important Directories and Files.
2. Change the `crowd.url` property to the new port on which the Crowd Administration Console will be accessed.
3. Change the `demo.url` property to the new port on which the Crowd 'demo' application will be accessed.
4. Change the `openidserver.url` property to the new port on which the CrowdID Server will be accessed.
5. Run the `build.xml` script, as described in Important Directories and Files.

RELATED TOPICS

- Supported Platforms
  - Setting JAVA_HOME
- Installing Crowd and CrowdID
  - Connecting Crowd to a Database

- HSQLDB
- MS SQL Server
- MySQL
- Oracle
- PostgreSQL
- Connecting CrowdID to a Database
  - HSQLDB for CrowdID
  - MS SQL Server for CrowdID
  - MySQL for CrowdID
  - Oracle for CrowdID
  - PostgreSQL for CrowdID
- Installing Crowd and CrowdID WAR Distribution
  - Installing Crowd WAR Distribution
    - Configuring Crowd and CrowdID on Tomcat
    - Installing Crowd WAR on JBoss
  - Installing CrowdID WAR Distribution
- Specifying your Crowd Home Directory

- Running the Setup Wizard
  - Troubleshooting your Configuration on Setup
- Configuring Crowd
  - Important Directories and Files
    - The crowd.properties File
  - Changing the Port that Crowd uses
  - Configuring Crowd to Work with SSL
- Installing Crowd as a Windows Service
  - Specifying Startup Order of Windows Services
  - Changing the User for the Crowd Windows Service
  - Removing the Crowd Windows Service
  - Troubleshooting Crowd as a Windows Service

Configuring Crowd to Work with SSL

When web applications are accessed across the internet, there is always the possibility of usernames and passwords being intercepted by intermediaries. These intercepts may occur when the data is travelling between a client and the server. It is often a good idea to enable access via HTTPS (HTTP over SSL) and require the use of HTTPS for pages where passwords are sent.

In some cases where transmitted data is sensitive, all pages should be accessed via HTTPS.

Note: Using HTTPS may result in slower performance.

> **What is SSL?**
> The Secure Sockets Layer (SSL) is a commonly-used protocol for managing the security of message transmission on the internet. SSL is included as part of most web browsers and web server products. For more information, take a look at Sun's Introduction to SSL.

On this page:

- Using Crowd over SSL
  - Step 1: Enable Tomcat SSL Access
  - Step 2: Create or Import your SSL Key (Self-Signed or CA-Issued)
    - Creating a Self-Signed SSL Key
    - Importing a CA-Issued Certificate
  - Step 3: Modify crowd.properties
  - Step 4: Create or Modify setenv.sh or setenv.bat
- Troubleshooting
- Using SSL between an LDAP Server and Crowd
  - Microsoft Active Directory Connector using SSL Certificate
  - Other LDAP Servers

Using Crowd over SSL

The process of enabling SSL access is specific to each application server, but specifying which pages require protection is generic. Below we describe the process for Tomcat, the application server bundled with Crowd.

Step 1: Enable Tomcat SSL Access

Edit `CROWD/apache-tomcat/conf/server.xml`, and at the bottom before the `</Service>` tag, add this section (or uncomment it if it's already there):

```
<Connector port="8443" maxHttpHeaderSize="8192" maxThreads="150"
minSpareThreads="25" maxSpareThreads="75" enableLookups="false"
disableUploadTimeout="true" acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS" />
```

If using Apache Tomcat 6, an extra attribute will be necessary: **SSLEnabled="true"**. Crowd 2.0.2 Standalone is the first Crowd version using Tomcat 6 and therefore must have this attribute in its SSL Connector.

This enables SSL access on port 8443. (The default for HTTPS is 443, but just as Tomcat uses 8080 instead of 80 to avoid conflicts, 8443 is used instead of 443 here).

Step 2: Create or Import your SSL Key (Self-Signed or CA-Issued)

You can either create a self-signed SSL key or import a certificate issued by a Certificate Authority (CA). We describe both methods below.

Creating a Self-Signed SSL Key

You can create a self-signed key for testing purposes with one of the following commands:

```
1.%JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA (Windows)
2.$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA  (Unix)
```

The keytool utility will prompt you for two passwords: the keystore password and the key password for Tomcat. You must use the same value for both passwords, and the value must be either:

1. 'changeit' (this is the default value Tomcat expects), or
2. if you use a value other than 'changeit', you must also specify this value in `conf/server.xml`. You must add the following attribute to the Connector tag described above:

```
1.keystorePass="<password value>"
```

For information on adding a key pair issued by a Certificate Authority (CA), refer to the section entitled 'Installing a Certificate from a Certificate Authority' in the Apache Tomcat documentation.

IE7 on Vista Issue
If your clients will access Crowd from Internet Explorer 7 on Vista, please ensure that you specify the **-keyalg RSA** flag. By default the SHA1 algorithm is used, which results in error 'Internet Explorer cannot display the webpage'.
Apparently on JDK 1.6 you also need to specify the **-sigalg MD5withRSA** flag since **-keyalg RSA** will still result in SHA1 being used.
If you like, you can refer to this Atlassian developer blog post for more information.

Importing a CA-Issued Certificate

When using certificates issued by a Certificate Authority, you also need import the certificate using the `keytool` command, rather than generating a self-signed key.

Here is an example of the command:

```
1.keytool -import -alias tomcat -file certificate.cer -keystore some/path/to/file -storepass
something.secure
```

The `-file` is your certificate and the `-keystore` is an optional destination, but it will guarantee that you know where your keystore is. By default, the keystore is placed in your user home directory. You can refer to the following Sun documentation for more information on the keytool:

- Solaris and Linux
- Windows

Try this blog post for a handy tutorial:

- Talkingtree blog post

Now edit the `server.xml` file as described in section 'Edit the Tomcat Configuration File' in the Apache Tomcat documentation. Basically, you'll need to add the `keystoreFile` and `keystorePass` to the SSL Connector definition to match your keystore settings.

Step 3: Modify crowd.properties

Modify your `<Crowd-Home-Directory>/crowd.properties` file to reflect your new SSL settings. For example:

```
01.#Wed Apr 09 12:36:21 EST 2008
02.session.lastvalidation=session.lastvalidation
03.session.isauthenticated=session.isauthenticated
04.application.password=password
05.application.name=crowd
06.session.validationinterval=0
07.crowd.server.url=https\://localhost:8443/crowd/services/
08.session.tokenkey=session.tokenkey
09.application.login.url=https\://localhost:8443/crowd/console/
```

When changing crowd to use ssl after going through web based set up, `<Crowd-Home-Directory>/crowd.properties`, `<Crowd-install>/build.properties`, and `<Crowd-install>/client/conf/crowd.properties` need to be updated with `https\://host:port/...` Just updating `crowd.properties` is not enough. The symptom is unable to log in from the web interface, and the logs show xfire unable to message with the web service.

Step 4: Create or Modify setenv.sh or setenv.bat

In order to ensure that XFire calls work over SSL you will need to pass keystore values to the JVM. To do this either edit or create a `setenv.sh` or `setenv.bat` file located in Tomcat's bin directory: `apache-tomcat/bin/setenv.sh or setenv.bat`

The contents of the file should look similar to this:

```
1.JAVA_OPTS="-Xms128m -Xmx256m $JAVA_OPTS -Djavax.net.ssl.keyStore=/<pathtokeystore>/.keystore
-Djavax.net.ssl.keyStorePassword=changeit -Djavax.net.ssl.trustStore=/<pathtokeystore>/.keystore
-Djavax.net.ssl.trustStorePassword=changeit"
```

Replace `<pathtokeystore>` with the path to your .keystore file and the password with your keystore's password if modified.

Now restart your Crowd instance. You should be able to access Crowd at this URL:

```
1.https://localhost:8443/crowd/console
```

Troubleshooting

Here are some troubleshooting tips if you are using a self-signed key created by keytool, as described above.

When you enter 'https://localhost:8443' in your browser, if you get a message such as 'Cannot establish a connection to the server at localhost:8443', look for error messages in your `logs/catalina.out` log file. Here are some possible errors with explanations:

Can't Find the Keystore

```
java.io.FileNotFoundException: /home/<username>/.keystore (No such file or directory)
```

This indicates that Tomcat cannot find the keystore. The keytool utility creates the keystore as a file called `.keystore` in the current user's home directory. For Unix/Linux the home directory is likely to be `/home/<username>`. For Windows it is likely to be `C:\Documents And Settings\<UserName>`.

Make sure you are running Crowd as the same user who created the keystore. If this is not the case, or if you are running Crowd on Windows as a service, you will need to specify where the keystore file is in `conf/server.xml`. Add the following attribute to the connector tag you uncommented: `keystoreFile="<location of keystore file>"`

Incorrect Password

```
java.io.IOException: Keystore was tampered with, or password was incorrect
```

You used a different password than 'changeit'. You must either use 'changeit' for both the keystore password and for the key password for Tomcat, or if you want to use a different password, you must specify it using the `keystorePass` attribute of the Connector tag, as described above.

Passwords don't Match

```
java.io.IOException: Cannot recover key
```

You specified a different value for the keystore password and the key password for Tomcat. Both passwords must be the same.

To find out more about the options that Tomcat offers, please take a look at the Apache Tomcat documentation.

Using SSL between an LDAP Server and Crowd

Microsoft Active Directory Connector using SSL Certificate

Please refer to Configuring an SSL Certificate for Microsoft Active Directory.

Other LDAP Servers

For other LDAP servers, please consult your LDAP server documentation.

On the Crowd side, when configuring the connector properties, you will have to simply check the '**Secure SSL**' box and make sure you use the correct port in the '**URL**' field (usually 636).

RELATED TOPICS

Configuring an SSL Certificate for Microsoft Active Directory
Configuring Crowd

Installing Crowd as a Windows Service

For long-term use, you should configure Crowd to restart automatically when the operating system restarts. For Windows servers, this means configuring Crowd to run as a Windows service.

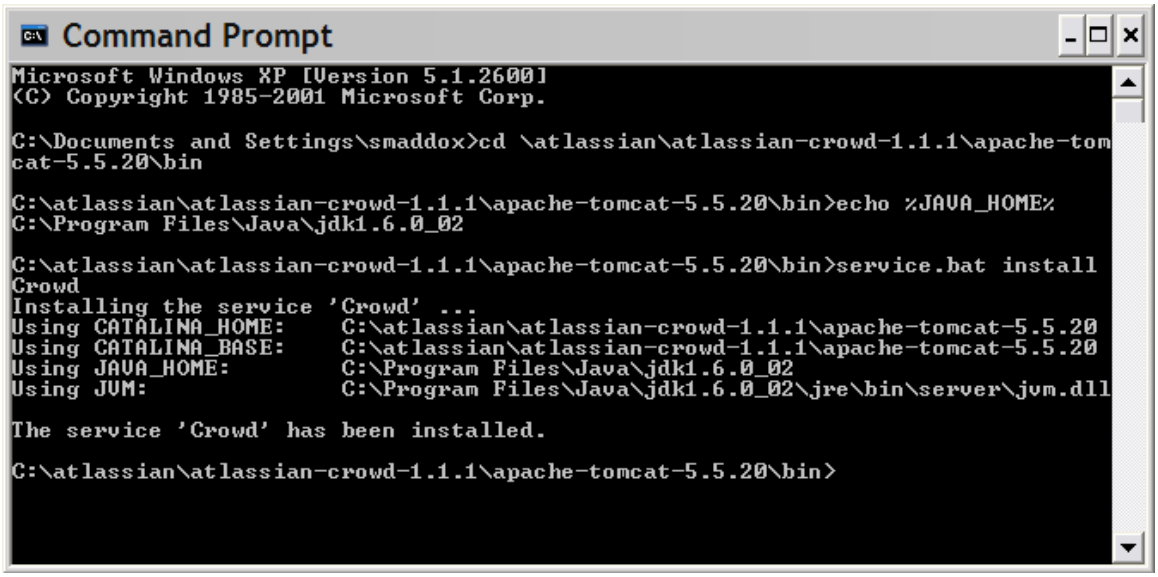
Running Crowd as a Windows service has other advantages. When Crowd is started manually, a console window opens - there is a risk that someone may accidentally shut down Crowd by closing the window. Also, the Crowd logs are properly managed by the Windows service (reliably found in `\atlassian-crowd.log` in the root Crowd directory, and rotated by file size).

Installing Crowd as a Windows Service

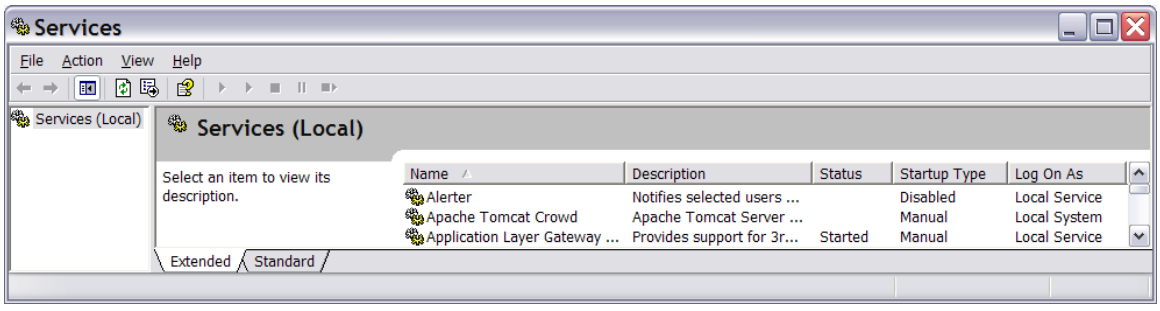
1. Open a DOS prompt.
2. 'cd' to your Crowd directory, and then the Tomcat `bin` subdirectory, e.g. `{CROWD_INSTALL}\apache-tomcat\bin`
3. If a directory in the path has spaces (e.g. `C:\Program Files\..`), please convert it to its eight-character equivalent (e.g. `c:\Progra~1\..`).
4. Ensure the JAVA_HOME variable is set to the JDK base directory. Use `echo %JAVA_HOME%` to confirm this.
5. Run the following command:

```
1.service.bat install Crowd
```

Screenshot: Installing Crowd as a Windows Service



Crowd should now have been installed as a service, and will be visible in the Windows Services console.
Screenshot: Windows Services Console



6. Run the following command, to have the Crowd service start automatically when the server starts:

```
1.tomcat6 //US//Crowd --Startup auto
```

The Crowd service will automatically start up the next time the server reboots.

- You can manually start the Crowd service with the command `net start Crowd`, and stop it with `net stop Crowd`.
- To see what parameters the Crowd service is starting with, go to **Start -> Run** and run `regedt32.exe`. There should be an entry at `HKEY_LOCAL_MACHINE -> SOFTWARE -> Apache Software Foundation -> Procrun 2.0 -> Crowd`.

Additional Crowd Setup Options (Optional)

- To increase the maximum memory Crowd can use (the default will already be 256MB), run:

```
1.tomcat6 //US//Crowd --JvmMx 512
```

- If you are running Crowd with JIRA and/or Confluence in the same JVM, increase the MaxPermSize to 512 MB:

```
1.tomcat6 //US//Crowd ++JvmOptions="-XX:MaxPermSize=512m"
```

- Occasionally, it may be useful to view Crowd's Garbage Collection information. This is especially true when investigating memory issues.
  - To turn on the Verbose GC (garbage collection) logging, execute the following command in the command prompt

```
1.tomcat6 //US//Crowd ++JvmOptions="-Xloggc:path\to\logs\atlassian-gc.log"
```

  - The path (denoted by \path\to) refers to the directory in which Crowd is currently installed. For example:

```
1.tomcat6 //US//Crowd ++JvmOptions="-Xloggc:c:\crowdinstall\logs\atlassian-gc.log"
```

- If you are using HSQL as your database server: after installing Crowd as a Windows service, you will need to copy your database files.
  1. Create a folder called c:\windows\system32\database
  2. Copy over the database files from your `atlassian-crowd-1.1.2/database`.

     We recommend strongly that you use an external database server rather than the HSQL database supplied with Crowd for evaluation purposes.

     Refer to the Tomcat documentation for further service options.

RELATED TOPICS

- Specifying Startup Order of Windows Services
- Changing the User for the Crowd Windows Service
- Removing the Crowd Windows Service
- Troubleshooting Crowd as a Windows Service

Specifying Startup Order of Windows Services

This page is relevant if you have installed Crowd as a Windows service.

If you have multiple Windows services that depend on each other, it is important that they are started in the correct order. For example, if you are running both JIRA and Crowd, it is important to start Crowd first, so that Crowd is running before people try to login to JIRA.

For information about specifying the startup order for multiple services, please refer to http://support.microsoft.com/kb/193888.

Related Topics

- Specifying Startup Order of Windows Services
- Changing the User for the Crowd Windows Service
- Removing the Crowd Windows Service
- Troubleshooting Crowd as a Windows Service

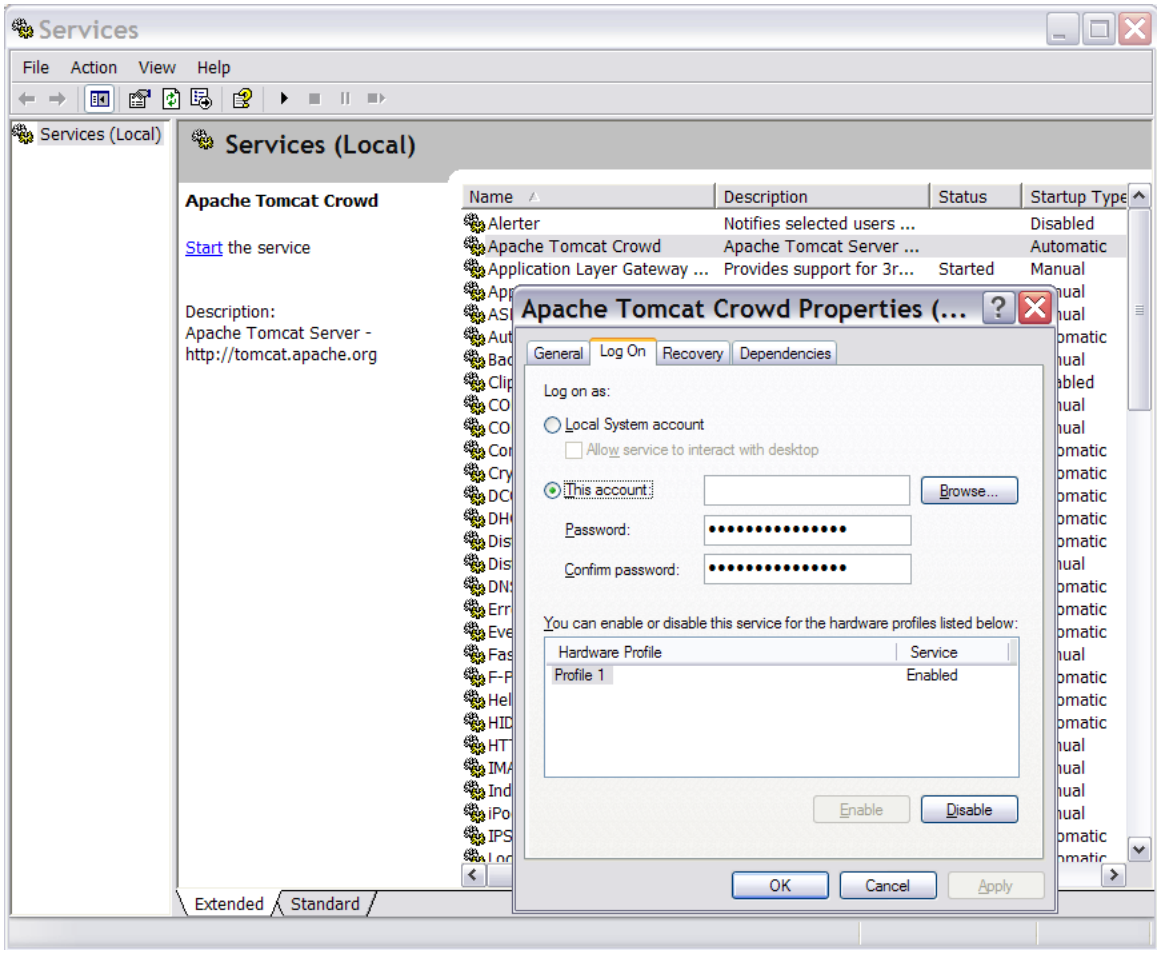
- Installing Crowd as a Windows Service

Changing the User for the Crowd Windows Service

This page is relevant if you have installed Crowd as a Windows service. You may want to change the user under which the Crowd Windows service is running, for security reasons.

Changing the Windows User for the Crowd Service

1. Navigate to the service: **Control Panel** -> **Administrative Tools** -> **Services**.
2. Locate the '**Apache Tomcat Crowd**' service, right-click and view the '**Properties**'.
3. Go to the '**Log On**' tab and change the user as desired.
   Screenshot: Changing the User for the Windows Service



RELATED TOPICS

- Specifying Startup Order of Windows Services
- Changing the User for the Crowd Windows Service
- Removing the Crowd Windows Service
- Troubleshooting Crowd as a Windows Service

- Installing Crowd as a Windows Service

Removing the Crowd Windows Service

This page is relevant if you have installed Crowd as a Windows service

To remove the Crowd Windows service:

1. Open a DOS prompt.
2. 'cd' to your Crowd directory, and then the Tomcat `bin` subdirectory, e.g. `{CROWD_INSTALL}\apache-tomcat-5.5.20\bin`
3. Run one of the following commands:
    - Either:

      ```
      1.service.bat remove Crowd
      ```

    - Or if the above does not work, use

      ```
      1.tomcat5 //DS//Crowd
      ```

RELATED TOPICS

- Specifying Startup Order of Windows Services
- Changing the User for the Crowd Windows Service
- Removing the Crowd Windows Service
- Troubleshooting Crowd as a Windows Service

- Installing Crowd as a Windows Service

Troubleshooting Crowd as a Windows Service

This page is relevant if you have installed Crowd as a Windows service.

Problems may occur when trying to set up Crowd to run as a Windows service with JDK 1.6. The problem is caused by a failure to locate `MSVCR71.DLL`, which can be found in your `%JAVA_HOME%/bin`. There are two options to resolve this problem:

- Add `%JAVA_HOME/bin` to `PATH`, then restart the server.
- Or copy `MSVCR71.DLL` to system path: either `C:\WINDOWS\SYSTEM32` or `C:\WINNT\SYSTEM32`

Notes for Windows Server 64-bit Operating Systems

Windows Server 64-bit will not start Crowd as a service as the `tomcat.exe` that ships is 32-bit. Install a 64-bit JDK and set JAVA_HOME to its location. Then follow the same steps above for Installing Crowd as a Windows Service. You'll need to replace { `CROWD_INSTALL}\apache-tomcat-5.5.20\bin\tomcat.exe` with one compiled for 64-bit from this location: http://svn.apache.org/viewvc/tomcat/tc5.5.x/trunk/connectors/procrun/bin/.

RELATED TOPICS

- Specifying Startup Order of Windows Services
- Changing the User for the Crowd Windows Service
- Removing the Crowd Windows Service
- Troubleshooting Crowd as a Windows Service

- Installing Crowd as a Windows Service

Upgrading Crowd

Below are instructions on upgrading an existing Crowd installation to the latest version of Crowd.

In summary, you will need to:

- Back up your Crowd database to XML before starting the upgrade.
- Do a clean installation of Crowd, pointing to a new Crowd Home directory.
- Restore your database from the XML backup as part of the setup process.

There is no automated process for the upgrade to Crowd 2.0, because we have changed Crowd's database schema in this version of Crowd.

On this page:

- Preparation: Read the Release Notes and Upgrade Notes
- Step 1. Export your Crowd Database to XML
- Step 2. Shut down Crowd and All Integrated Applications
- Step 3. Back Up your Crowd Files
- Step 4. Download and Re-Install Crowd
- Step 5. Start Crowd and Run the Setup Wizard
- Step 6. Update your Integrated Applications
- Troubleshooting

Preparation: Read the Release Notes and Upgrade Notes

Please read:

- The Release Notes for the version you are upgrading to, and
- The Upgrade Notes for any versions you are skipping as well as the version you are upgrading to:
  - Crowd 2.0 Upgrade Notes
  - Crowd 1.6 Upgrade Notes
  - Crowd 1.5 Upgrade Notes
  - Crowd 1.4 Upgrade Notes
  - Crowd 1.3 Beta Upgrade Notes
  - Crowd 1.3 Upgrade Notes
  - Crowd 1.2 Upgrade Notes
  - Crowd 1.1 Upgrade Notes
  - Crowd 1.0 Upgrade Notes

Step 1. Export your Crowd Database to XML

In the Crowd Administration Console, click the '**Administration**' tab and then click '**Backup**'. Follow the screen prompts to back up your Crowd database to an XML file. For full instructions, see our guide on backing up data.

Step 2. Shut down Crowd and All Integrated Applications

Shut down Crowd and all Crowd-connected applications.

Step 3. Back Up your Crowd Files

1. Use your database backup tools to back up your Crowd database and your CrowdID database. We **highly recommend** this step, in case something goes wrong during the upgrade process and you need to restore your data from backup.

2. Make backup copies of the following files:
   - The crowd.properties file for the **CrowdID** application, located at `{CROWD_INSTALL}/crowd-openidserver-webapp/WEBINF/classes/crowd.properties` — You will need to copy this file to your new Crowd installation.
   - Your Crowd JDBC Driver if you have configured Crowd with a database.
   - Your customised configuration files, if you have installed Crowd as a WAR distribution — You will need to copy these files to your new Crowd installation.
   - Your Crowd Home directory, in the location specified in the `crowd-init.properties` file — Recommended in case something goes wrong during the upgrade process.

3. We recommend that you rename your existing {CROWD_INSTALL} directory, because legacy files may cause problems if you unzip the new Crowd installation into an existing directory.

Step 4. Download and Re-Install Crowd

1. Download Crowd.

2. Unzip the downloaded archive into a directory of your choice, taking note of the following:
   - Please make sure that your new {CROWD_INSTALL} directory has a different name from your old {CROWD_INSTALL} directory.
   - Please check your unzip program before extracting the downloaded archive – see the note on the Crowd installation front page.
   - Do not specify directory names that contain spaces.
   - We will refer to this installation directory, where you unzipped the archive, as **{CROWD_INSTALL}**.

3. Specify a **new Crowd Home** directory for your new Crowd installation, by editing the configuration file at `{CROWD_INSTALL}\crowd-webapp\WEB-INF\classes\crowd-init.properties`.

   The **Crowd Home** directory is where Crowd will store its configuration information. If you are using the embedded HSQL database, supplied for evaluation purposes, Crowd will also store its database in this directory. To specify the directory:

   - Open the `crowd-init.properties` file.
   - Choose the appropriate line in the file, depending upon your operating system (see below).
   - Remove the # at the beginning of the line.
   - Enter the name of the directory you want Crowd to use as its Home directory. For example,
     - On Windows:

       ```
       crowd.home=c:/data/crowd-home
       ```

       Note: On Windows, make sure you use forward slashes as shown above, not backward slashes.

     - On Mac and UNIX-based systems:

       ```
       crowd.home=/var/crowd-home
       ```

   - Save the `crowd-init.properties` file.

   Make sure you point the new Crowd installation to a **new** Crowd Home directory, so that Crowd will do a clean installation. Do not point it at your existing Crowd Home directory.

4. Copy the following files, saved in *Step 3* above, to your new Crowd installation folder:
   - Copy the `crowd.properties` file for the **CrowdID** application to your new `{CROWD_INSTALL}/crowd-openidserver-webapp/WEB-INF/classes` directory.
   - Copy your Crowd JDBC Driver if you have configured Crowd with a database.
   - If you have installed Crowd as a WAR distribution, copy your customised configuration files.

Step 5. Start Crowd and Run the Setup Wizard

1. Run the start-up script, found in your {CROWD_INSTALL} directory:
   - `start_crowd.bat` for Windows.
   - `start_crowd.sh` for Mac and Unix-based systems.

2. Point a web browser at http://localhost:8095/crowd where you will see the **Crowd Setup Wizard**.

3. Enter your license key on the '**License**' screen, as described in the instructions on the Setup Wizard.

4. When asked for your Installation Type, choose '**Import data from an XML Backup**'. This step is required, to import your Crowd data from the XML file which you created in *Step 1* above.
5. The Setup Wizard will now ask you to configure your database. Supply the JNDI datasource or JDBC connection details of a new database.
6. The Import Existing Crowd Data screen will appear. Enter the location of your XML backup file and click '**Continue**'.
7. The Setup Wizard is now complete. You are now ready to log in to the Crowd Administration Console, using your administrator account from your earlier Crowd installation.

Step 6. Update your Integrated Applications

1. Copy the new `{CROWD_INSTALL}\client\crowd-integration-client-X.X.X.jar` file to each Crowd-integrated application's `WEB-INF/lib` folder, replacing the existing `crowd-integration-client-X.X.X.jar` file.
   For details please see the configuration instructions for each application:
   - Integrating Crowd with Atlassian Bamboo
   - Integrating Crowd with Atlassian Confluence
   - Integrating Crowd with Atlassian CrowdID
   - Integrating Crowd with Atlassian Crucible
   - Integrating Crowd with Atlassian FishEye
   - Integrating Crowd with Atlassian JIRA
   - Integrating Crowd with Acegi Security
   - Integrating Crowd with Apache
   - Integrating Crowd with Jive Forums
   - Integrating Crowd with Spring Security
   - Integrating Crowd with Subversion
   - Integrating Crowd with a Custom Application
2. If you have installed Crowd on a new server, or changed Crowd's URL or port number, you will also need to edit the `crowd.properties` file in each integrated application accordingly.
3. For better caching, copy the new `{CROWD_INSTALL}\client\conf\crowd-ehcache.xml` file to each Crowd-integrated application's `WEB-INF/classes/` folder, replacing the existing file.
4. If you are using CrowdID with an external database, you will need to use the manual JNDI datasource configuration method to configure an external database connection.

Troubleshooting

If you have any problems during upgrade, please raise a support request at https://support.atlassian.com/ and attach your `atlassian-crowd.log` file so that we can help you find out what's gone wrong.

RELATED TOPICS

- Crowd Release Notes
- Installing Crowd
- Upgrading Crowd
- Migrating Crowd between Servers

Upgrade Notes

- Crowd 1.0 Upgrade Notes
- Crowd 1.1 Upgrade Notes
- Crowd 1.2 Upgrade Notes
- Crowd 1.3 Beta Upgrade Notes
- Crowd 1.3 Upgrade Notes
- Crowd 1.4 Upgrade Notes
- Crowd 1.5 Upgrade Notes
- Crowd 1.6 Upgrade Notes
- Crowd 2.0 Upgrade Notes

Crowd 1.0 Upgrade Notes

- All LDAP configuration now need to have filters set
- If you are using PostgreSQL you need to change the column name attributevalues.attributevalueid to attributevalues.ATTRIBUTEVALUEID (make it uppercase).

Crowd 1.1 Upgrade Notes

To upgrade to Crowd 1.1.x from 1.0.x or earlier,

- Follow the usual steps for upgrading crowd.
- Configure two additional web applications, as described below.

Configuring OpenID Server and OpenID Demo Client applications

In Crowd 1.1, two new web applications have been added to Crowd, along with the Crowd Administration Console and the Demo Application. The new applications are:

Application	Description
OpenID Server	Note: Logically, the OpenID Server is a client application of the Crowd Server, and must be configured as such. The OpenID Server requires a database. By default, a HSQL database is used.
OpenID Demo Client	A simple web application which can be used as a starting point to develop OpenID-enabled Java applications. This application is lightweight. It has no persistence store and does not talk to the Crowd Security Server.

Perform the following steps to finish the upgrade:

1. Create a database to house the data specific to the OpenID Server.
2. Point the application context to the new database. The application context for the OpenID Server is in `atlassian-crowd-1.1.0/apache-tomcat-5.5.20/conf/catalina/localhost/openidserver.xml` More information on how to modify this file for your particular database can be found in Connecting CrowdID to a Database.
3. Update the `atlassian-crowd-1.1.0/crowd-openidserver-webapp/WEB-INF/classes/jdbc.properties` to reflect the dialect of your database.
4. Update `atlassian-crowd-1.1.0/crowd-openidserver-webapp/WEB-INF/classes/crowd.properties` to use a secure password for the OpenID Server application.
5. Add the application via the Crowd Administration Console. The default name of the application is **crowd-openid-server** and the password is whatever you specified in `crowd.properties` in the previous step. For more information on how to add an application, see Adding an Application.
6. Restart the server. This should set up the OpenID Server in Crowd.

Crowd 1.2 Upgrade Notes

Upgrade Procedure

To upgrade to Crowd 1.2.x from 1.1.x or earlier,

- Follow the instructions on upgrading Crowd.

Upgrade Notes

Application Directory Permissions

With Crowd 1.2, directory permissions can now be set at application level. When you upgrade to Crowd 1.2:

- The upgrade procedure will set all application-level permissions equal to your existing directory-level permissions. This means that, for a particular directory, all applications will have the same permissions immediately after the upgrade i.e. the permissions which were set at directory level before the upgrade.
- You can alter the permissions for each application after the upgrade is complete, if you wish.

Developer Notes

SOAP Service API

There are changes to the Crowd API, including new SOAP methods (see CWD-459 and CWD-537), so you should re-generate your WSDL bindings to the Crowd server.

Crowd 1.3 Beta Upgrade Notes

Crowd 1.3 will be launched in early March 2008. A beta release is currently undergoing internal testing. These upgrade notes apply to **Crowd 1.3 beta**. We'll publish the final upgrade notes with the release of Crowd 1.3.0.

Upgrade Procedure

To upgrade to Crowd 1.3.x from 1.2.x or earlier, please follow these upgrade instructions.

Upgrade Notes

Database Configuration

Crowd database configuration is now part of the Setup Wizard. You can choose between a JNDI datasource (i.e. server-managed) or a

JDBC configuration.

If you are using CrowdID with an external database, you will still need to use the manual JNDI datasource configuration method to configure an external database connection.

Database Import

You can now import an XML backup of your Crowd database when upgrading. So you don't have to go through the whole Setup Wizard again, nor do a manual backup and restore of your Crowd database files. Full instructions are in the Upgrade Guide.

Integrated Applications

Crowd's client libraries have been slimmed down to a single JAR file containing all required classes for a Crowd client. (See CWD-767.)

Before upgrading, please remove all previous client libraries (`crowd-XXXX-X.X.X.jar`) from each Crowd-integrated application's `WEB-INF/lib` folder.

Developer Notes

Restructuring of Crowd Client Library

In Crowd 1.3, the Java client library API has been upgraded. This affects applications using the Crowd Client libraries and connectors. Read more about the Client API Changes.

Spring Configuration Upgrade for Crowd Acegi Connector

Applications using the Crowd Acegi connector will need to upgrade their Spring configuration. Refer to the updated documentation for more information.

Crowd 1.3 Upgrade Notes

On this page:

- Upgrade Notes
  - Database Configuration
  - Database Import
  - Integrated Applications
- Developer Notes
  - Restructuring of Crowd Client Library
  - Spring Configuration Upgrade for Crowd Acegi Connector
- Upgrade Procedure

Upgrade Notes

Database Configuration

Crowd database configuration is now part of the Setup Wizard. You can choose between a JNDI datasource (i.e. server-managed) or a JDBC configuration.

If you are using CrowdID with an external database, you will still need to use the manual JNDI datasource configuration method to configure an external database connection.

Database Import

You can now import an XML backup of your Crowd database when upgrading. So you don't have to go through the whole Setup Wizard again, nor do a manual backup and restore of your Crowd database files. Full instructions are in the Upgrade Guide.

Integrated Applications

Crowd's client libraries have been slimmed down to a single JAR file containing all required classes for a Crowd client. (See CWD-767.)

Before upgrading, please remove all previous client libraries (`crowd-XXXX-X.X.X.jar`) from each Crowd-integrated application's `WEB-INF/lib` folder.

Developer Notes

Restructuring of Crowd Client Library

In Crowd 1.3, the Java client library API has been upgraded. This affects applications using the Crowd Client libraries and connectors. Read more about the Client API Changes.

Spring Configuration Upgrade for Crowd Acegi Connector

Applications using the Crowd Acegi connector will need to upgrade their Spring configuration. Refer to the updated documentation for more information.

Upgrade Procedure

To upgrade to Crowd 1.3.x from 1.2.x or earlier, please follow these upgrade instructions.

Crowd 1.4 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 1.4. You can see the features of this release in the Crowd 1.4 Release Notes.

On this page:

- Upgrade Notes
  - Crowd administrators must be in a group mapped to the 'crowd' application
  - Additional file to copy for client applications: crowd-ehcache.xml
  - Additional file to copy for integration with JIRA 3.12.2
- Upgrade Procedure

Upgrade Notes

Crowd administrators must be in a group mapped to the 'crowd' application

With Crowd 1.4 and later, non-administrators as well as Crowd administrators can log in to Crowd. Non-administrators can update their user profiles and view their authorisation details. To support this, the Crowd permissions now distinguish between Crowd administrators (users in groups mapped to the 'crowd' application) and other Crowd users (all users in directories allowed to authenticate to Crowd).

Impact:

- In previous versions of Crowd, any user authorised to log in to the 'crowd' application had access to the full functionality of the Crowd Administration Console. The default setup used the 'crowd-administrators' group to manage these users. Most of our customers will have used the default group or customised groups for their Crowd administrators. But it was possible to grant entire directories administration access to Crowd, by mapping the directory to the 'crowd' application and allowing all to authenticate.
- In Crowd 1.4 and later, every Crowd administrator must be a member of a group mapped to the 'crowd' application (in any mapped directory). Other users will be able to log in to Crowd and use the Self-Service Console if they are members of mapped directories where all can authenticate. But if they are not members of mapped groups, they will not have full access to the Administration Console.

Before upgrading, check that you have a valid administrator
Before starting the upgrade, ensure that there is at least one user in a group that is mapped to the '**crowd**' application.

Additional file to copy for client applications: crowd-ehcache.xml

For better caching, you will need to copy the new `{CROWD_INSTALL}\client\conf\crowd-ehcache.xml` file to each Crowd-integrated application's `WEB-INF/classes/` folder, replacing the existing file.

We have included the above step in the upgrade instructions.

Additional file to copy for integration with JIRA 3.12.2

If you are using **JIRA 3.12.2 or earlier**, you will need to update JIRA's xfire libraries:

- Remove the `xfire-all-1.2.1.jar` file from JIRA's `WEB-INF/lib/` directory.
- Copy the following two files from Crowd's `client/lib/` directory to JIRA's `WEB-INF/lib/` directory:
  - `xfire-aegis-1.2.6.jar`
  - `xfire-core-1.2.6.jar`

Upgrade Procedure

To upgrade to Crowd 1.4.x from 1.3.x or earlier, please follow these upgrade instructions.

Crowd 1.5 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 1.5. You can see the features of this release in the Crowd 1.5 Release Notes.

On this page:

- Upgrade Notes
  - The `crowd.properties` file is now in Crowd Home
  - There are new required JAR files for Crowd WAR deployments
- Upgrade Procedure

Upgrade Notes

The `crowd.properties` file is now in Crowd Home

As from Crowd 1.5, the `crowd.properties` file for the Crowd Administration Console is located in the Crowd Home directory and not the Installation directory. When upgrading from an earlier version of Crowd, you will need to copy the `crowd.properties` file to the root of your Crowd Home directory.

Notes

- The `crowd.properties` file for the CrowdID application is still located in the Installation directory.
- For future upgrades after Crowd 1.5.0, the upgrade process becomes easier because you will no longer need to copy the `crowd.properties` file.

The instructions are incorporated into the Upgrade Guide for Upgrading from Crowd 1.3.0 or Later and Upgrading from Crowd 1.2.x or Earlier .

There are new required JAR files for Crowd WAR deployments

WAR deployments need to ensure that JavaMail classes and the Java Beans Activation Framework are located in the application server's classpath. For more information, please review this guide.

Upgrade Procedure

To upgrade to Crowd 1.5.x from 1.4.x or earlier, please follow these upgrade instructions.

Crowd 1.6 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 1.6. You can see the features of this release in the Crowd 1.6 Release Notes.

On this page:

- Upgrade Notes
- Upgrade Procedure

Upgrade Notes

Crowd 1.6 provides event-based caching updates for some LDAP directories. You may wish to enable it for better performance with client applications such as JIRA. As there are some important limitations to be aware of, please read the documentation before enabling it for your directory.

Upgrade Procedure

To upgrade to Crowd 1.6.x from 1.5.x or earlier, please follow these upgrade instructions.

Crowd 2.0 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 2.0. You can see the features of this release in the Crowd 2.0 Release Notes.

On this page:

- Upgrade Notes
  - Upgrade Procedure Requires New Home Directory and Database XML Export/Import
  - MySQL Database Deployment
  - Improved Search API
  - Backwards-Compatible SOAP API
  - Roles in Crowd now Deprecated
- Upgrade Procedure

Upgrade Notes

Please read the following sections and take action where the note applies to your Crowd installation, before upgrading to the new release of Crowd.

Upgrade Procedure Requires New Home Directory and Database XML Export/Import

With this release, we have redesigned Crowd's database schema. For that reason, you will need to:

- Back up your Crowd database to XML before starting the upgrade.
- Do a clean installation of Crowd, pointing to a new Crowd Home directory.
- Restore your database from the XML backup as part of the setup process.

The full instructions are in our Upgrade Guide.

MySQL Database Deployment

If you are currently using a MySQL database with Crowd, we **strongly** recommend you follow the updated MySQL documentation and use the *READ-COMMITTED* transaction isolation level.

Improved Search API

This point is of interest to developers who have created custom application integrations for Crowd. You can now make use of the performance benefits and other features provided by the new search API. The details are in the JavaDocs.

Backwards-Compatible SOAP API

This point is of interest to developers who have created custom application integrations for Crowd. Even though we have made major changes to the object model in Crowd to improve performance, the SOAP API is still backwards compatible with the previous version.

Roles in Crowd now Deprecated

At present, the implementation of roles in Crowd is identical to the implementation of groups. This design does not provide much useful functionality, so we are planning to redesign the way Crowd supports roles. If you would like to help us to design better role-based access control, please add a comment to the improvement request CWD-931, letting us know how you would like to see it work.

Advance Notice: We recommend that you move away from the use of roles in your Crowd installation, so that you will not be adversely affected by the planned redesign of role functionality. For this reason, roles are now disabled by default when you create a new LDAP directory.

Upgrade Procedure

To upgrade to Crowd 2.0.x from 1.6.x or earlier, please follow these upgrade instructions.

Migrating Crowd between Servers

This guide applies to situations when you may need to migrate Crowd because:

- Your Crowd server is changing.
- You are cloning your production server for a staging, test or development instance.

Preparation

1. Make sure you have a Crowd license for the new server you are targeting. Developer/staging licenses are available for any commercial or academic license. Create a developer license or contact us for help.
2. Add the IP address or hostname of the target Crowd server to the remote addresses in your existing Crowd server:
    - Find the IP address or hostname of the target Crowd server.
    - Log in to the Crowd Administration Console on your existing Crowd server.
    - Click the '**Applications**' tab, find the '***Crowd Console***' application and open the '**Remote Addresses**' tab.
    - If you have '**localhost**' or '**127.0.0.1**' listed there already, you should be fine. If not, add the new IP address or hostname to the list of remote addresses. This list determines the hosts that can access the Crowd Administration Console.
3. Perform an XML backup of your existing Crowd server. Make sure that you check the '**Reset Domain**' checkbox, otherwise you may be prevented from logging in to the new Crowd Administration Console.

From this point on, we will call your existing Crowd server the 'original' server.

Migration

1. Copy the XML backup over to the target server.
2. Install Crowd on the target server using our installation guide.
    - The Crowd version can be the same or higher than the version on the original Crowd server.
    - When specifying your Crowd Home directory, make sure you choose a new location and *not* your original Crowd Home directory.
3. Run the Setup Wizard.
    - When asked for the type of installation, choose '**Import data from an XML backup**'. Provide the full path to your XML backup file and import the data.
    - When given the option of configuring Crowd to target a database, make sure you choose a new one and *not* your original Crowd database.
4. When the import finishes, shut down Crowd.
5. Locate the `crowd.properties` file in the target server's Crowd Home directory. (This file will have been generated from the data in the XML backup.) Edit the file and modify the line `crowd.server.url` so that it points to your new Crowd server.

Post Migration Verification

1. In your original Crowd server, you can now remove the IP address or hostname you added during the preparation steps. This will help prevent you from accidentally logging into your original Crowd server.
2. Start Crowd on the new server. You should be able to authenticate and access Crowd using the same credentials as on your original Crowd server.

Applications and Customisations

1. For any application you are going to test against this new Crowd server, you will need to modify the application's `crowd.properties` file to point to this new server.
2. If you have installed any Crowd plugins or added other customisations, you will need to re-apply them on the new server.

If you encounter any difficulties, please feel free to contact support and let us know which step you are having problems with.

RELATED TOPICS

Specifying your Crowd Home Directory
Crowd Installation & Upgrade Guide
Running the Setup Wizard

CrowdID User Guide

With Crowd comes **CrowdID**, your OpenID provider.

CrowdID is an Atlassian product which allows you to use a single login for all OpenID-enabled websites.

This means that you don't have to remember a separate username and password for each different site that you visit. You can just use your OpenID for all of them.

You can use CrowdID if your administrator has installed it for your organisation. For instructions on setting up CrowdID, please see the *CrowdID Administration Guide*.

The *CrowdID User Guide* tells you how to

- Log in to websites using CrowdID.
- Instruct CrowdID to always allow login to a specific site.
- Set up your own profile(s) within CrowdID.
- Use CrowdID to change your password.

Contents of the CrowdID User Guide

- 1. Getting started with CrowdID
  - 1.1 What is OpenID?
  - 1.2 What is CrowdID?
  - 1.3 What is an OpenID URL or identifier?
  - 1.4 Viewing the CrowdID page
- 2. Logging in to a website using OpenID
  - 2.1 Does the website support OpenID?
  - 2.2 Entering your OpenID URL
  - 2.3 Logging in to CrowdID
  - 2.4 Allowing or denying a login
  - 2.5 Providing additional profile information to a website
- 3. Viewing your always-approved websites
- 4. Viewing your login history
- 5. Updating your profile
- 6. Using more than one profile
  - 6.1 Adding a profile
  - 6.2 Choosing a profile for a website
  - 6.3 Setting a default profile
  - 6.4 Deleting a profile
- 7. Changing or resetting your password
  - 7.1 Changing your password
  - 7.2 Resetting your password

1. Getting started with CrowdID

CrowdID is an Atlassian product which allows you to use a single login for all OpenID-enabled websites.

This means that you don't have to remember a separate username and password for each different site that you visit. You can just use your

OpenID for all of them.

You can use CrowdID if your administrator has installed it for your organisation.

- 1.1 What is OpenID?
- 1.2 What is CrowdID?
- 1.3 What is an OpenID URL or identifier?
- 1.4 Viewing the CrowdID page

1.1 What is OpenID?

The term '**OpenID**' has two meanings:

- The OpenID protocol, described below.
- Your own identifier or URL.

OpenID is an open, free protocol which allows you to use a single identifier to login to any OpenID-enabled website. OpenID allows the website to communicate with your OpenID provider (e.g. your organisation's CrowdID server) when attempting to verify your login.

Do you have a zillion usernames and passwords, which you use for logging in to blogs and websites all over the place? **OpenID** allows you to throw them all away, for all websites that support it. More and more sites are coming on board.

RELATED TOPICS

- 1.1 What is OpenID?
- 1.2 What is CrowdID?
- 1.3 What is an OpenID URL or identifier?
- 1.4 Viewing the CrowdID page

CrowdID User Guide

1.2 What is CrowdID?

CrowdID is an Atlassian product which makes use of the OpenID protocol to allow you to use a single login for a number of websites. To put it another way: CrowdID is an '**OpenID provider**'. You can use CrowdID if your administrator has installed it for your organisation.

This means that you can:

- Securely store your username and password on your organisation's server.
- Use your OpenID as a single identifier to log in to all websites which support OpenID.
- Control how you allow or deny login requests from websites.

Your organisation can use **CrowdID** to set up an internal OpenID provider. There are also other OpenID providers, where you can get a free OpenID.

RELATED TOPICS

- 1.1 What is OpenID?
- 1.2 What is CrowdID?
- 1.3 What is an OpenID URL or identifier?
- 1.4 Viewing the CrowdID page

CrowdID User Guide

1.3 What is an OpenID URL or identifier?

To log in to an OpenID-enabled website you need an OpenID identifier, also called an OpenID URL or simply an OpenID. Your OpenID is a URL (web address) which points to your organisation's CrowdID server. Here are some examples of what your OpenID may look like:

```
http://my.server.name/myname
http://myname.mysite.com
```

To find your OpenID URL, you can:

- Ask your system administrator, or
- Click the 'My OpenID' link on the 'Home' tab of the CrowdID page.

RELATED TOPICS

- 1.1 What is OpenID?
- 1.2 What is CrowdID?
- 1.3 What is an OpenID URL or identifier?
- 1.4 Viewing the CrowdID page

CrowdID User Guide

1.4 Viewing the CrowdID page

The **CrowdID** page allows you to:

- View your OpenID URL.
- Set up your profile(s).
- View your list of always-approved sites.
- View your login history.
- Resume approval of a login. (This option appears only during a login process, if you move away from the 'OpenID Verification' page.)
- Change your password.

There are two ways to access the CrowdID page:

- While you are logging in to another site.
- Directly via the CrowdID URL.

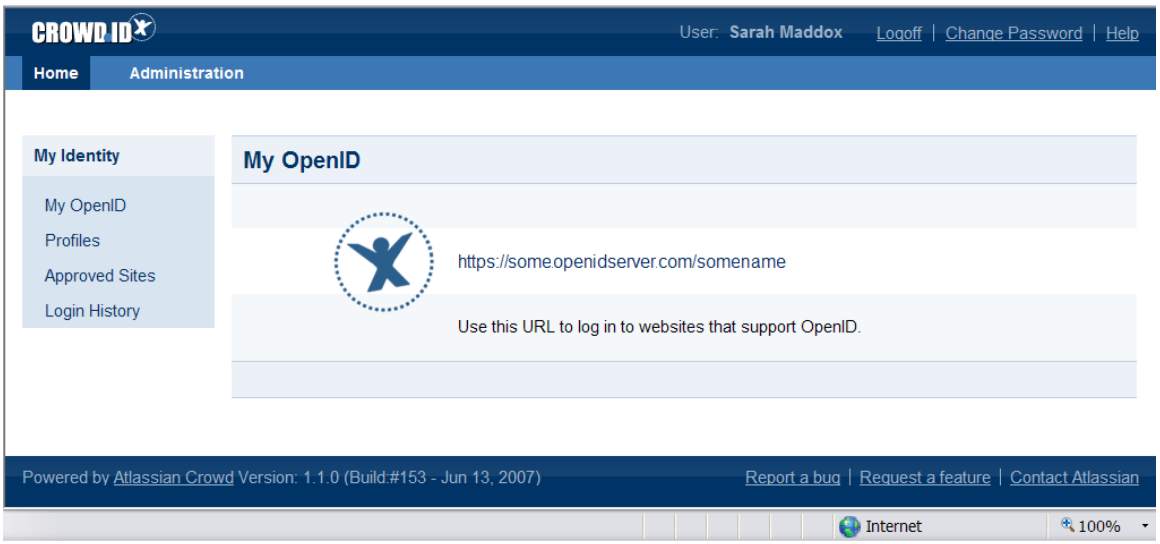
To access the CrowdID page while you are logging in to another site,

1. Use your OpenID to log in to the website you want to visit.
2. Log in to CrowdID if prompted.
3. The CrowdID **'OpenID Verification'** page will appear, provided that you have not previously added the website to your list of always-approved sites. You can choose any of the CrowdID options on the left-hand navigation panel, even during the login process.
4. When you have finished your tasks in CrowdID, you can resume the login.

To access CrowdID directly via the CrowdID URL,

1. Ask your administrator for the CrowdID address (URL) as configured for your organisation.
2. Type or paste the address into the address or navigation bar of your internet browser.
3. The CrowdID Login page will appear. Type in your username and password.
4. Click the '**Login**' button.
5. The CrowdID **'My OpenID'** page will appear. The CrowdID options are displayed in the left-hand navigation panel and top menu bar.

Screenshot: CrowdID My OpenID page



RELATED TOPICS

- 1.1 What is OpenID?
- 1.2 What is CrowdID?
- 1.3 What is an OpenID URL or identifier?
- 1.4 Viewing the CrowdID page

CrowdID User Guide

2. Logging in to a website using OpenID

CrowdID enables you to log in to a website using your OpenID. The login process depends upon the following:

- Have you logged in to CrowdID already during this browser session?
- Have you previously added the website to your list of always-approved sites?
- Does the website you are visiting require additional profile information?

Steps in the login process:

1. Find the OpenID login page or section on the website you want to visit.
2. Enter your OpenID and click the login button.
3. If prompted, log in to CrowdID. (Required if you have not already logged in during this browser session.)
4. If prompted, instruct CrowdID to allow the website login. (Required if you have not previously added the website to your list of always-approved sites.)
5. If prompted, supply additional profile information. (Required if the website you are visiting wants more information.)

The login process can be very simple: just the first two steps above, provided that you have already logged in to CrowdID this session and have already added the website to your list of always-approved sites.

2.1 Does the website support OpenID?

You can only use your OpenID (also called an OpenID URL or identifier) to log in to a website if the site supports the OpenID protocol. The number of websites that support OpenID is growing rapidly.

To see if a particular website supports OpenID, check the site's login page for one or more of the following:

- The word 'OpenID'.
- The OpenID logo

RELATED TOPICS

- 2.1 Does the website support OpenID?
- 2.2 Entering your OpenID URL
- 2.3 Logging in to CrowdID

- 2.4 Allowing or denying a login
- 2.5 Providing additional profile information to a website

CrowdID User Guide

2.2 Entering your OpenID URL

With CrowdID, you can use your '**OpenID**' (also called an OpenID URL or identifier) to log in to a website that supports the OpenID protocol.

To log in to a website which supports OpenID,

1. Go to the login page of the website you want to visit.
2. Look for the OpenID login section.
   Sometimes the OpenID login will be on the same page as the standard login. Other sites will have a separate OpenID login page.
3. Type or paste your OpenID into the login text box.
   Usually, you must enter the full OpenID. In some sites, you can enter the OpenID without 'http://'
4. Click the login button. The button will probably be labelled 'Log in', 'Sign in' or 'Go'.

One of the following things will happen now:

- If you have not already logged in to CrowdID during this browser session, you will see the CrowdID login page.
- If you have already logged in to CrowdID and you have previously instructed CrowdID to allow this website always, then you will be logged straight into the website.
- If you have already logged in to CrowdID but have not previously set this site to "Allow Always", then CrowdID will ask you to approve the login.
- If your administrator has blocked access to this website, CrowdID will display an 'OpenID Verification Error' message.

RELATED TOPICS

- 2.1 Does the website support OpenID?
- 2.2 Entering your OpenID URL
- 2.3 Logging in to CrowdID
- 2.4 Allowing or denying a login
- 2.5 Providing additional profile information to a website

CrowdID User Guide

2.3 Logging in to CrowdID

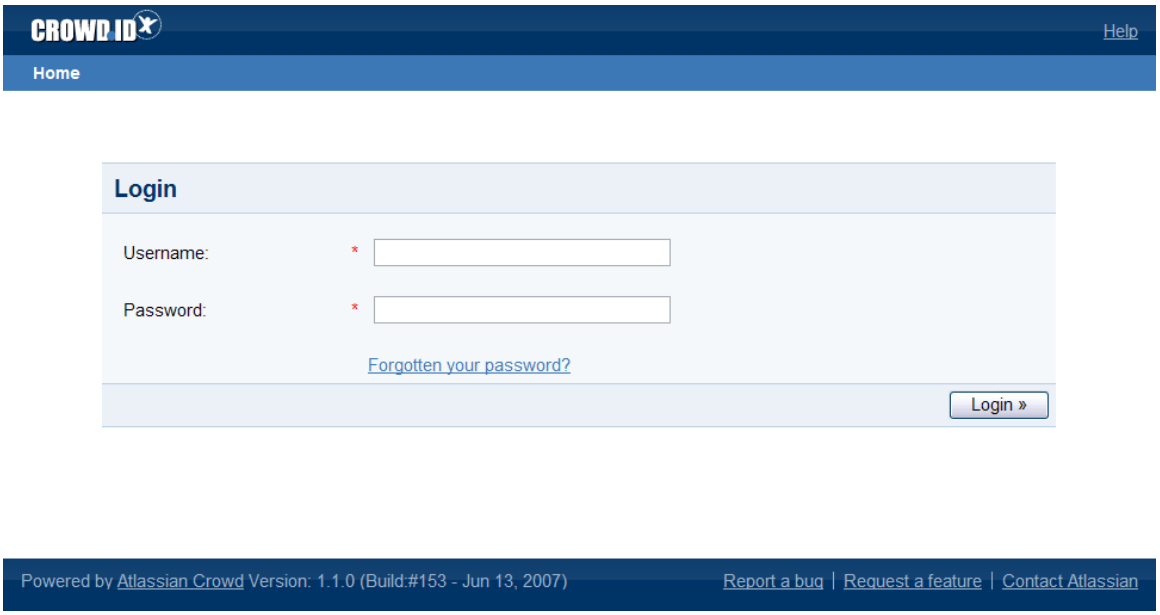
CrowdID will ask you to log in, if you have not already done so during this browser session or if your session has timed out. The CrowdID login may appear during the process of logging in to another website, or when you are accessing CrowdID directly.

To log in to CrowdID,

1. Type in your username and password.
2. Click the '**Login**' button.

You can reset your password, if you have forgotten it.

Screenshot: CrowdID login page



If you are in the process of logging in to another web site, CrowdID will now ask you to approve the login.

RELATED TOPICS

- 2.1 Does the website support OpenID?
- 2.2 Entering your OpenID URL
- 2.3 Logging in to CrowdID
- 2.4 Allowing or denying a login
- 2.5 Providing additional profile information to a website

CrowdID User Guide

2.4 Allowing or denying a login

When you use your OpenID to log in to a website, CrowdID will present the **'OpenID Verification'** page where you can allow or deny the login.

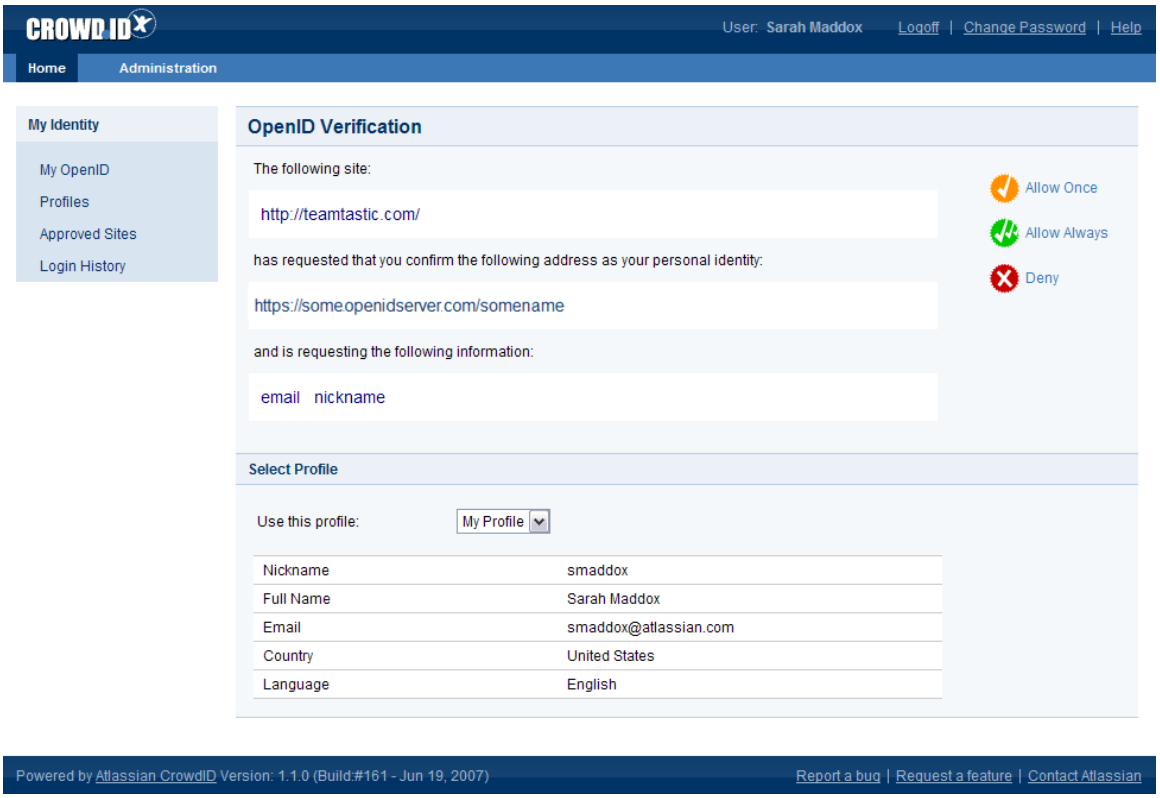
> If you have previously instructed CrowdID to allow this site always, you will not see this page. You can remove a site from the 'Allow Always' list in CrowdID.

You can instruct CrowdID to:

- Allow the login for this session only (**'Allow Once'**).
- Allow login to this site every time you use your OpenID (**'Allow Always'**).
- Refuse login to this site (**'Deny'**).
- Use a specific profile.

If you move away from the 'OpenID Verification' page within CrowdID, you can go back to the page and resume approval.

Screenshot: OpenID Verification page



To allow the login for this session only,

1. Click '**Allow Once**' on the right of the CrowdID 'OpenID Verification' page.
2. CrowdID will send you back to the original site, passing your profile information as well as the confirmed login. The website you are visiting may ask you to complete your profile information.

To allow login to this site every time you use your OpenID,

1. Click '**Allow Always**' on the right of the CrowdID 'OpenID Verification' page.
2. CrowdID will add the website to your list of approved sites and send you back to the original site, passing your profile information as well as the confirmed login. The website you are visiting may ask you to complete your profile information.

To refuse login to this site,

1. Click '**Deny**' on the right of the CrowdID 'OpenID Verification' page.
2. CrowdID will send you back to the original site and refuse the login. The original site will probably show a message something like 'Verification cancelled'.

To use a specific profile,

1. If you have defined more than one profile, you can choose a specific profile for the website you are visiting. Select a profile from the dropdown list labelled '**Use this profile**' on the CrowdID 'OpenID Verification' page.
2. The profile details will change in the 'Select Profile' section of the page. CrowdID will pass these profile details to the website when you allow the login.

To go back to the 'OpenID Verification' page and resume approval,

1. Click '**Resume Approval**' in the left-hand navigation panel.
   This option will appear if you move away from the 'OpenID Verification' page during the login process.
2. CrowdID will return to the '**OpenID Verification**' page, where you can allow the login.

RELATED TOPICS

- 2.1 Does the website support OpenID?
- 2.2 Entering your OpenID URL
- 2.3 Logging in to CrowdID
- 2.4 Allowing or denying a login
- 2.5 Providing additional profile information to a website

CrowdID User Guide

2.5 Providing additional profile information to a website

When you log in to a website using your OpenID, CrowdID passes your profile information to the website. Some websites will then log you in immediately, while other websites may ask you to confirm or complete the profile information.

You are now outside CrowdID. Any dialogue here is between you and the website you are visiting.

To provide additional profile information to a website,

1. Check the profile information displayed, and add extra information as you wish.
2. Click the button or other option supplied by the website to complete the login process.

You can change your profile information and define more than one profile in CrowdID.

RELATED TOPICS

- 2.1 Does the website support OpenID?
- 2.2 Entering your OpenID URL
- 2.3 Logging in to CrowdID
- 2.4 Allowing or denying a login
- 2.5 Providing additional profile information to a website

CrowdID User Guide

3. Viewing your always-approved websites

When logging in to a website, you can instruct CrowdID to allow login to the site every time you use your OpenID (**'Allow Always'**).

The CrowdID '**Approved Sites**' page allows you to:

- View your list of always-approved sites.
- Remove a site from the list.
- Choose a profile for use when logging in to a site.

- If you have never instructed CrowdID to 'Allow Always' for any sites, The 'Approved Sites' page will display a message like '**You currently have no approved sites.**'
- You can add profiles on the CrowdID 'Profiles' page.

To view your list of always-approved sites,

1. Access CrowdID.
2. Click '**Approved Sites**' in the left-hand navigation panel.

To remove a site from the list,

1. Access CrowdID.
2. Click '**Approved Sites**' in the left-hand navigation panel.
3. Your list of always-approved sites will appear. Click the remove button next to the site which you want to remove.
4. Click the '**Apply**' button.
5. '**Update Successful**' message is displayed.

If you do not click the 'Apply' button, your changes will be cancelled.

To choose a profile for use when logging in to a site,

1. Access CrowdID.
2. Click '**Approved Sites**' in the left-hand navigation panel.
3. Your list of always-approved sites will appear. Select the profile you want from the dropdown list next to the applicable site.
4. Click the '**Apply**' button.
5. '**Update Successful**' message is displayed.

If you do not click the 'Apply' button, your changes will be cancelled.

Screenshot: CrowdID Approved Sites page

RELATED TOPICS

- 1. Getting started with CrowdID
- 2. Logging in to a website using OpenID
- 3. Viewing your always-approved websites
- 4. Viewing your login history
- 5. Updating your profile
- 6. Using more than one profile
- 7. Changing or resetting your password

4. Viewing your login history

The CrowdID '**Login History**' page displays a list of the sites you have visited and the type of approval you gave on each visit:

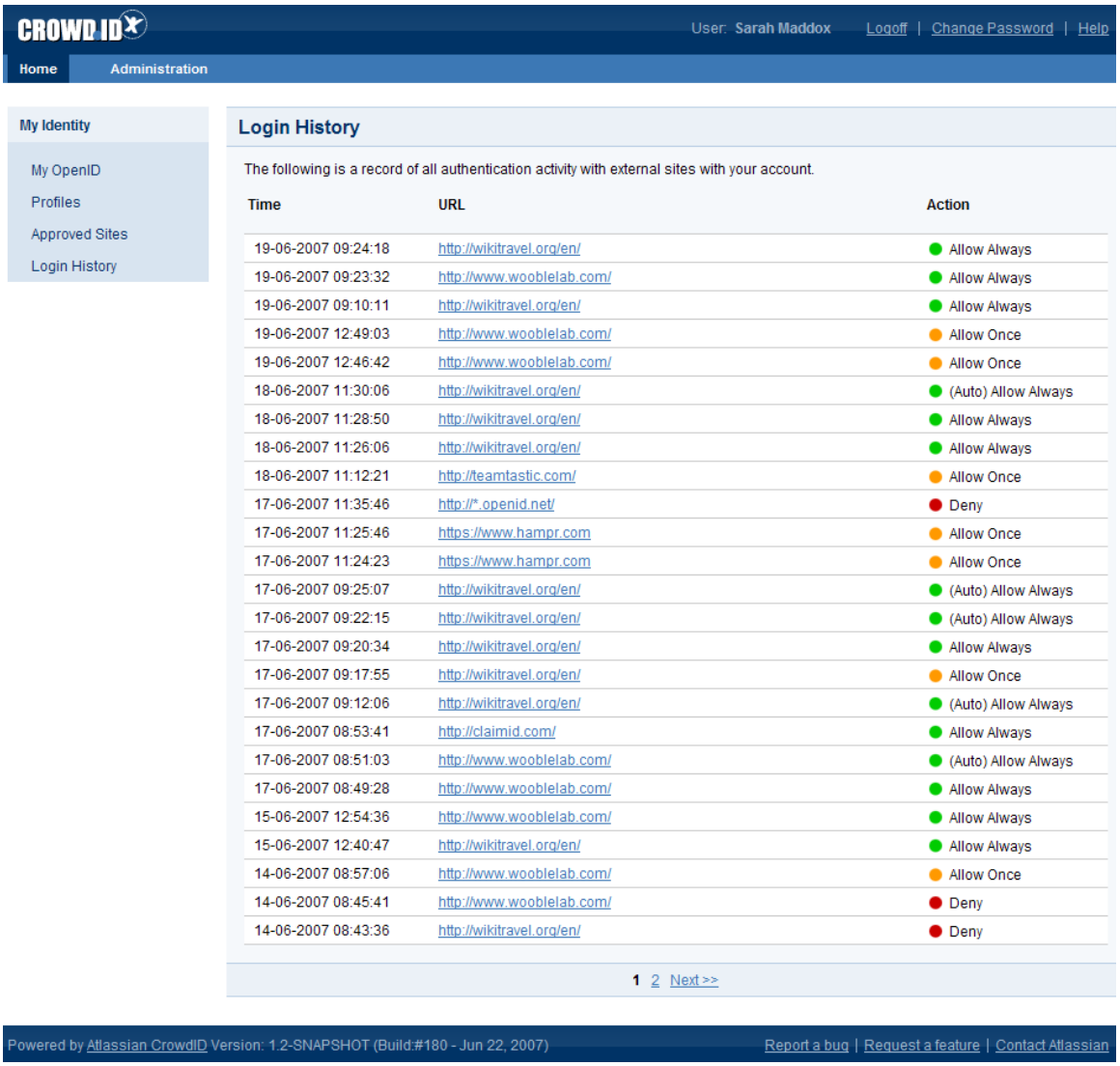
- '**Allow Always**' - At the time of this login, you instructed CrowdID to allow login to the site every time you use your OpenID.
- '**(Auto) Allow Always**' - This login was allowed automatically, because you have previously instructed CrowdID to allow login to the site every time you use your OpenID.
- '**Allow Once**' - You instructed CrowdID to allow login to the site at that time only.
- '**Deny**' - You instructed CrowdID to refuse the login to the site at that time.

To view your login history,

1. Access CrowdID.
2. Click '**Login History**' in the left-hand navigation panel.

If you have used your OpenID many times, the login history items will be shown on more than one page. To move from one page to another, click the page numbers or the '**Next**' and '**Prev**' links at the bottom of the page.

Screenshot: CrowdID Login History page



RELATED TOPICS

- 1. Getting started with CrowdID
- 2. Logging in to a website using OpenID
- 3. Viewing your always-approved websites
- 4. Viewing your login history
- 5. Updating your profile
- 6. Using more than one profile
- 7. Changing or resetting your password

5. Updating your profile

When you log in to a website using your OpenID, CrowdID will pass some information to the website. The information is copied from your profile on CrowdID. When your profile is first created, CrowdID will auto-fill the information where possible, by copying:

- Country and language from the language information in your browser.

- Name and email address from your organisation's user directory.

You can update your profile information on CrowdID, as described below.

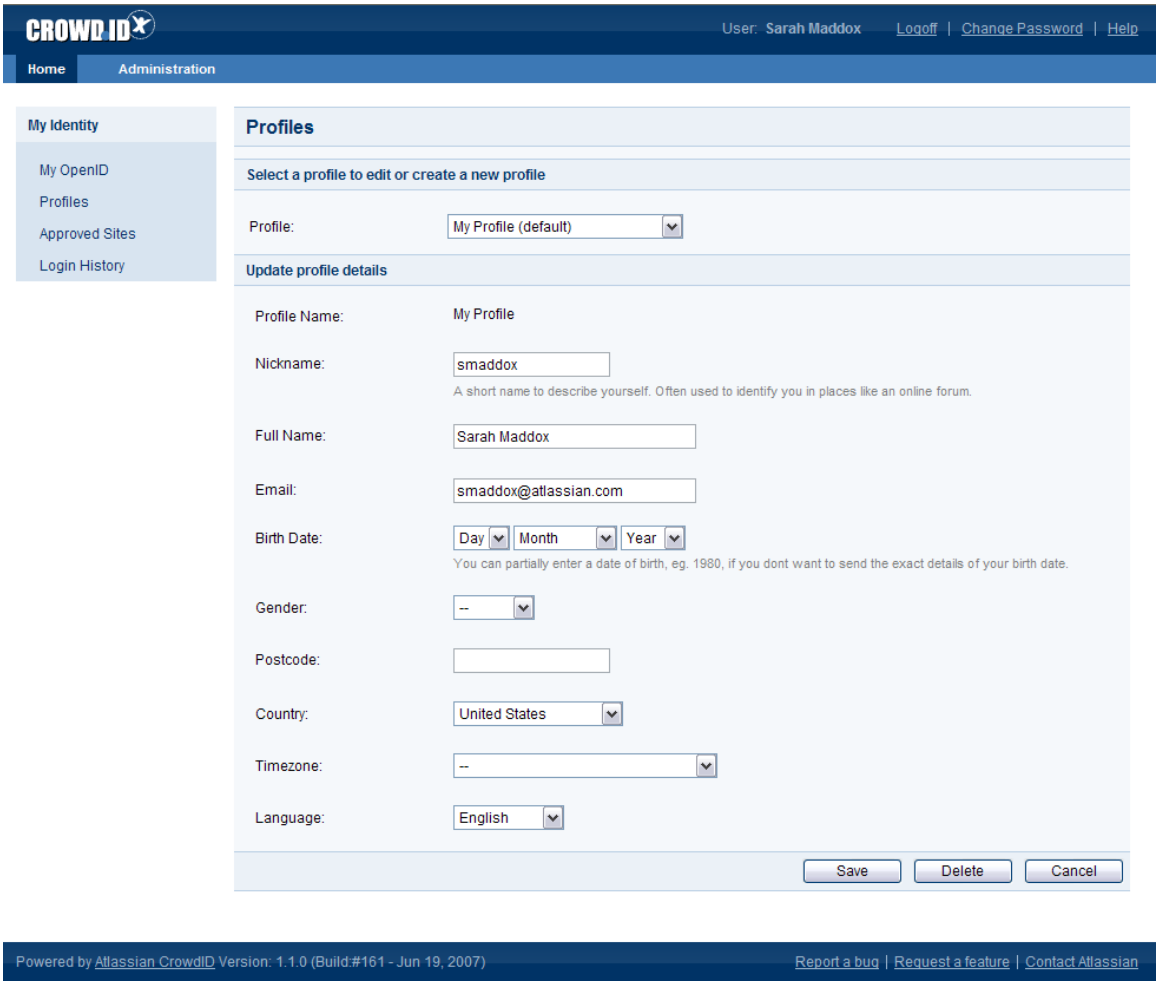
You can also:

- Add a new profile.
- Choose a profile for a website.
- Set a profile as default.
- Delete a profile.

To update your profile,

1. Access CrowdID.
2. Click '**Profiles**' in the left-hand navigation panel.
3. Select the required profile from the '**Profile**' dropdown list, if you have more than one profile.
4. Update the profile details then click the '**Save**' button.
5. '**Profile updated**' message is displayed at the top of the page.

Screenshot: CrowdID Profiles page



RELATED TOPICS

- 1. Getting started with CrowdID
- 2. Logging in to a website using OpenID
- 3. Viewing your always-approved websites
- 4. Viewing your login history
- 5. Updating your profile
- 6. Using more than one profile
- 7. Changing or resetting your password

6. Using more than one profile

You can create multiple profiles in CrowdID and then allocate specific profiles to specific websites.

- 6.1 Adding a profile
- 6.2 Choosing a profile for a website
- 6.3 Setting a default profile
- 6.4 Deleting a profile

6.1 Adding a profile

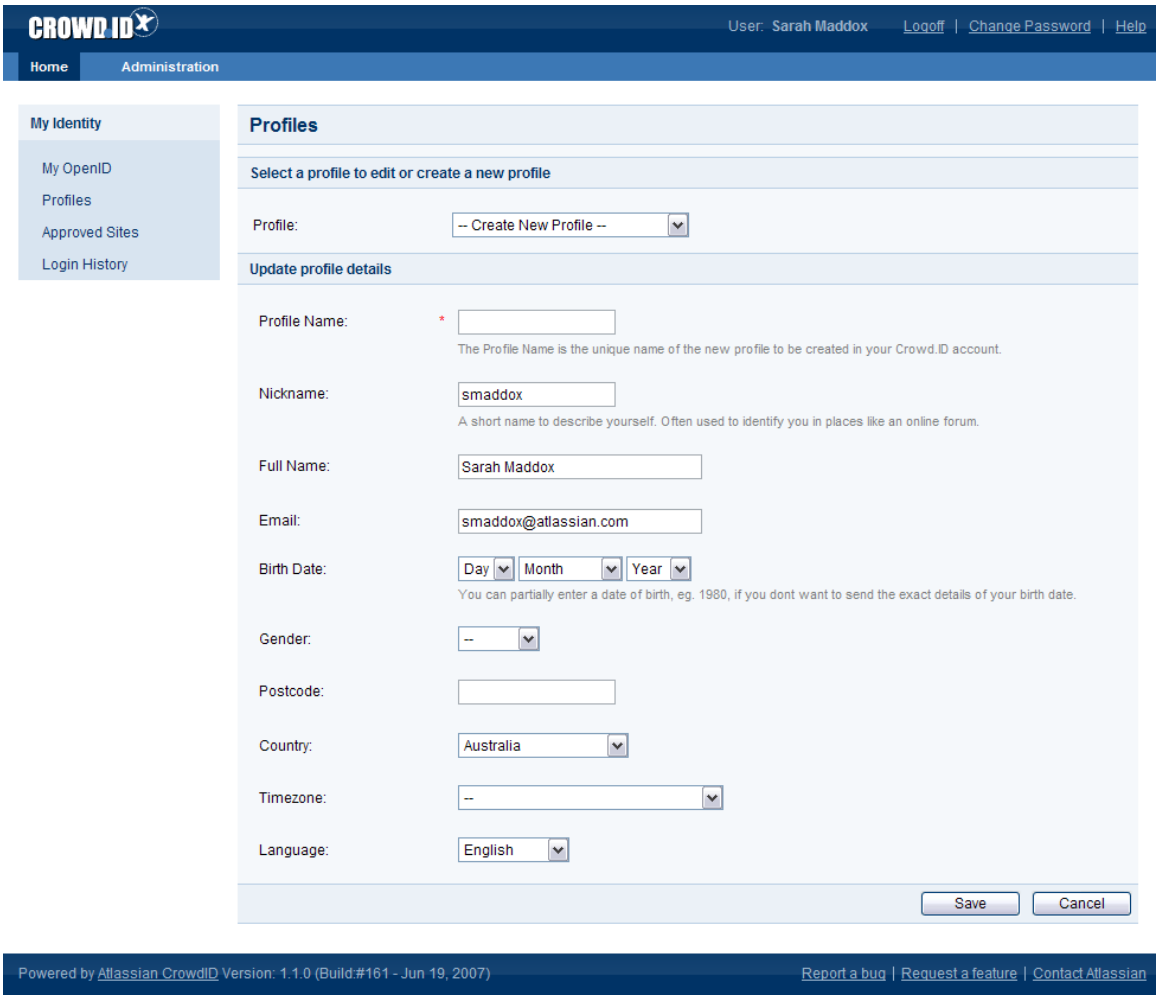
When you log in to a website using your OpenID, CrowdID will pass some information to the website. The information is copied from your profile on CrowdID. When your profile is first created, CrowdID will auto-fill the information where possible, by copying:

- Country and language from the language information in your browser.
- Name and email address from your organisation's user directory.

To add a profile,

1. Access CrowdID.
2. Click '**Profiles**' in the left-hand navigation panel.
3. Select '**-- Create New Profile --**' from the '**Profile**' dropdown list.
4. CrowdID will auto-fill the information where possible. Update the profile details then click the '**Save**' button.
5. '**Profile updated**' message is displayed at the top of the page.

Screenshot: CrowdID adding a profile



RELATED TOPICS

- 6.1 Adding a profile
- 6.2 Choosing a profile for a website
- 6.3 Setting a default profile
- 6.4 Deleting a profile

CrowdID User Guide

6.2 Choosing a profile for a website

You can choose a specific profile for use when logging in to a website. There are different ways to choose a profile:

- Choose a profile for a specific login, during the login process. You can do this for sites which you have not set to 'Allow Always'.
- Choose a profile for a specific website, on the CrowdID 'Approved Sites' page. You can do this for sites which you have set to 'Allow Always'.
- Set your default profile on the CrowdID '**Profiles**' page.

RELATED TOPICS

- 6.1 Adding a profile
- 6.2 Choosing a profile for a website
- 6.3 Setting a default profile
- 6.4 Deleting a profile

CrowdID User Guide

6.3 Setting a default profile

If you have more than one profile, you can choose one of them as default.

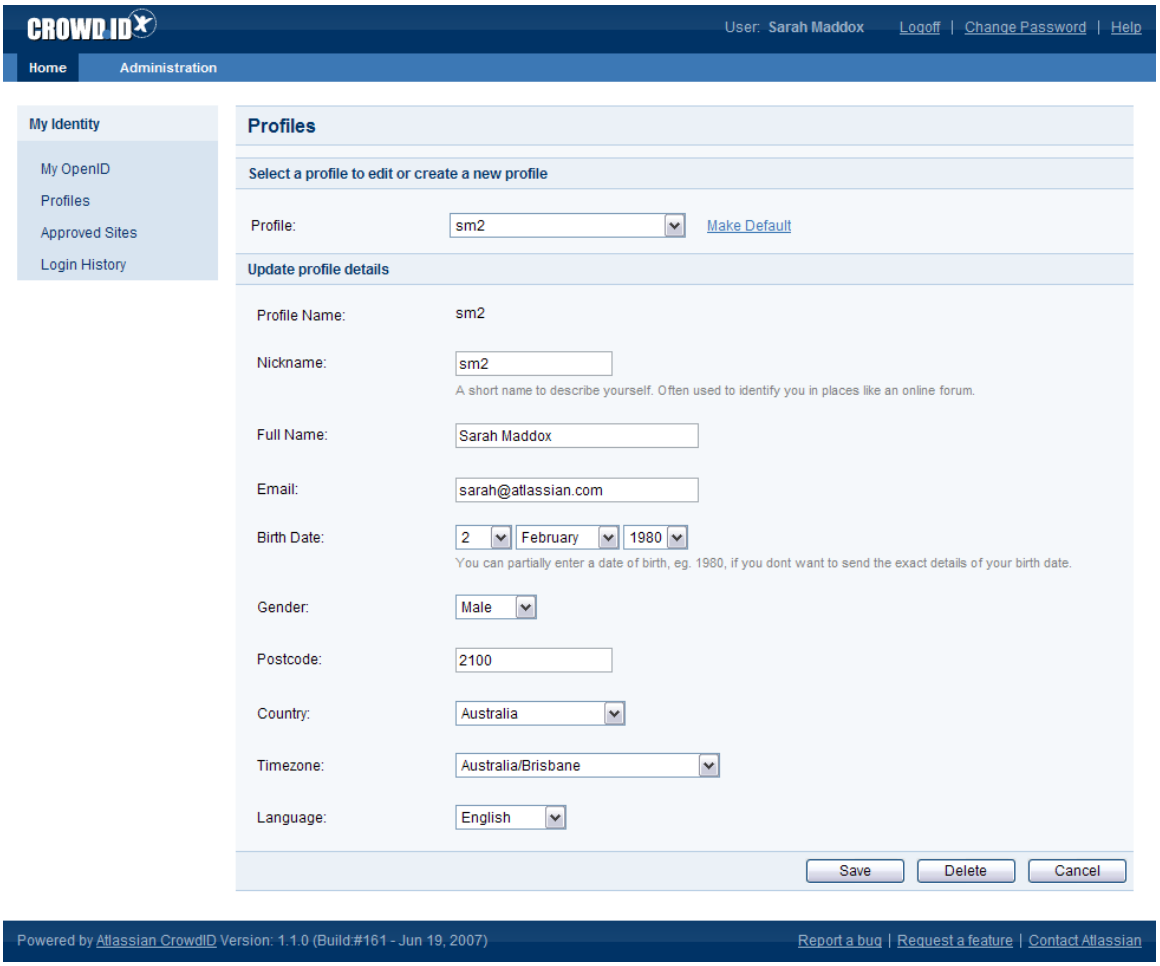
Effect of the 'default' profile when you are logging in to a website:

- If you have never logged in to the website before or have previously allowed or denied authentication to that site, the default profile will be pre-selected. You can still choose a different profile during the login.
- If you have set the website to 'Always Allow', CrowdID will use the profile selected for the site on the Approved Sites page.

To set a default profile,

1. Access CrowdID.
2. Click '**Profiles**' in the left-hand navigation panel.
3. Select the required profile in the '**Profile**' dropdown list
4. Click the '**Make Default**' link next to the 'Profile' dropdown list.
   The 'Make Default' link does not appear if the selected profile is already the default.
5. The word '**(default)**' appears next to the profile name in the dropdown list.

Screenshot: CrowdID setting a default profile



RELATED TOPICS

- 6.1 Adding a profile
- 6.2 Choosing a profile for a website
- 6.3 Setting a default profile
- 6.4 Deleting a profile

CrowdID User Guide

6.4 Deleting a profile

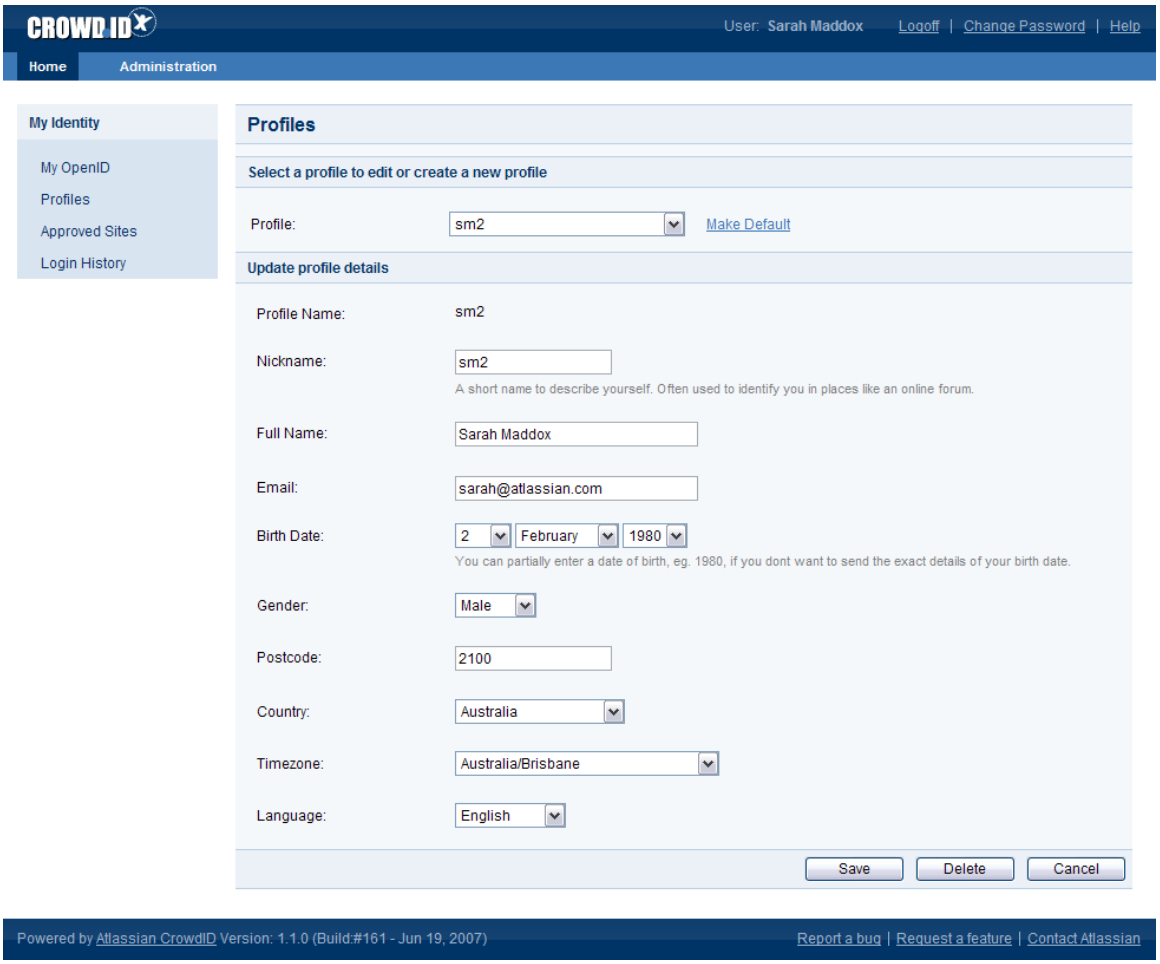
You can delete one of your profiles on CrowdID, provided that it is not your default profile.

To delete a profile,

1. Access CrowdID.
2. Click '**Profiles**' in the left-hand navigation panel.
3. Select the required profile in the '**Profile**' dropdown list
4. Click the '**Delete**' button.
5. '**Profile deleted**' message is displayed at the top of the page.

If you delete a profile which is linked to one or more of your always-approved websites, CrowdID will remove the affected website(s) from the list.

Screenshot: CrowdID profiles page



RELATED TOPICS

- 6.1 Adding a profile
- 6.2 Choosing a profile for a website
- 6.3 Setting a default profile
- 6.4 Deleting a profile

CrowdID User Guide

7. Changing or resetting your password

If your administrator has allowed it, you can use CrowdID to change your password across all Crowd applications. Note that you will need to be logged in to Crowd before you can do this.

When attempting to log in to Crowd, you can also reset your password, if you have forgotten the old one. A new password will be emailed to you.

7.1 Changing your password

The CrowdID '**Change Your Password**' page allows you to change your password across all applications in your organisation, provided that the application is linked to **Crowd**.

Note:

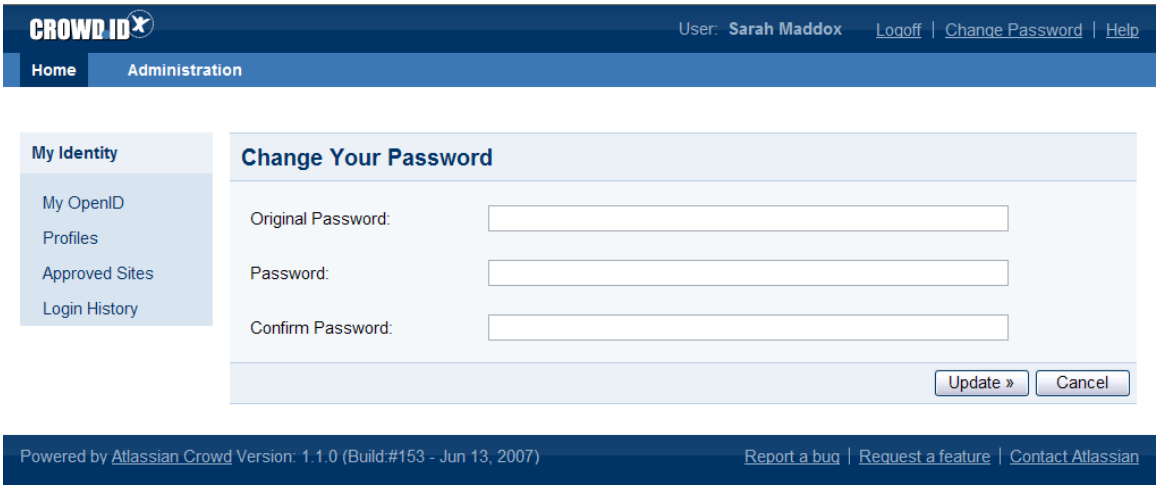
- Crowd will attempt to change your password in all the user directories linked to Crowd. This will be successful where the directory allows it.
- Your administrator may disable password-change via CrowdID. In that case, you will receive an error message when you apply the change.

To change your password,

1. Access CrowdID.
2. Click '**Change Password**' in the top menu bar.
3. The '**Change Your Password**' page will appear. Type in your old password once, and the new password twice.
4. Click the '**Update**' button.
5. '**Password updated**' message is displayed.

If the change is successful, your password may also have changed in other Crowd-connected applications.

Screenshot: CrowdID Change Your Password page



RELATED TOPICS

- 1. Getting started with CrowdID
- 2. Logging in to a website using OpenID
- 3. Viewing your always-approved websites
- 4. Viewing your login history
- 5. Updating your profile
- 6. Using more than one profile
- 7. Changing or resetting your password

7.2 Resetting your password

The CrowdID '**Login**' page allows you to reset your password, which is useful when you have forgotten the password.

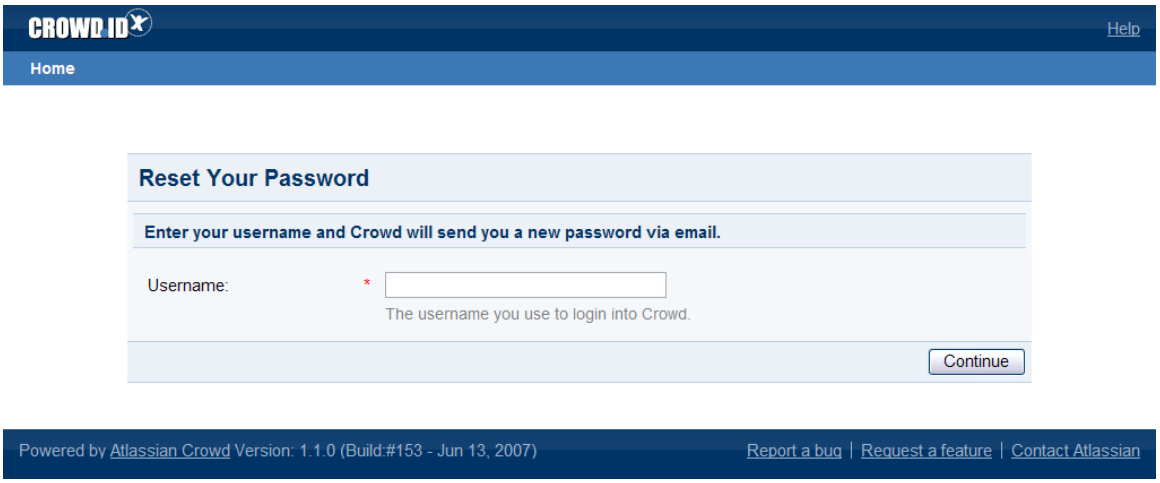
This will reset your password across all applications in your organisation, provided that the application is linked to **Crowd**.

To reset your password,

1. Access CrowdID.
2. Click the '**Forgotten your password?**' link on the CrowdID Login page.
3. The '**Reset Your Password**' page will appear. Type in your Crowd username and click the '**Continue**' button.
4. A message will appear: '**Your new password is on the way!**'. Click the '**Home**' link at the top of the page.
5. You will receive an email message with your new password. Copy the password.
6. Log in to CrowdID using the new password.
7. Change your password to one you can remember easily.

If the change is successful, your password may also have changed in other Crowd-connected applications.

Screenshot: CrowdID Reset Your Password page



RELATED TOPICS

- 1. Getting started with CrowdID
- 2. Logging in to a website using OpenID
- 3. Viewing your always-approved websites
- 4. Viewing your login history
- 5. Updating your profile
- 6. Using more than one profile
- 7. Changing or resetting your password

CrowdID Administration Guide

CrowdID is a free add-on that ships with Crowd versions 1.1 and later. It gives administrators a secure way to provide OpenID accounts for their users.
The *CrowdID Administration Guide* is for people who have CrowdID administration rights. For instructions on using CrowdID to access OpenID-enabled websites, please see the *CrowdID User Guide*.

Table of Contents

- 1. About CrowdID
  - 1.1 How CrowdID works with Crowd
    - 1.1.1 Determining the name of the CrowdID application
    - 1.1.2 Locating the Crowd Server that CrowdID is using
  - 1.1 How OpenID sites interact with CrowdID
- 2. Allowing users to access CrowdID
  - 2.1 Granting CrowdID access rights to a user
  - 2.2 Granting CrowdID Administration Rights to a User
- 3. Specifying the sites to which users can login
  - 3.1 Allowing all hosts
  - 3.2 Allowing all except specified hosts ('Blacklist')
  - 3.3 Allowing specified hosts only ('Whitelist')
- 4. Configuring CrowdID system settings
  - 4.1 Specifying the CrowdID URL
  - 4.2 Enabling localhost authentication
  - 4.3 Enabling immediate authentication requests
  - 4.4 Enabling communication with stateless clients

1. About CrowdID

CrowdID is a free add-on that ships with Crowd versions 1.1 and later. It gives administrators a secure way to provide [OpenID|http://openid.net/] accounts for their users.
Crowd is a middleware application that connects web applications (such as CrowdID, JIRA and Confluence) to specified directories (e.g. Microsoft Active Directory, OpenLDAP). For details please see Concepts in the *Crowd Administration Guide*.

- 1.1 How CrowdID works with Crowd
  - 1.1.1 Determining the name of the CrowdID application
  - 1.1.2 Locating the Crowd Server that CrowdID is using
- 1.1 How OpenID sites interact with CrowdID

To access CrowdID, go to http://localhost:8095/openidserver.

1.1 How CrowdID works with Crowd

CrowdID is a free add-on that ships with Crowd versions 1.1 and later. It gives administrators a secure way to provide OpenID accounts for their users.
Crowd is a middleware application that connects web applications (such as CrowdID, JIRA and Confluence) to specified directories (e.g. Microsoft Active Directory, OpenLDAP). For details please see Concepts in the *Crowd Administration Guide*.

This means that:

- CrowdID is a Crowd-connected application.
- CrowdID users are authenticated against Crowd-connected directories.
- If a user has already logged into any other Crowd-connected application (and single sign-on is enabled), they will not be prompted for any further login once they have entered their OpenID URL at an OpenID-enabled website.
- Multiple CrowdID instances can use one Crowd instance. Large organisations often find this useful.

CrowdID is automatically installed when you install Crowd. When you start Crowd for the first time and run the Setup Wizard, you will be offered the option of configuring CrowdID. If you choose not to setup CrowdID at that time, you can always set it up later as described in 4. Configuring CrowdID system settings. Note that you will also need to define the CrowdID application in Crowd, and map it to an appropriate directory — for details please see the *Crowd Administration Guide*.

To access CrowdID, go to http://localhost:8095/openidserver.

RELATED TOPICS

- 1.1 How CrowdID works with Crowd
  - 1.1.1 Determining the name of the CrowdID application
  - 1.1.2 Locating the Crowd Server that CrowdID is using
- 1.1 How OpenID sites interact with CrowdID

Crowd Documentation

1.1.1 Determining the name of the CrowdID application

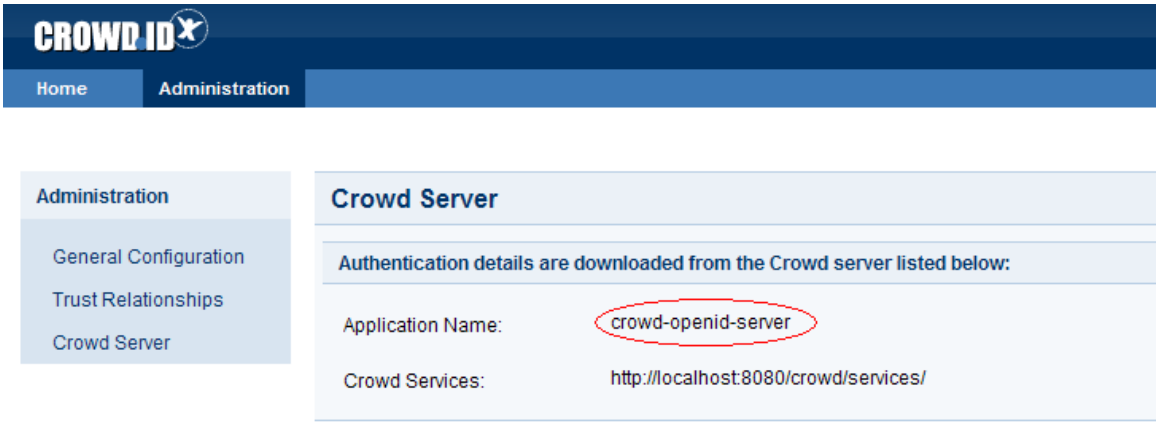
CrowdID is a Crowd-connected application (for more information please see Managing Applications in the *Crowd Administration Guide*).

To change the details or users of your CrowdID application within Crowd, you will need to know the name by which your Crowd application is defined in your Crowd server.

To see the name of your CrowdID application,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**Crowd Server**' link in the left navigation column.
4. This will display the '**Crowd Server**' details.
   The '**Application Name**' field contains the name by which your CrowdID application is known to your Crowd server.

Screenshot: 'Application Name'



RELATED TOPICS

- 1.1 How CrowdID works with Crowd
  - 1.1.1 Determining the name of the CrowdID application
  - 1.1.2 Locating the Crowd Server that CrowdID is using
- 1.1 How OpenID sites interact with CrowdID

Crowd Documentation

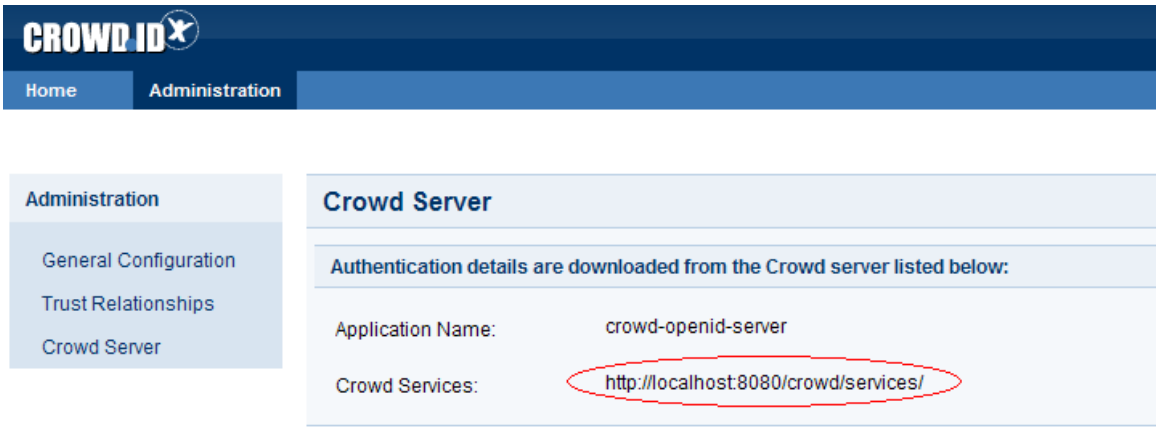
1.1.2 Locating the Crowd Server that CrowdID is using

To change the details or users of your CrowdID application within Crowd, you will need to login to your Crowd server.

To determine the location of your Crowd server,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**Crowd Server**' link in the left navigation column.
4. This will display the '**Crowd Server**' details.
   The '**Crowd Services**' field contains the URL of your Crowd server. Go to this URL to login to Crowd.

Screenshot: 'Crowd Server'



RELATED TOPICS

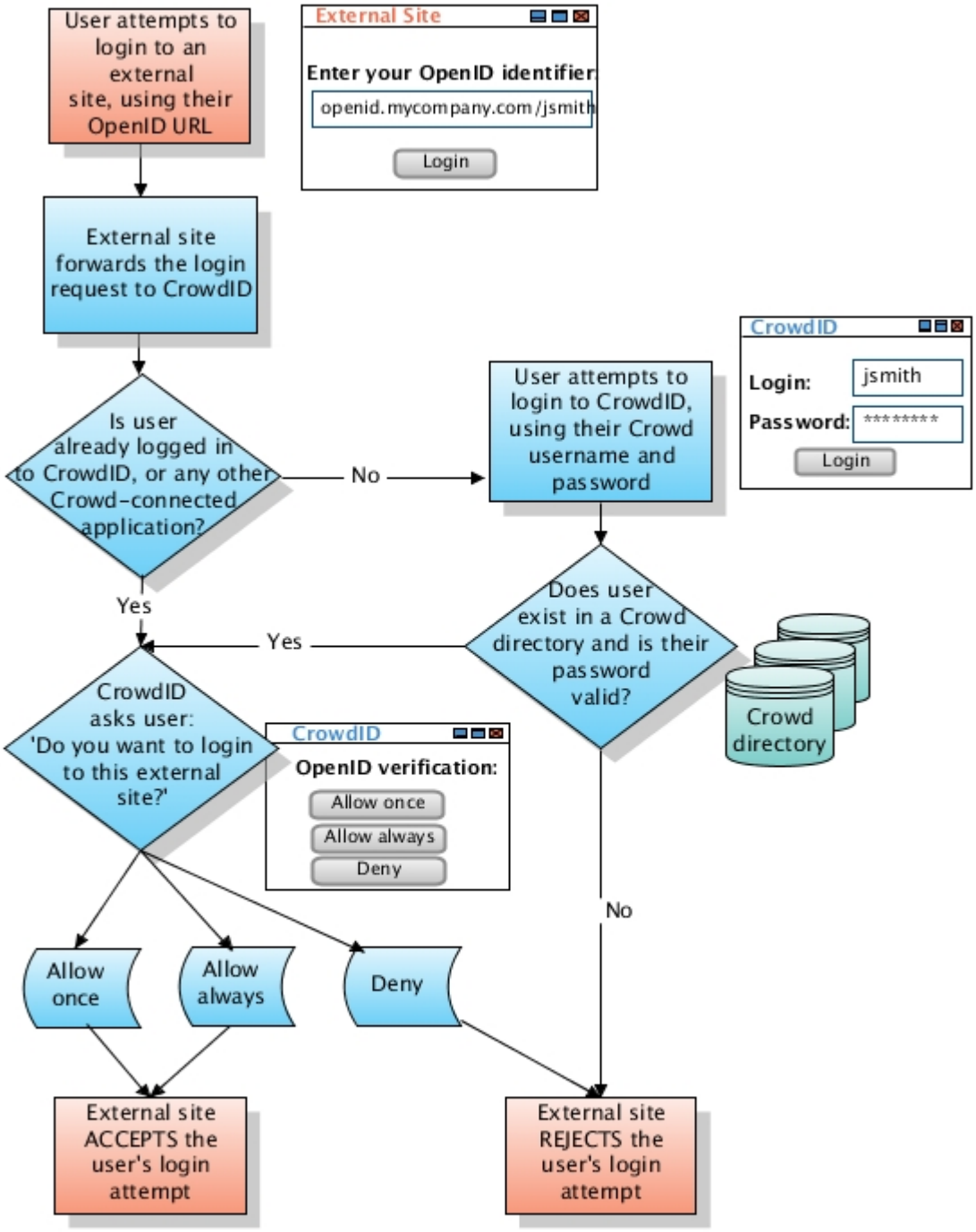
- 1.1 How CrowdID works with Crowd
  - 1.1.1 Determining the name of the CrowdID application
  - 1.1.2 Locating the Crowd Server that CrowdID is using
- 1.1 How OpenID sites interact with CrowdID

Crowd Documentation

1.1 How OpenID sites interact with CrowdID

This diagram shows how an OpenID-enabled website (known as a 'Relying Party') interacts with CrowdID (an 'OpenID Provider') to validate an end-user's login attempt.

For more information about the OpenID protocol please see http://openid.net.



RELATED TOPICS

- 1.1 How CrowdID works with Crowd
  - 1.1.1 Determining the name of the CrowdID application
  - 1.1.2 Locating the Crowd Server that CrowdID is using
- 1.1 How OpenID sites interact with CrowdID

Crowd Documentation

2. Allowing users to access CrowdID

Granting access to CrowdID is done through Crowd. You can grant people rights to:

- *use* CrowdID —
  Granting CrowdID access rights to a user allows them to use CrowdID to access OpenID websites and perform all the actions described in the *CrowdID User Guide*.
- *administer* CrowdID —
  Granting administration rights to a user allows them to use the '**Administration**' menu within CrowdID, which enables them to perform the actions described in the *CrowdID Administration Guide*.

2.1 Granting CrowdID access rights to a user

Granting CrowdID access rights to a user allows them to use CrowdID to access OpenID websites and perform all the actions described in the *CrowdID User Guide*.
Access to CrowdID is managed via Crowd. A user can only access CrowdID if they belong to a directory that is *mapped* to the CrowdID application within Crowd.

To grant CrowdID access rights to a particular user,

1. Login to your Crowd server[1].
2. View your CrowdID application[2] as described in Using the Application Browser in the *Crowd Administration Guide*.
3. Click the '**Directories**' tab to see a list of directories that are mapped to your CrowdID application. You will need to add the user to one of these directories.
4. If your directory capabilities permit, add the user to the directory via Crowd as described in Adding a User in the *Crowd Administration Guide*. (Otherwise you may need to use your specific directory-management tool, instead of Crowd, to add the user to the directory.)

To grant CrowdID access rights to *all* the users in a particular directory,

1. Login to your Crowd server[1].
2. Map the directory to your CrowdID application[2] as described in Mapping a Directory to an Application in the *Crowd Administration Guide*.

To grant CrowdID access rights to a particular *group* of users within a directory,

1. Login to your Crowd server[1].
2. Map the group to your CrowdID application[2] as described in Specifying which Groups can access an Application in the *Crowd Administration Guide*.

[1] To find your Crowd server's URL, see 1.1.2 Locating the Crowd Server that CrowdID is using.
[2] To identify the name by which your CrowdID application is known within Crowd, see 1.1.1 Determining the name of the CrowdID application .

RELATED TOPICS

- 2.1 Granting CrowdID access rights to a user
- 2.2 Granting CrowdID Administration Rights to a User

Crowd Documentation

RELATED TOPICS

- 2.1 Granting CrowdID access rights to a user
- 2.2 Granting CrowdID Administration Rights to a User

Crowd Documentation

2.2 Granting CrowdID Administration Rights to a User

Granting administration rights to a user allows them to use the '**Administration**' menu within CrowdID, which enables them to perform the actions described in the *CrowdID Administration Guide*.

CrowdID administration rights are managed via Crowd. To grant administration rights to a user, you need to add them to the '**crowd-administrators**' group as described below.

Note:

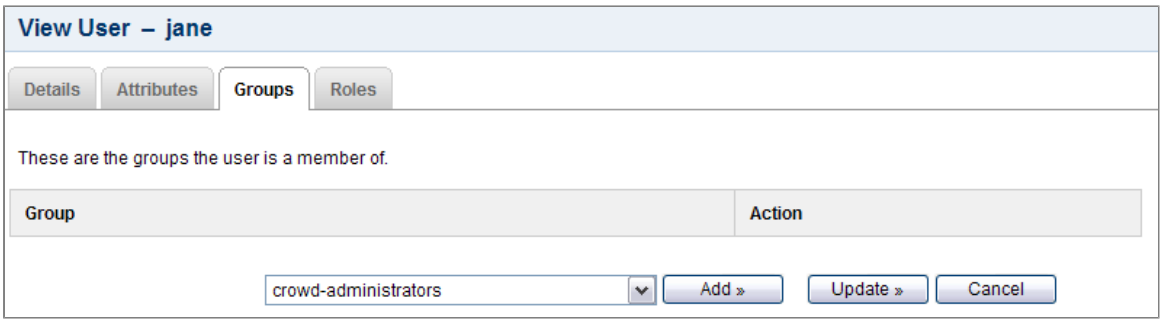
- Adding a user to the '**crowd-administrators**' group will also give them Crowd administration rights (unless you choose to use a different group to contain Crowd administrators). See Granting Crowd Administration Rights to a User in the *Crowd Administration Guide*.
- The '**crowd-administrators**' group always contains CrowdID administrators, regardless of whether you are using it to contain Crowd administrators.

To grant administration rights to a user,

1. Log in to your Crowd server[1].
2. Click the '**Users**' tab in the top navigation bar.
3. This will display the User Browser. Select the directory that contains the user to whom you wish to grant administration rights.
4. Use the '**Search**' to locate the user, then click the '**View**' link that corresponds to the user.
5. This will display the '**User Details**' screen. Click the '**Groups**' tab.
6. A list of the user's current groups (if any) will be displayed. Select the '**crowd-administrators**' group from the drop-down box below the list, then click the '**Add**' button.

[1] To find your Crowd server's URL, see 1.1.2 Locating the Crowd Server that CrowdID is using.

Screenshot: Granting Crowd administration rights



RELATED TOPICS

- 2.1 Granting CrowdID access rights to a user
- 2.2 Granting CrowdID Administration Rights to a User

Crowd Documentation

3. Specifying the sites to which users can login

There are three ways to specify which OpenID hosts (i.e. websites or IP addresses) your users can login to using their CrowdID:

- No restriction — your CrowdID users can login to any OpenID host
- Blacklist — your CrowdID users can login to any OpenID host except the one(s) that you specify

- Whitelist — your CrowdID users can login to only those OpenID host(s) that you specify

3.1 Allowing all hosts

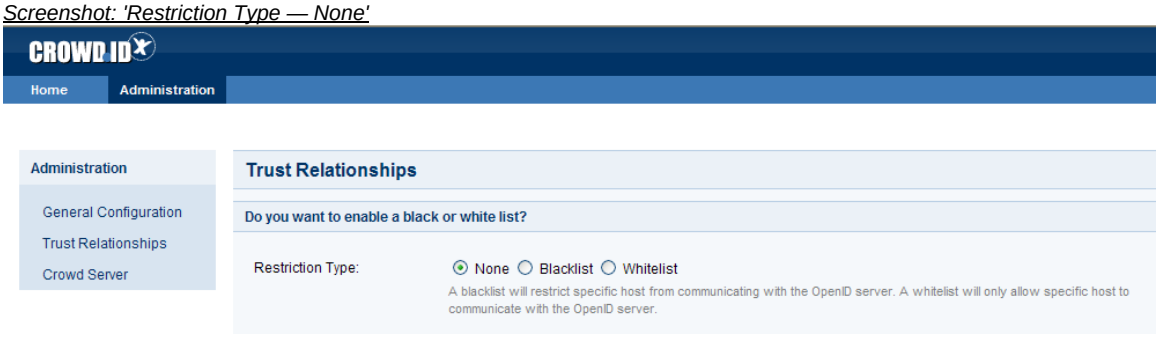
There are three ways to specify which OpenID hosts (i.e. websites or IP addresses) your users can login to using their CrowdID:

- No restriction — your CrowdID users can login to any OpenID host
- Blacklist — your CrowdID users can login to any OpenID host except the one(s) that you specify
- Whitelist — your CrowdID users can login to only those OpenID host(s) that you specify

To allow users to login to any OpenID host,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**Trust Relationships**' link in the left navigation column.
4. For '**Restriction Type**', select '**None**'.

Screenshot: 'Restriction Type — None'



RELATED TOPICS

- 3.1 Allowing all hosts
- 3.2 Allowing all except specified hosts ('Blacklist')
- 3.3 Allowing specified hosts only ('Whitelist')

Crowd Documentation

3.2 Allowing all except specified hosts ('Blacklist')

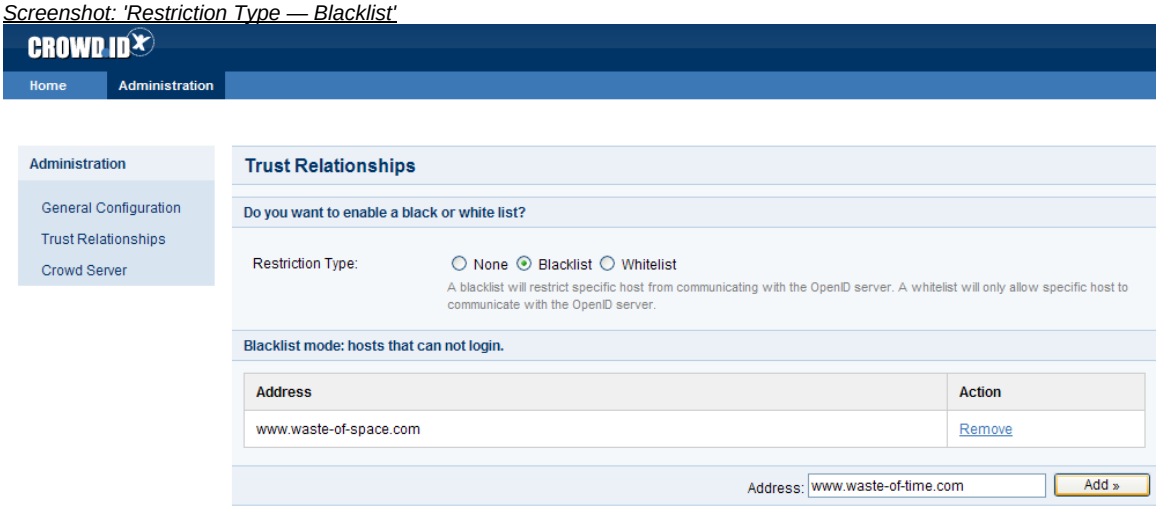
There are three ways to specify which OpenID hosts (i.e. websites or IP addresses) your users can login to using their CrowdID:

- No restriction — your CrowdID users can login to any OpenID host
- Blacklist — your CrowdID users can login to any OpenID host except the one(s) that you specify
- Whitelist — your CrowdID users can login to only those OpenID host(s) that you specify

To specify an OpenID blacklist,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**Trust Relationships**' link in the left navigation column.
4. For '**Restriction Type**', select '**Blacklist**'.
5. Wait for a section titled '**Blacklist mode: hosts that can not login**' to appear on the screen.
6. For each site to which you want to prevent users logging in,
    a. Type the URL or IP address in the '**Address**' field.
    b. Click the '**Add**' button.

Screenshot: 'Restriction Type — Blacklist'



RELATED TOPICS

- 3.1 Allowing all hosts
- 3.2 Allowing all except specified hosts ('Blacklist')
- 3.3 Allowing specified hosts only ('Whitelist')

Crowd Documentation

3.3 Allowing specified hosts only ('Whitelist')

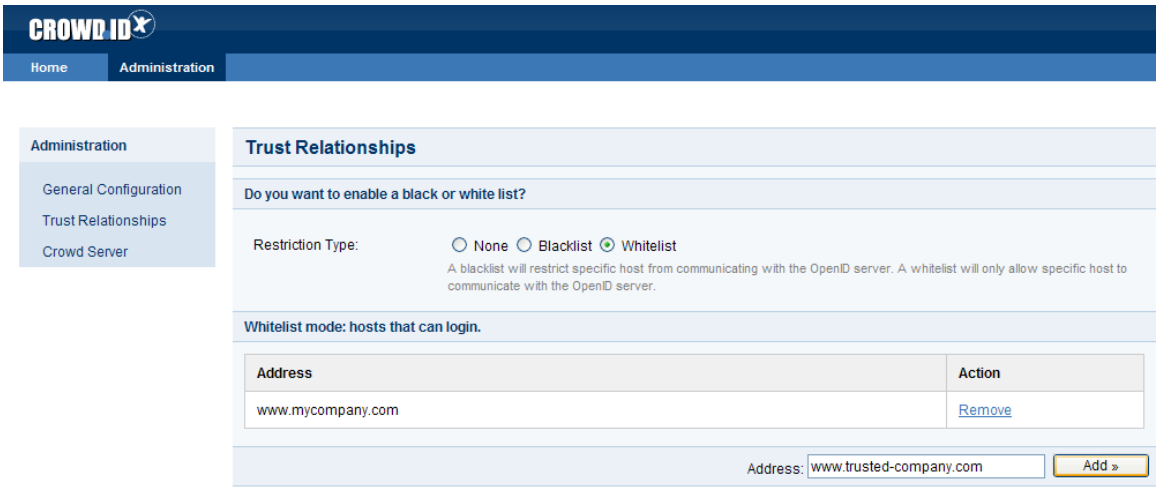
There are three ways to specify which OpenID hosts (i.e. websites or IP addresses) your users can login to using their CrowdID:

- No restriction — your CrowdID users can login to any OpenID host
- Blacklist — your CrowdID users can login to any OpenID host except the one(s) that you specify
- Whitelist — your CrowdID users can login to only those OpenID host(s) that you specify

To specify an OpenID whitelist,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**Trust Relationships**' link in the left navigation column.
4. For '**Restriction Type**', select '**Blacklist**'.
5. Wait for a section titled '**Whitelist mode: hosts that can login**' to appear on the screen.
6. For each site to which you want to allow users to login,
    a. Type the URL or IP address in the '**Address**' field.
    b. Click the '**Add**' button.

Screenshot: 'Restriction Type — Whitelist'



RELATED TOPICS

- 3.1 Allowing all hosts
- 3.2 Allowing all except specified hosts ('Blacklist')
- 3.3 Allowing specified hosts only ('Whitelist')

Crowd Documentation

4. Configuring CrowdID system settings

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

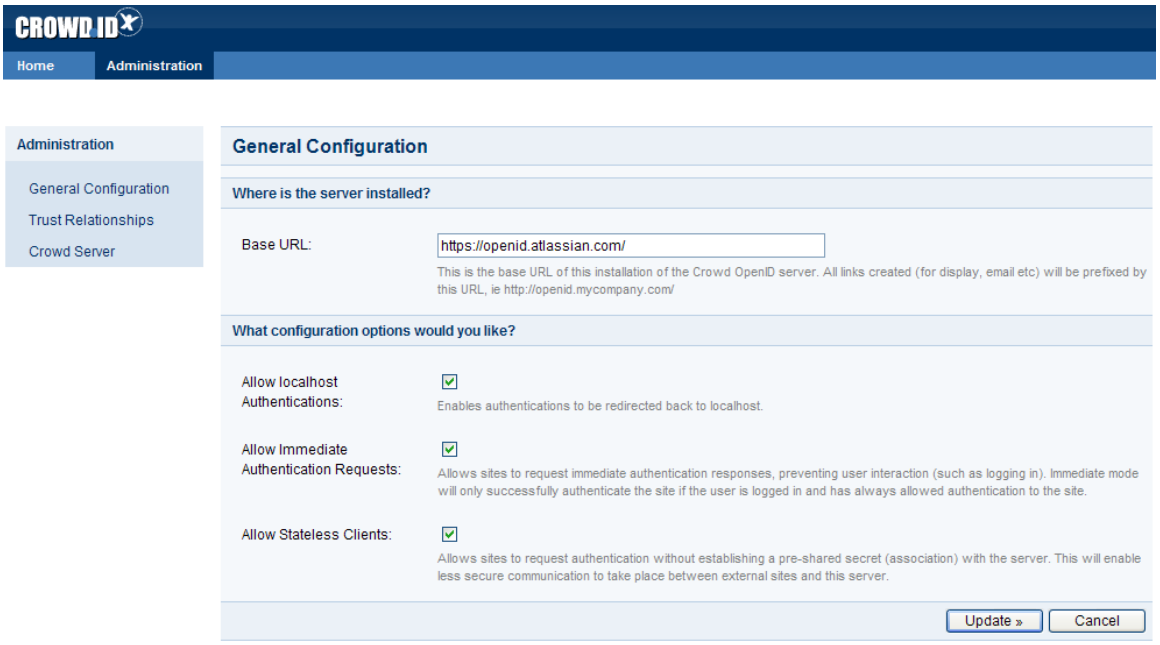
4.1 Specifying the CrowdID URL

The **CrowdID URL** is the URL that your end-users will type when logging into OpenID-enabled websites.

To define the URL of your CrowdID instance,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**General Configuration**' link in the left navigation column.
4. Type the URL into the '**Base URL**' field.
5. Click the '**Update**' button.

Screenshot: 'General Configuration'



RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

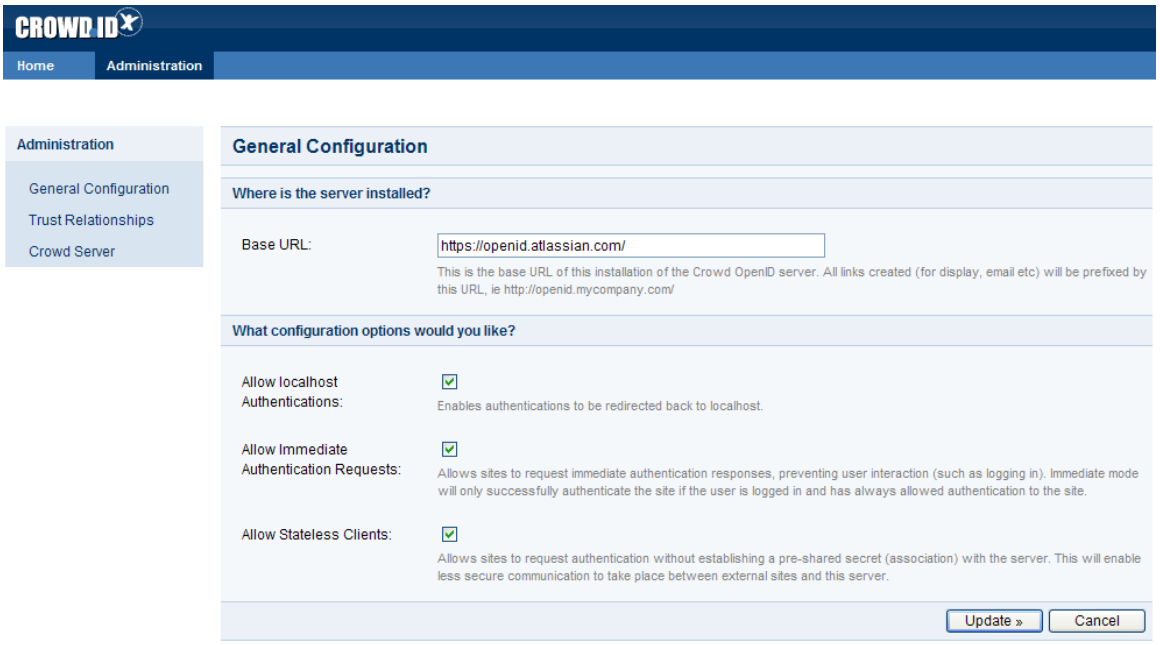
4.2 Enabling localhost authentication

Enabling **localhost authentication** prevents OpenID-enabled sites from directly accessing your end-users' local machines.

To enable localhost authentication,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**General Configuration**' link in the left navigation column.
4. Select the '**Allow localhost authentications**' checkbox.
5. Click the '**Update**' button.

Screenshot: 'General Configuration'



RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

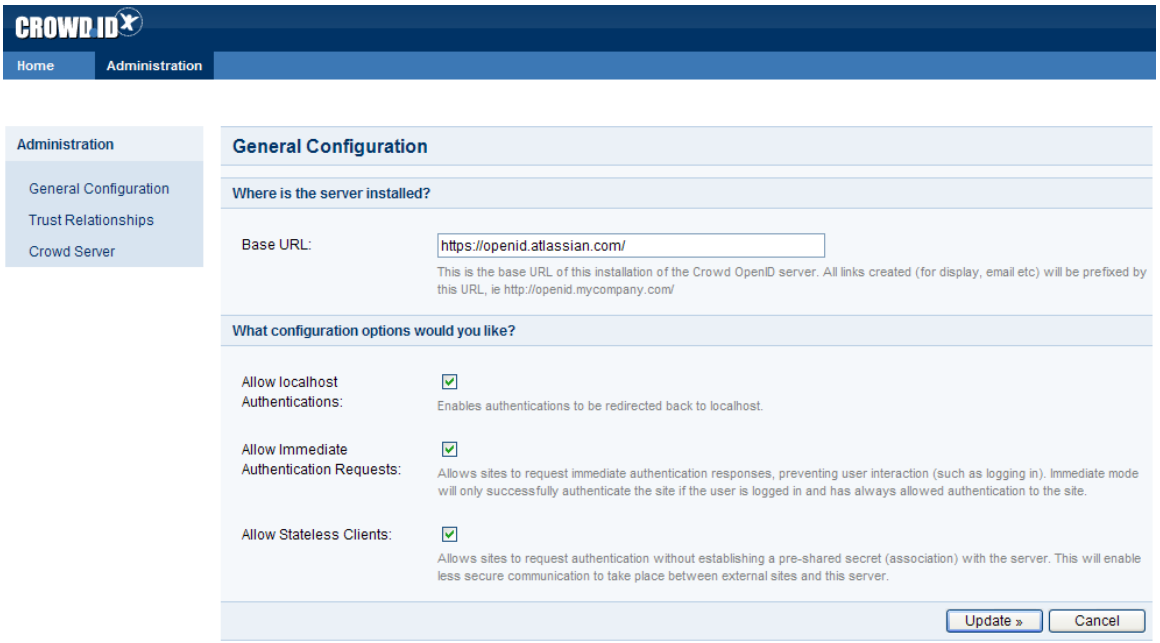
4.3 Enabling immediate authentication requests

Enabling '**Allow immediate authentication requests**' allows an OpenID-enabled site to check whether the user is logged in, without actually prompting the user to login. Known as *pass-through authentication*, this provides greater convenience for end-users, particularly when an end-user visits a site for which they have previously selected 'Allow Always' (see 2.4 Allowing or denying a login in the *CrowdID User Guide*).

To enable 'Allow immediate authentication requests',

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**General Configuration**' link in the left navigation column.
4. Select the '**Allow immediate authentication requests**' checkbox.
5. Click the '**Update**' button.

Screenshot: 'General Configuration'



RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

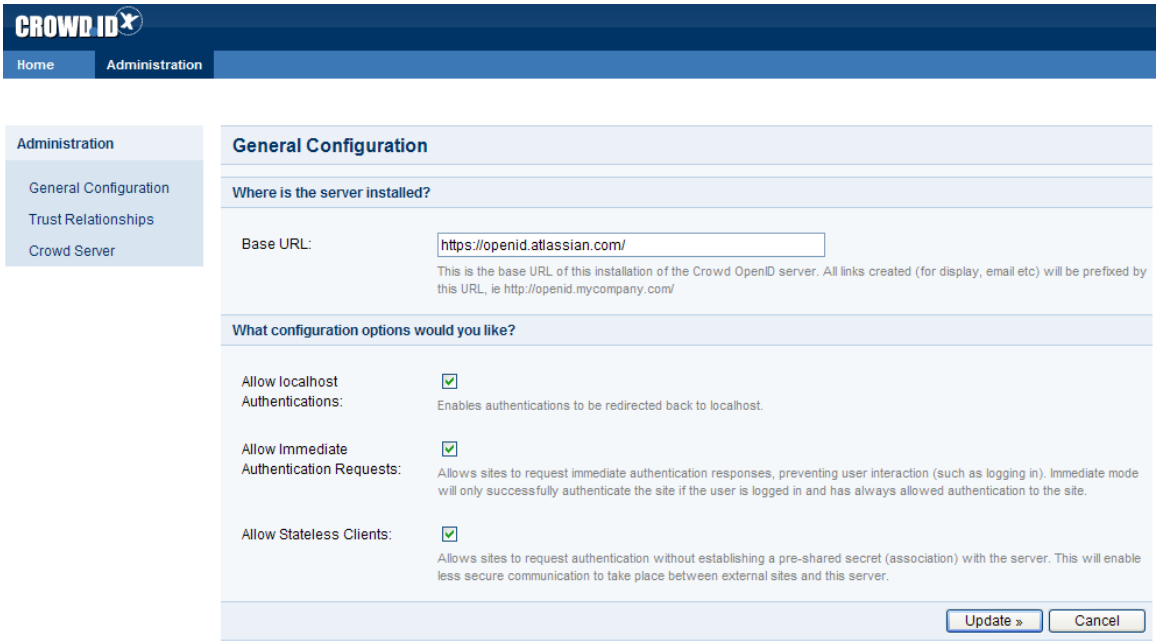
4.4 Enabling communication with stateless clients

Some OpenID-enabled sites do not support pre-shared secrets (associations). Selecting **allow stateless clients** enables your CrowdID server to communicate with such sites.

To allow stateless clients,

1. Login to CrowdID.
2. Click the '**Administration**' link in the top navigation bar.
3. Click the '**General Configuration**' link in the left navigation column.
4. Select the '**Allow stateless clients**' checkbox.
5. Click the '**Update**' button.

Screenshot: 'General Configuration'



RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

RELATED TOPICS

- 4.1 Specifying the CrowdID URL
- 4.2 Enabling localhost authentication
- 4.3 Enabling immediate authentication requests
- 4.4 Enabling communication with stateless clients

Crowd Documentation

Crowd FAQ

Crowd Frequently Asked Questions

Known issues, hints and tips and answers to commonly raised questions about Crowd:

General FAQ on the Atlassian Website

Concepts:

- What is single sign-on (SSO)?
- What is authorisation?
- What is authentication?
- What is centralised authentication?
- What is identity management?
- What is a directory?

Technical:

- How does Crowd work? How is Crowd an "application security framework"?
- What is an application connector?
- What is a directory connector?
- How many users can Crowd manage?
- How many applications can be used with Crowd?
- We already have an LDAP server for Confluence and/or JIRA. Do we really need Crowd?

Compatibility:

- What are Crowd's system requirements?
- What directories and applications does Crowd support out of the box?
- How can Crowd be connected to new or currently unsupported applications?
- How does Crowd integrate with other Atlassian products?
- Does Crowd include Kerberos integration?
- Does Crowd support SAML or Liberty Alliance?

Common Evaluator Questions:

- Can Crowd run alongside another SSO solution?
- Can I setup a user frontend and login page for Crowd?
- Can I setup password-only delegated LDAP and AD integration?
- How can I filter unwanted LDAP entries?
- How do I fix a 'User Limited Exceeded' error?
- How do I fix slow performance?
- Is clustering supported?

Deployment FAQ

- Finding the atlassian-crowd.log File
- Finding your Crowd Home Directory
- Recovering your Console application password
- Resetting the Domain Cookie Value
- Restarting the Setup Wizard from Scratch
- Self Signed Certificate
- Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX
- Using Crowd in a Cluster is Not Supported

Guides, Hints and Tips

- Principals and Users
- Using Apache Directory Studio for Crowd LDAP Configuration
  - Creating a Connection to your LDAP Directory
  - Getting an LDIF Export of a User or Group
  - Restricting LDAP Scope for User and Group Search

Integration FAQ

- All Integrations
  - If I delete a user from Crowd, how will this affect integrated applications?
  - Passing the crowd.properties File as an Environment Variable
- Atlassian Product Integration
  - Application Caching
  - JIRA integration
  - Public Signup Setup
- IBM Lotus Domino Integration
- IBM Websphere Integration

Support Policies

- Bug Fixing Policy
- How to Report a Security Issue
- New Features Policy
- Patch Policy
- Security Advisory Publishing Policy
- Security Patch Policy
- Severity Levels for Security Issues

Troubleshooting

- Finding Known Issues
- Characters in User or Group DN's that will cause problems when using Crowd
- Problems when Importing Users into MySQL
- Troubleshooting LDAP Error Codes
  - Active Directory LDAP Errors
- Troubleshooting SSL certificates and Crowd
- How to Optimise Crowd Client Caching
- Troubleshooting Crowd Performance
- Troubleshooting SSO with Crowd
  - Debugging SSO in environments with Proxy Servers
- Troubleshooting CrowdID

RELATED TOPICS

- Troubleshooting your Configuration on Setup

Deployment FAQ

- Finding the atlassian-crowd.log File
- Finding your Crowd Home Directory
- Recovering your Console application password
- Resetting the Domain Cookie Value
- Restarting the Setup Wizard from Scratch
- Self Signed Certificate
- Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX
- Using Crowd in a Cluster is Not Supported

Finding the atlassian-crowd.log File

When you report a problem to Atlassian Support, we may ask you to send us your `atlassian-crowd.log` file. The location of the log file may vary, depending on your Crowd installation type. Provided that you have not changed the log file location from the default, the Crowd log file is at the location described below.

Installation Type	Location of Log File
Crowd Standalone edition	The root directory of your Crowd application, e.g. `atlassian-crowd-2.0.0/atlassian-crowd.log`
Crowd Standalone running as a Windows service	`C:\Windows\system32\atlassian-crowd.log`
Crowd WAR edition	The directory from which you start the application server, e.g. `apache-tomcat-6.0.16/bin/atlassian-crowd.log`

How do I Change the Location?

You can change the location of the log file by modifying the following line in the `WEB-INF/classes/log4j.properties` file of your Crowd installation to use an absolute file path:

```
log4j.appender.filelog.File=atlassian-crowd.log
```

For more information, please refer to the page on logging and profiling.

RELATED TOPICS

Logging and Profiling
Important Directories and Files

Finding your Crowd Home Directory

The **Crowd Home** directory is where Crowd stores its configuration information. If you are using the embedded HSQLDB database supplied for evaluation purposes, Crowd will also store its database in this directory.

Crowd's System Information screen shows the location of your Crowd Home directory.

Read more about:

- Setting your Home Directory during installation.
- The location and function of the Crowd Home directory and other important files and directories.

Recovering your Console application password

The Crowd console itself must authenticate to the Crowd framework to perform authentication and authorisation calls.

Like an integrated application, if you have an improper password in the `crowd.properties` configuration file, the following exception will be thrown when the application attempts to connect to Crowd SOAP services:

```
Caused by: com.atlassian.crowd.integration.exception.InvalidAuthenticationException: Invalid
application client.
 at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
 at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:39)
 at
sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:27)
at java.lang.reflect.Constructor.newInstance(Constructor.java:494)
 at org.codehaus.xfire.aegis.type.basic.BeanType.createFromFault(BeanType.java:235)
 at org.codehaus.xfire.aegis.type.basic.BeanType.readObject(BeanType.java:105)
 at org.codehaus.xfire.aegis.AegisBindingProvider.readParameter(AegisBindingProvider.java:169)
 at
org.codehaus.xfire.client.ClientFaultConverter.processFaultDetail(ClientFaultConverter.java:51)
 at org.codehaus.xfire.client.ClientFaultConverter.invoke(ClientFaultConverter.java:32)
 at org.codehaus.xfire.handler.HandlerPipeline.invoke(HandlerPipeline.java:131)
 at org.codehaus.xfire.client.Client.onReceive(Client.java:424)
 at org.codehaus.xfire.transport.http.HttpChannel.sendViaClient(HttpChannel.java:139)
 at org.codehaus.xfire.transport.http.HttpChannel.send(HttpChannel.java:48)
 at org.codehaus.xfire.handler.OutMessageSender.invoke(OutMessageSender.java:26)
 at org.codehaus.xfire.handler.HandlerPipeline.invoke(HandlerPipeline.java:131)
 at org.codehaus.xfire.client.Invocation.invoke(Invocation.java:79)
 at org.codehaus.xfire.client.Invocation.invoke(Invocation.java:114)
 at org.codehaus.xfire.client.Client.invoke(Client.java:336)
 at org.codehaus.xfire.client.XFireProxy.handleRequest(XFireProxy.java:77)
 at org.codehaus.xfire.client.XFireProxy.invoke(XFireProxy.java:57)
 at $Proxy8.authenticateApplication(Unknown Source)
 at
com.atlassian.crowd.integration.service.soap.client.GenericClient.authenticate(GenericClient.java:263
... 73 more
Caused by: org.codehaus.xfire.fault.XFireFault: Invalid application client.
 at org.codehaus.xfire.fault.Soap11FaultSerializer.readMessage(Soap11FaultSerializer.java:31)
 at org.codehaus.xfire.fault.SoapFaultSerializer.readMessage(SoapFaultSerializer.java:28)
 at org.codehaus.xfire.soap.handler.ReadHeadersHandler.checkForFault(ReadHeadersHandler.java:111)
 at org.codehaus.xfire.soap.handler.ReadHeadersHandler.invoke(ReadHeadersHandler.java:67)
 at org.codehaus.xfire.handler.HandlerPipeline.invoke(HandlerPipeline.java:131)
 at org.codehaus.xfire.client.Client.onReceive(Client.java:406)
 ... 84 more
```

If the password for the Crowd console is lost, the only method of recovery is to reset the password in the `crowd.properties` configuration file to a known application password. To do this you will need to have access to the Crowd database server and run the following commands:

1. Get a list of the applications integrated with Crowd:

```
mysql> select id, application_name from cwd_application;
+--------+--------------------+
| id     | application_name   |
+--------+--------------------+
|  98305 | crowd              |
|  98306 | demo               |
|  98307 | crowd-openid-server |
| 655361 | jira               |
| 753665 | jiveforums         |
+--------+--------------------+
```

2. Choose an application for which you have the password, and where you're happy to use the same password for the Crowd application. Let's call your application 'X'. Use application X's `application_name` to query the database and retrieve X's credentials:

```
mysql> select credential from cwd_application where name = 'jira';
+-----------------------------------------------------------------------------------------+
| credential                                                                              |
+-----------------------------------------------------------------------------------------+
| sQnzu7wkTrgkQZF+0G1hi5AI3Qmzvv0bXgc5THBqi7mAsdd4Xll27ASbRt9fEyavWi6m0QP9B8lThf+rDKy8hg== |

+-----------------------------------------------------------------------------------------+
```

3. Now set Crowd's application credentials to the `credential` of your application X:

```
mysql> update cwd_application set credential =
'sQnzu7wkTrgkQZF+0G1hi5AI3Qmzvv0bXgc5THBqi7mAsdd4Xll27ASbRt9fEyavWi6m0QP9B8lThf+rDKy8hg==='
where application_name = 'crowd';
Query OK, 0 rows affected (0.00 sec)
Rows matched: 1  Changed: 0  Warnings: 0
```

4. Update your `crowd.properties` `application.password` value to the value of X's password. If you are using Crowd 1.5 or earlier, the file is located at `atlassian-crowd-X.X.X/crowd-webapp/WEB-INF/classes/`. If using 1.5.1 or later, the file will be located inside your Crowd-Home Directory.
5. You may now start Crowd.

Further information

- If you have installed only Crowd and no other integrated applications, you'll need to clear all the database tables (if you've already hooked up to a database server) and re-install Crowd. This should not cause you to lose much data, since no other applications have yet been defined.
- The issue is that the password for the crowd application is being changed during the setup process for crowd. This problem will be resolved with Crowd 1.2 - see CWD-488.
- You may be tempted to try changing the password back to 'password'. Alas, this won't work, because the passwords are encrypted using SHA1.

Resetting the Domain Cookie Value

If you have set the SSO Domain to an invalid value, you may be prevented from authenticating to the Crowd Console.

To reset the SSO (single sign-on) cookie domain, run the following SQL command on the Crowd database:

```
1.update cwd_property set property_value = '' where property_name = 'domain';
```

Once you have done this you will need to restart Crowd and then log in. This will reset any domain SSO token misconfiguration.

Restarting the Setup Wizard from Scratch

If you get part-way through the Crowd Setup Wizard and then decide you want to start again from scratch, you can delete the **Crowd Home** directory. (See Important Directories and Files.)

Crowd uses the `crowd.cfg.xml` file, stored in the Crowd Home directory, to 'remember' the step you have reached in the setup procedure. Clearing the file will cause the Setup Wizard to start at the beginning again.

This strategy is useful if you want to re-do your setup without having to download Crowd again.

To restart the Crowd Setup Wizard:

1. Shut down Crowd.
2. Delete your **Crowd Home** directory.
3. Start Crowd again.
4. Go go `http://localhost:8095/crowd`.
5. The Crowd Setup Wizard will start. Follow the steps from the beginning, as described in Running the Setup Wizard.

Embedded database will disappear too
If you are using the embedded database, the database files are stored in the Crowd Home directory too. Deleting the Crowd Home directory will remove all your Crowd Administration Console data as well (users, groups, roles, directories, applications and other configuration data).

Self Signed Certificate

I have a self Signed Certificate

You will need to add the self-signed certificate to your JDK truststore using the JDK keytool:
http://java.sun.com/j2se/1.3/docs/tooldocs/win32/keytool.html

Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

This page contains some useful information about running Crowd under Linux/UNIX:

- **Dedicated system user.** For security reasons, and to keep your system administrator happy, you should probably create a dedicated non-root user to run Crowd.
- **Automatic startup.** It is useful to set up Crowd to run automatically on UNIX startup.

Running Crowd as an Unprivileged User

Here is an example of some of the changes you can make to *harden up* the directory and file permissions for Crowd to run as a non-root user.

You will need to update the environment variables to suit your installation. This is also for use in BASH. If you are using a different shell, you might need to tweak some things.

```
01.#!/bin/bash
02.CROWD_USER="crowd"
03.CROWD_GROUP="crowd"
04.INSTALL_BASE="/opt/crowd/atlassian-crowd-2.0.3"
05.CROWD_HOME="/opt/crowd/crowd-home"
06.sudo chgrp ${CROWD_GROUP}
${INSTALL_BASE}/apache-tomcat/{logs,work,bin/{catalina,startup,shutdown,setclasspath}.sh}
07.sudo chmod g+x ${INSTALL_BASE}/apache-tomcat/bin/{catalina,startup,shutdown,setclasspath}.sh
08.sudo chmod g+w ${INSTALL_BASE}/apache-tomcat/{logs,work}
09.sudo chmod -R g+w ${CROWD_HOME}
```

Getting Crowd to Start Automatically

1. Create an `init.d file` (for example, 'crowd.init.d') inside your {`CROWD_INSTALL`} directory:

```
01.#!/bin/bash
02.# Crowd startup script
03.#chkconfig: 2345 80 05
04.#description: Crowd
05.
06.
07.# Based on script at http://www.bifrost.org/problems.html
08.
09.RUN_AS_USER=crowd
10.CATALINA_HOME=/opt/crowd/apache-tomcat
11.
12.start() {
13.                     echo "Starting Crowd: "
14.                     if [ "x$USER" != "x$RUN_AS_USER" ]; then
15.                          su - $RUN_AS_USER -c "$CATALINA_HOME/bin/startup.sh"
16.                     else
17.                          $CATALINA_HOME/bin/startup.sh
18.                     fi
19.                     echo "done."
20.}
21.stop() {
22.                     echo "Shutting down Crowd: "
23.                     if [ "x$USER" != "x$RUN_AS_USER" ]; then
24.                          su - $RUN_AS_USER -c "$CATALINA_HOME/bin/shutdown.sh"
25.                     else
26.                          $CATALINA_HOME/bin/shutdown.sh
27.                     fi
28.                     echo "done."
29.}
30.
31.case "$1" in
32.     start)
33.                     start
34.                     ;;
35.     stop)
36.                     stop
37.                     ;;
38.     restart)
39.                     stop
40.                     sleep 10
41.                     #echo "Hard killing any remaining threads.."
42.                     #kill -9 `cat $CATALINA_HOME/work/catalina.pid`
43.                     start
44.                     ;;
45.     *)
46.                     echo "Usage: $0 {start|stop|restart}"
47.esac
48.
49.exit 0
```

2. Create a symbolic link from `/etc/init.d/crowd` to the `init.d file` file.

Thank you for this information
Thank you to Matthew Block and Pete Toscano for the original comments that we based this information on.

Using Crowd in a Cluster is Not Supported

Atlassian does not support clustering of Crowd, and we have not yet scheduled cluster support into the Crowd roadmap. The reason is that clustering problems are hard to diagnose and we do not have the expertise in-house to support the many possible configurations.

There is a feature request in CWD-1053. You can vote for the feature request, and "watch" it to receive progress reports.

Some of our customers are using Crowd clustering successfully. You may find some useful information in the discussion threads in our user forums here and here.

Guides, Hints and Tips

- Principals and Users
- Using Apache Directory Studio for Crowd LDAP Configuration

Principals and Users

As far as Crowd is concerned, the terms '**principals**' and '**users**' are equivalent — they mean the same thing. Earlier versions of Crowd used the term 'principals'. From Crowd 1.3 onwards, we call them 'users'.

Using Apache Directory Studio for Crowd LDAP Configuration

This is a basic tutorial on using a wonderful Eclipse-based LDAP browser, known as Apache Directory Studio, to gather the information you need for your Crowd LDAP configuration.

Before you Start

Step 1. Get Apache Directory Studio

- Download and install Apache Directory Studio.

Step 2. (Optional) Do Some Background Reading

If you are an LDAP newbie, there are two great articles that may help you gain a better understanding of LDAP and LDAP search filters before you begin using Apache Directory Studio:

- An Introduction to LDAP
- How to write an LDAP search filter

Table of Contents

Creating a Connection to your LDAP Directory

Getting an LDIF Export of a User or Group

Restricting LDAP Scope for User and Group Search

RELATED TOPICS

Configuring an LDAP Directory Connector

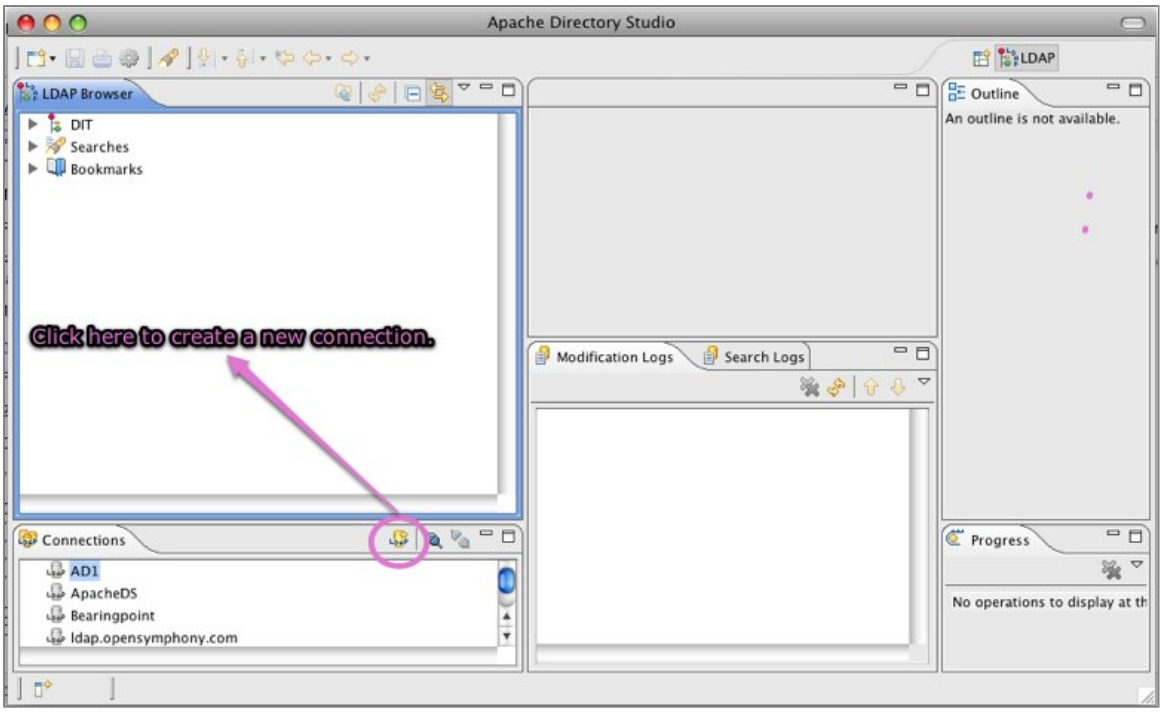
Creating a Connection to your LDAP Directory

You may find an LDAP browser useful to gather the information you need for your Crowd configuration. This page shows you how to create a connection to your LDAP directory when using Apache Directory Studio. You can then use the connection information gathered, to set up your LDAP directory in Crowd.

Step 1: Create a New Connection in Apache Directory Studio

1. Start up Apache Directory Studio.
2. Click the LDAP icon to create a new connection.

Screenshot: Creating a new connection in Apache Directory Studio



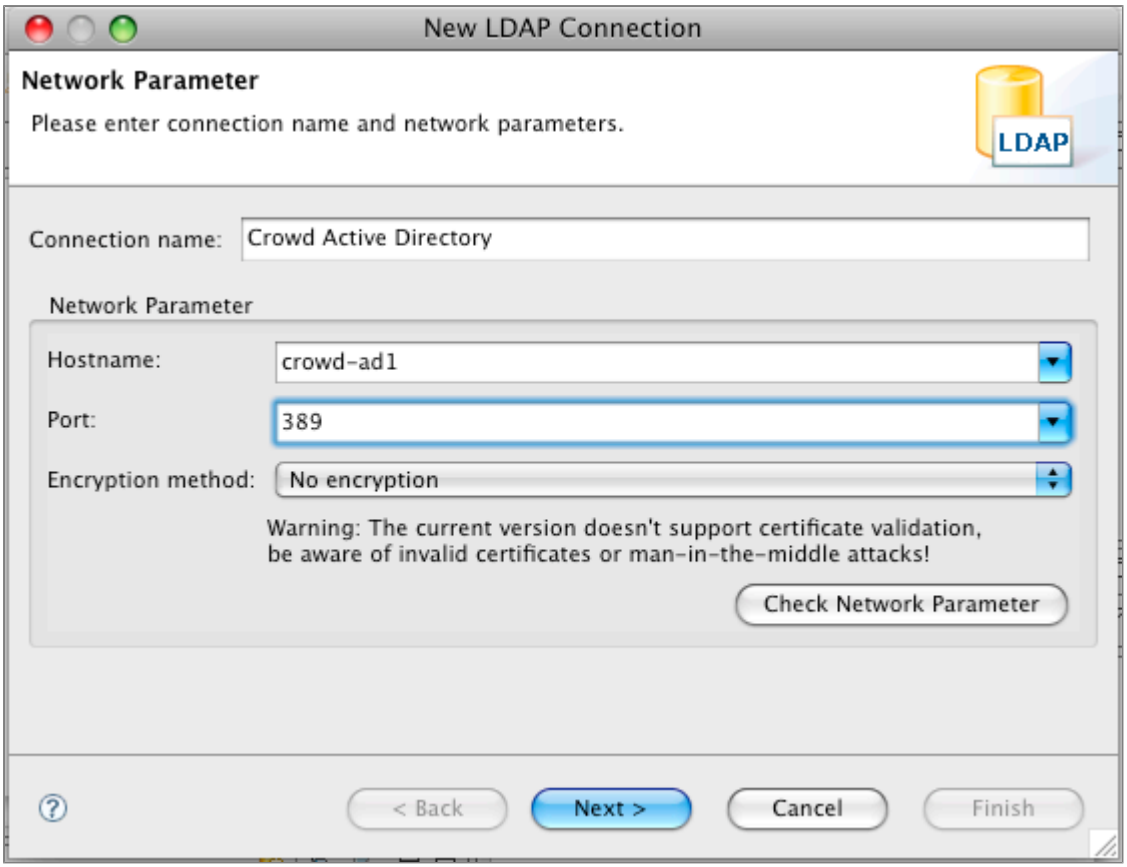
Step 2: Enter your Connection Information

1. Enter a name for your connection.
2. Enter the '**Network Parameter**' information as follows:

Hostname	The domain name for your LDAP server. If the LDAP server is not on the same network as Crowd, you may need to use the FQDN or IP address of the LDAP server.
Port	For normal LDAP connectivity, use 389. For SSL connectivity, use 636.

3. Click the '**Check Network Parameter**' button to ensure your connection is successful.
4. Click '**Next**'.

Screenshot: Entering the connection information in Apache Directory Studio



Step 3: Enter your Authentication Information

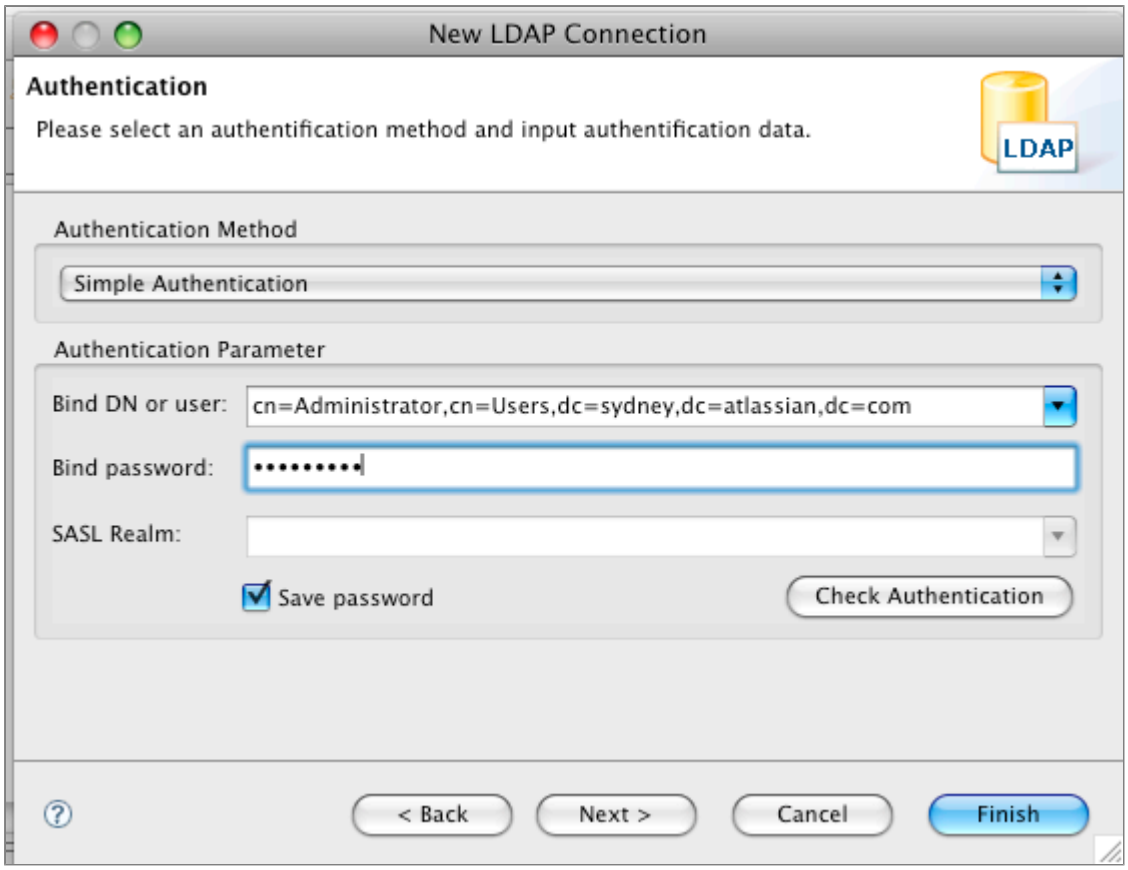
1. Choose the '**Authentication Method**' from the dropdown list.
   Some LDAP servers allow anonymous access. If your LDAP server allows this, you can change the 'Authentication Method' dropdown from 'Simple Authentication' to 'Anonymous Authentication' and click '**Finish**' to go straight to Step 4.

2. Enter the '**Authentication Parameter**' information as follows:

Bind DN or user	Enter the full DN of the account that will be used to connect to the LDAP directory. This account should have the ability to browse the entire LDAP directory tree.
Bind password	Enter the password for the Bind DN account.

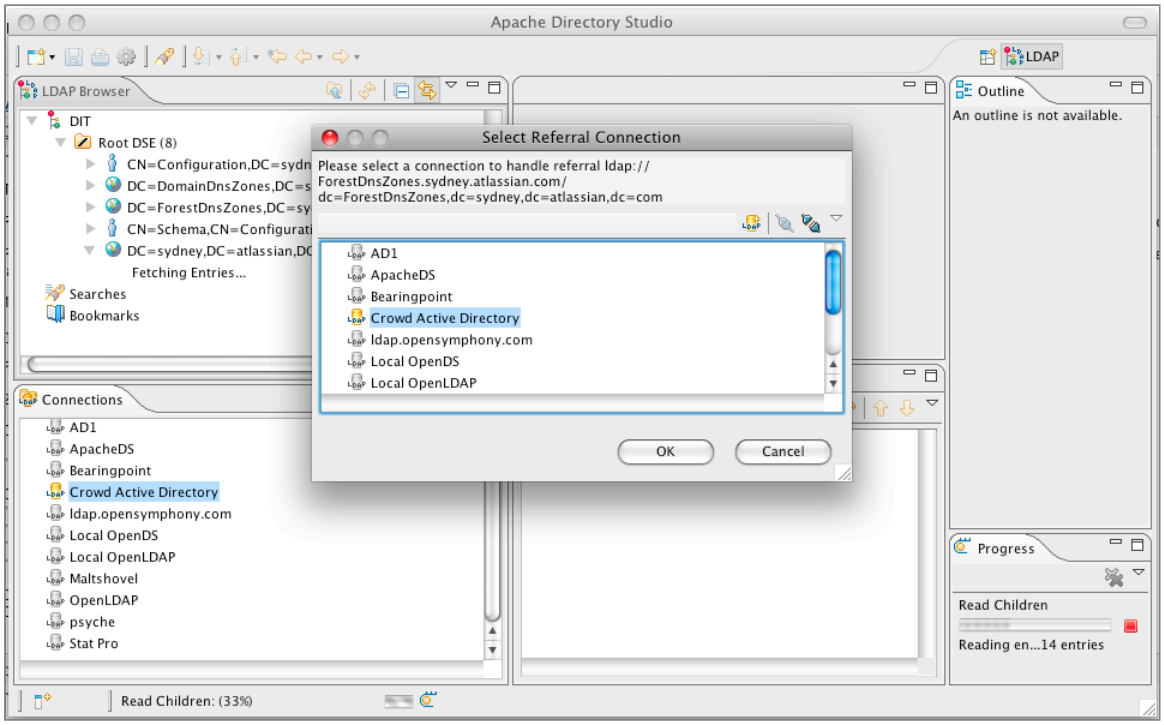
3. Click the '**Check Authentication**' button to ensure this account can authenticate.
4. If this authentication is successful, click '**Finish**'.

Screenshot: Entering the authentication information in Apache Directory Studio



5. If you are prompted for a 'Referral Connection', select the same directory.

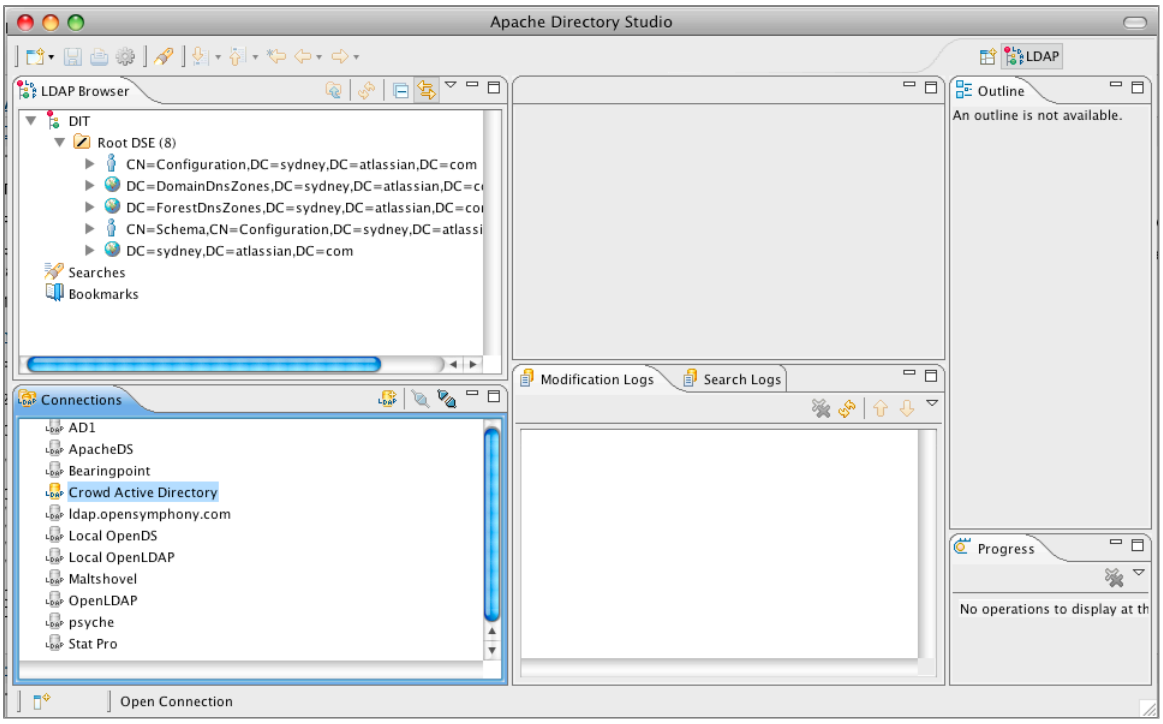
   Screenshot: Selecting a referral connection in Apache Directory Studio



Step 4: See the Base DNs

If the configuration is successful, you should now have a list of the base DNs available under this LDAP directory's root DSE.

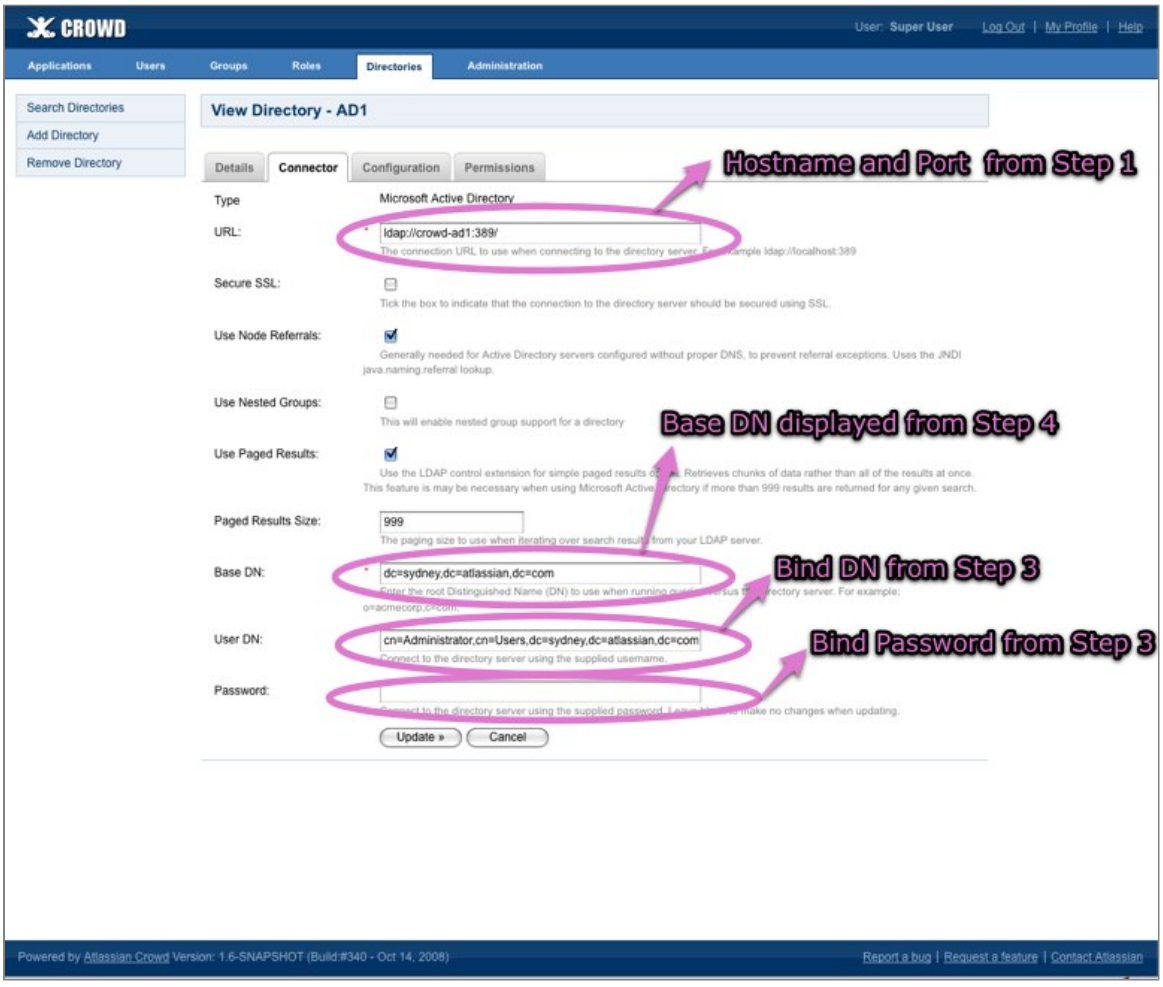
Screenshot: Viewing the base DNs in Apache Directory Studio



Step 5: Use the Same Connection Information in Crowd

Use the same connection information to set up your LDAP directory in Crowd.

Screenshot: LDAP directory configuration in Crowd



RELATED TOPICS

Using Apache Directory Studio for Crowd LDAP Configuration

Configuring an LDAP Directory Connector

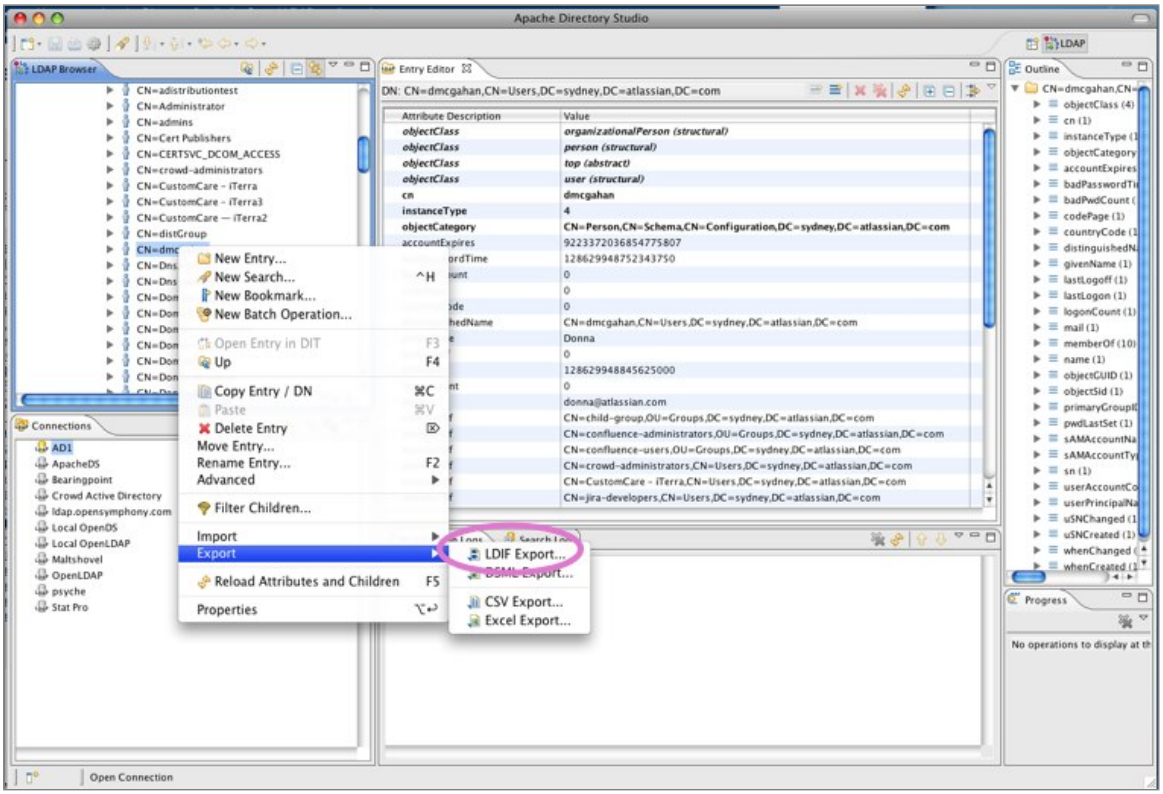
Getting an LDIF Export of a User or Group

Occasionally, Atlassian Crowd Support may request an LDIF export of a user or group. LDIF is the LDAP Data Interchange Format. You can export all or part of your LDAP directory to an LDIF file. This page shows you how to do that when using Apache Directory Studio.

To generate an LDIF export of a user or group,

1. Highlight the user or group in Apache Directory Studio.
2. Right-click on the user or group.
3. Choose **Export -> LDIF Export**.

Screenshot: Generating an LDIF export of a user in Apache Directory Studio



RELATED TOPICS

Creating a Connection to your LDAP Directory
Using Apache Directory Studio for Crowd LDAP Configuration

Restricting LDAP Scope for User and Group Search

While you should already know the user DN you are using for your LDAP connection, it can be helpful to review the users and groups in Apache Directory Studio to determine the best scope for your Crowd LDAP directory configuration.

Crowd comes with default configurations that will work for most customers. In the examples below, we illustrate some common options for changing your user and group configurations.

There are a number of other attributes, not shown here, that can also be used to narrow the scope of users and groups.

Important Search Filter Notes

- If you are unfamiliar with LDAP search filter syntax, please review this guide.
- In order to use Object Filters larger than 255 characters, you will need to upgrade to Crowd to 1.5.1 or later, by installing a new Crowd instance (with a new database) and restoring an XML backup from your previous Crowd installation. For more information on upgrading Crowd please review the Upgrade Guide

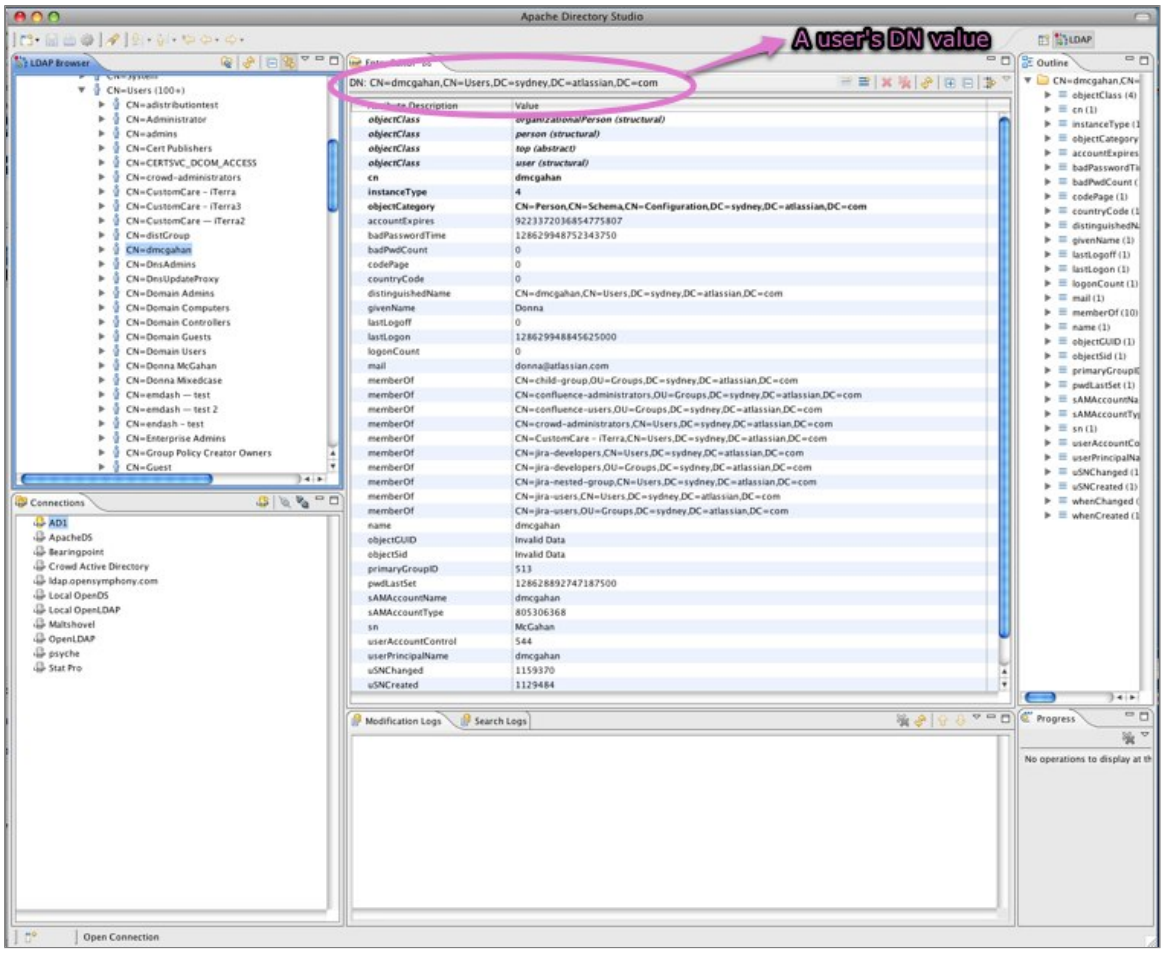
On this page:

- Example 1. Using a User's DN for Crowd Configuration
- Example 2: Using a Group's DN for Crowd Configuration

Example 1. Using a User's DN for Crowd Configuration

1. Find a user in the scope you wish to use for Crowd. Highlight that user in Apache Directory Studio.

   Screenshot: User information in Apache Directory Studio

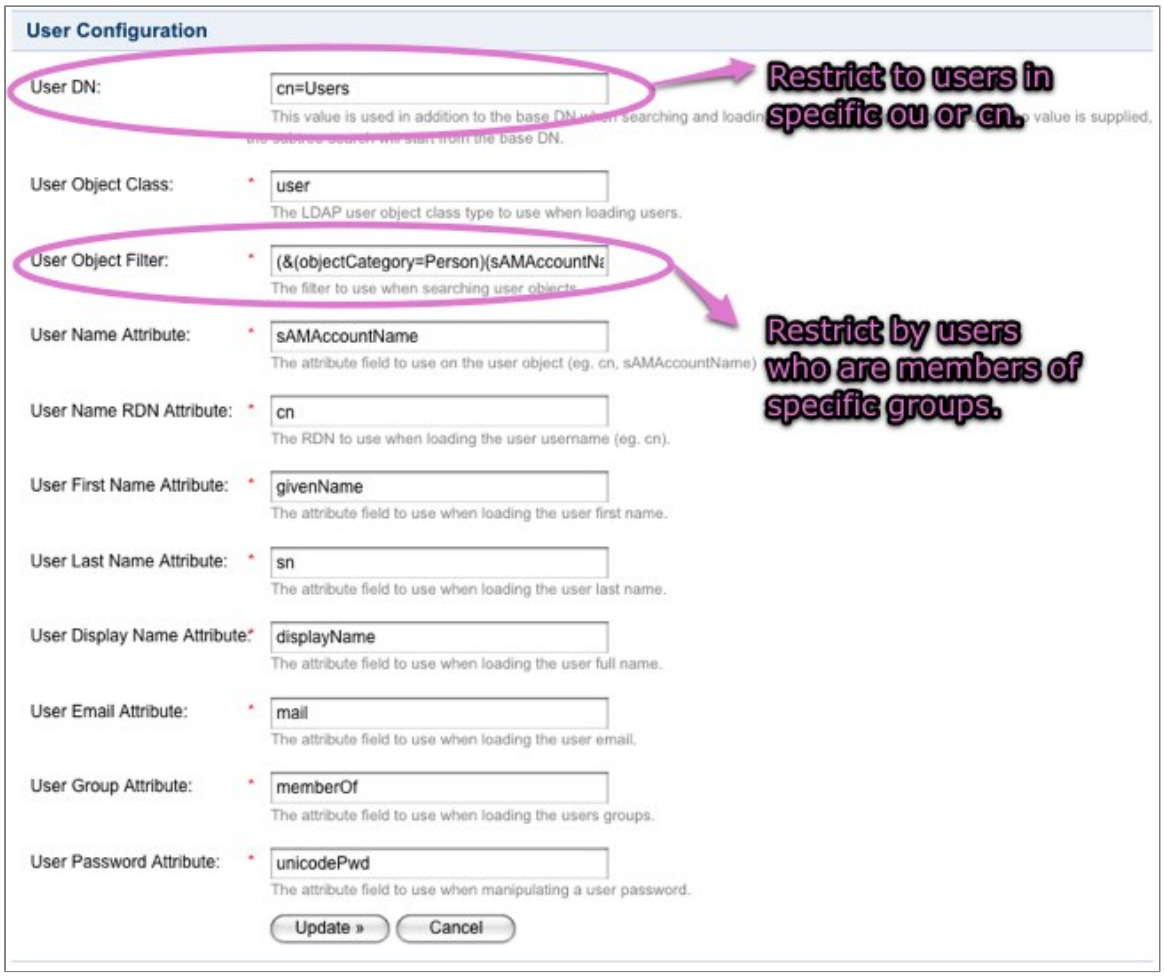


2. Using the information about the user *dmcgahan*, you can narrow down the users returned in the Crowd directory to those in *cn=Users* who are members of either the *confluence-users* or the *confluence-administrators* group.

User DN:	cn=Users

User Object Filter:	`(&(objectCategory=Person)(sAMAccountName=*)`
	`(|(memberOf=cn=confluence-users,ou=Groups,dc=sydney,dc=atlassian,dc=com)`
	`(memberOf=cn=confluence-administrators,ou=Groups,dc=sydney,dc=atlassian,dc=com)))`

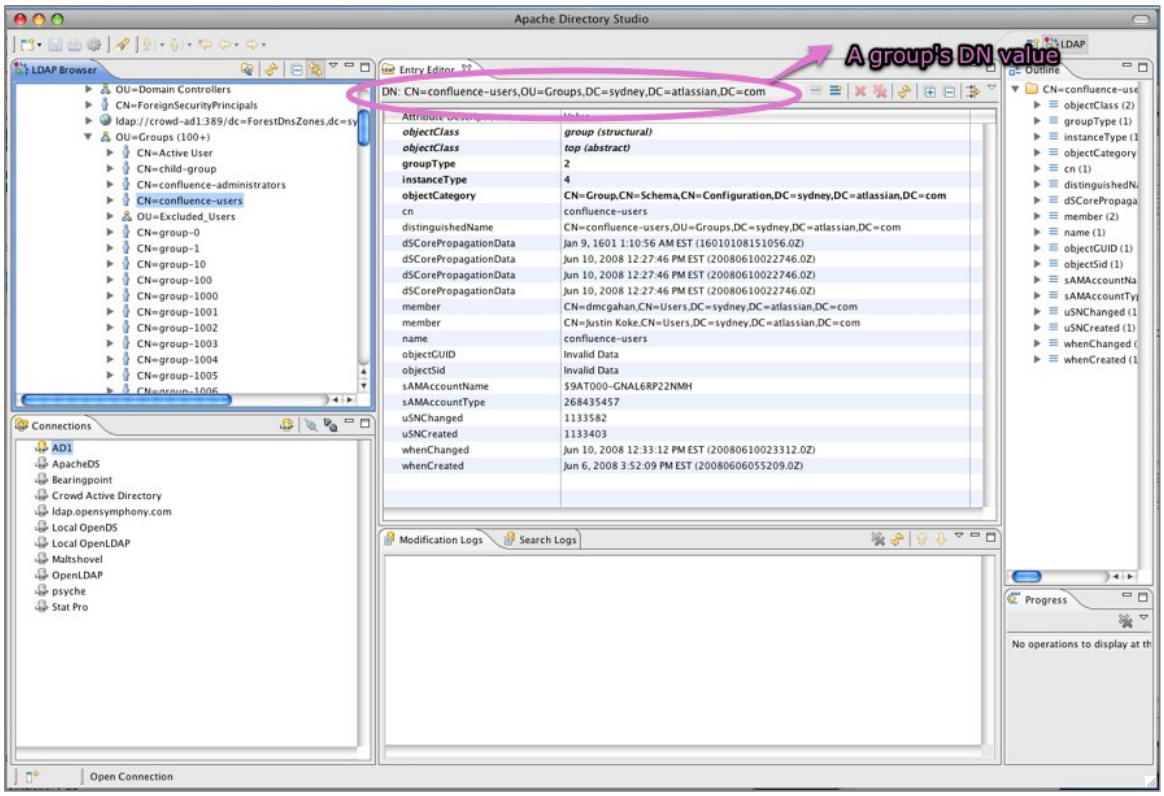
Screenshot: The resulting user configuration in Crowd



Example 2: Using a Group's DN for Crowd Configuration

1. Find a group in the scope you wish to use for Crowd. Highlight that group in Apache Directory Studio.

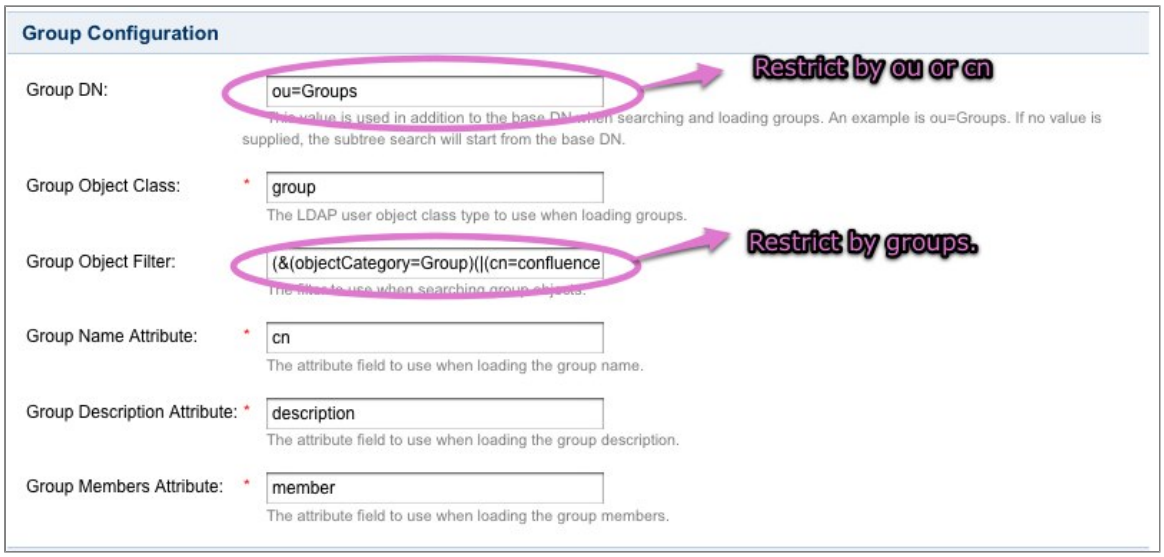
   Screenshot: Group information in Apache Directory Studio



2. Using the information about the group *confluence-users*, you can narrow down the groups returned in the Crowd directory to those in *ou=Groups* and return only the *confluence-users* or the *confluence-administrators* group. Under most circumstances, it is best to apply any changes to both group and role configuration for consistency.

Group DN:	ou=Groups
Group Object Filter:	`(&(objectCategory=Group)(|(cn=confluence-users)(cn=confluence-administrators)))`

Screenshot: The resulting group/role configuration in Crowd



RELATED TOPICS

Using Apache Directory Studio for Crowd LDAP Configuration

Integration FAQ

- All Integrations
  - If I delete a user from Crowd, how will this affect integrated applications?
  - Passing the crowd.properties File as an Environment Variable
- Atlassian Product Integration
  - Application Caching
  - JIRA integration
  - Public Signup Setup
- IBM Lotus Domino Integration
- IBM Websphere Integration

All Integrations

- If I delete a user from Crowd, how will this affect integrated applications?
- Passing the crowd.properties File as an Environment Variable

If I delete a user from Crowd, how will this affect integrated applications?

We recommend that you **deactivate** a user rather than deleting them, in case some applications contain historical data, e.g. documents that the user has created.

For example, a user may be a participant in a JIRA issue. If you remove the user from the directory managed by Crowd, JIRA will not be able to find the user details when referencing the issue. If you do need to remove the user from Crowd, you must first remove the user's involvement in any JIRA issues, as described in the JIRA documentation.

Read more about deleting or deactivating users in Crowd.

Passing the crowd.properties File as an Environment Variable

When integrating a client application with Crowd, you need a `crowd.properties` file containing configuration details for that application. (See Important Directories and Files.)

You can pass the location of a client application's `crowd.properties` file to the client application as an environment variable when starting the client application. This means that you can choose a suitable location for the `crowd.properties` file, instead of putting it in the client application's `WEB-INF/classes` directory.

This applies to the Crowd Administration Console's `crowd.properties` file too. You may find this particularly useful when integrating with a WAR deployment of an integrated application.

Example:

```
1.-Dcrowd.properties={FILE-PATH}/crowd.properties
```

Atlassian Product Integration

This section covers general questions around Crowd's integration with other Atlassian products.

General Integration Questions

Why don't my Groups and Users show up in Bamboo, Confluence, Fisheye or JIRA?

I want to allow public signups, but don't what 'public' users in my company LDAP repository. How should I configure Crowd?

Confluence Integration

JIRA Integration

What is the difference between JIRA's direct LDAP integration & Crowd's JIRA integration?
If I delete a user from Crowd, how will this affect JIRA?

Bamboo Integration

Fisheye Integration

Application Caching

When Crowd is deployed into Bamboo, Confluence, Fisheye or JIRA, the Crowd client may be using caching. If you notice that changes made in Crowd do not appear in one of Crowd's configured applications, this will most likely mean that the changes have not yet propagated into the client caches.

The Crowd development team has opened an improvement request (CWD-1283) for this issue. Please vote on this issue and add it to your JIRA watch list for future updates.

For more information, refer to:

- An overview of the different caching options in Crowd.
- Configuring caching for an application.
- Caching of user permissions on the Crowd server.
- Caching for LDAP directories.

JIRA integration

What is the difference between JIRA's LDAP integration and Crowd's JIRA integration?

JIRA's LDAP integration only delegates authentication to LDAP. This means that you still need to create groups and users in JIRA, and those users must have usernames that match your users in LDAP.

When you use Crowd's JIRA integration, all user and group management is delegated to Crowd. This means that you no longer have to create users and groups in JIRA. Crowd gives you access to all these users and groups in your underlying LDAP directories.

Public Signup Setup

This tip applies if you:

- Have public-facing JIRA, Confluence and Bamboo servers and private LDAP repositories.
- Allow public signup via JIRA, Confluence and/or Bamboo.
- Want to partition where users are created via the public signup functionality.

Crowd allows for multiple directories to be assigned to an application. Follow these steps to direct all public signups into your chosen Crowd directory:

1. Define two directories in Crowd:
    a. An internal directory for 'public' users.
    b. An LDAP directory for staff and contractors.
2. Assign both these directories to the 'JIRA' application in Crowd. (See Mapping a Directory to an Application.)
3. Use the 'ordering' arrows to move the internal 'public' directory into the first position. (See Specifying the Directory Order for an Application.)
4. Grant the 'Add User' permission to the 'JIRA' application in the internal 'public' directory. (See Specifying an Application's Directory Permissions.)
5. Ensure that the 'Add User' permission is disabled for the 'JIRA' application in the private LDAP directory.

Using this configuration, when Crowd receives a request from JIRA to create a user, Crowd will create the user in the 'public' internal directory only.

Unless otherwise instructed, Crowd will add the user to **all** directories assigned to the 'JIRA' application. The above steps allow you to ensure that the signed-up users are added to your 'public' directory only.

IBM Lotus Domino Integration

Customers have reported successful Crowd integration with IBM Lotus Domino. For more information, take a look at CWD-125.

The Atlassian Crowd team does not officially support this integration, because we do not have test environments set up for Lotus Domino.

IBM Websphere Integration

If your client application is running in Websphere, there is a known problem with Websphere's XML libraries.

Crowd uses XFire to handle the requests between the client application (JIRA, Confluence, Bamboo etc.) and Crowd, XFire requires a newer version of an XML library than what is shipped with Websphere 5.1.

More information and a link to a newer version of the relevant JAR file is available on the XFire website

You will need to add the **qname.jar** file to the `WebSphere\AppServer\lib` directory and remove the old file.

Some users have also reported errors like the following:

```
1.java.lang.VerifyError:
2.(class: org/codehaus/xfire/aegis/type/basic/ObjectType, method: writeSchema signature:
3.(Lorg/jdom/Element;)V) Incompatible argument to method
```

This is related to the following XFire issue the suggested fix for this is to upgrade the version of JDOM that is shipped with Websphere to something greater than 1.0 (Websphere ships with JDOM Beta 6).

If you add a later version of JDOM to the `WebSphere\AppServer\lib` directory and remove the old version, this should fix the above problem.

Support Policies

Welcome to the support policies index page. Here, you'll find information about how Atlassian Support can help you and how to get in touch with our helpful support engineers. Please choose the relevant page below to find out more.

- Bug Fixing Policy
- How to Report a Security Issue
- New Features Policy
- Patch Policy
- Security Advisory Publishing Policy
- Security Patch Policy
- Severity Levels for Security Issues

To request support from Atlassian, please raise a support issue in our online support system. To do this, visit support.atlassian.com, log in (creating an account if need be) and create an issue under Crowd. Our friendly support engineers will get right back to you with an answer.

Bug Fixing Policy

Summary

- Atlassian Support will help with workarounds and bug reporting.
- Critical bugs will generally be fixed in the next maintenance release.
- Non critical bugs will be scheduled according to a variety of considerations.

Raising a Bug Report

Atlassian Support is eager and happy to help verify bugs — we take pride in it! Please open a support request in our support system providing as much information as possible about how to replicate the problem you are experiencing. We will replicate the bug to verify, then lodge the report for you. We'll also try to construct workarounds if they're possible.

Customers and plugin developers are also welcome to open bug reports on our issue tracking systems directly. Use http://jira.atlassian.com for the stand-alone products and http://studio.atlassian.com for JIRA Studio.

When raising a new bug, you should rate the priority of a bug according to our JIRA usage guidelines. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

How Atlassian Approaches Bug Fixing

Maintenance (bug fix) releases come out more frequently than major releases and attempt to target the most critical bugs affecting our customers. The notation for a maintenance release is the final number in the version (ie the 1 in 3.0.1).

If a bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions) then it will be fixed in the next maintenance release provided that:

- The fix is technically feasible (i.e. it doesn't require a major architectural change).
- It does not impact the quality or integrity of a product.

For non-critical bugs, the developer assigned to fixing bugs prioritises the non-critical bug according to these factors:

- How many of our supported configurations are affected by the problem.
- Whether there is an effective workaround or patch.
- How difficult the issue is to fix.
- Whether many bugs in one area can be fixed at one time.

The developers responsible for bug fixing also monitor comments on existing bugs and new bugs submitted in JIRA, so you can provide feedback in this way. We give high priority consideration to security issues.

When considering the priority of a non-critical bug we try to determine a 'value' score for a bug which takes into account the severity of the bug from the customer's perspective, how prevalent the bug is and whether roadmap features may render the bug obsolete. We combine this

with a complexity score (i.e. how difficult the bug is). These two dimensions are used when developers self serve from the bug pile.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

How to Report a Security Issue

Finding and Reporting a Security Vulnerability

If you find a security bug in the product, please open an issue on http://jira.atlassian.com in the relevant project.

- Set the priority of the bug to 'Blocker'.
- Provide as much information on reproducing the bug as possible.
- Set the security level of the bug to 'Developer and Reporters only'.

All communication about the vulnerability should be performed through JIRA, so that Atlassian can keep track of the issue and get a patch out as soon as possible.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

New Features Policy

Summary

- We encourage and display customer comments and votes openly in our issue tracking systems, http://jira.atlassian.com and http://studio.atlassian.com.
- We do not publish roadmaps.
- Product Managers review our most popular voted issues on a regular basis.
- We schedule features based on a variety of factors.
- Our Atlassian Bug Fixing Policy is distinct from our Feature Request process.
- Atlassian provides consistent updates on the top 20 feature/improvement requests (in our issue tracker systems).

How to Track what Features are Being Implemented

When a new feature or improvement is scheduled, the 'fix-for' version will be indicated in the JIRA issue. This happens for the upcoming release only. We maintain roadmaps for more distant releases internally, but because these roadmaps are often pre-empted by changing customer demands, we do not publish them.

How Atlassian Chooses What to Implement

In every major release we *aim* to implement highly requested features, but it is not the only determining factor. Other factors include:

- **Direct feedback** from face to face meetings with customers, and through our support and sales channels.
- **Availability of staff** to implement features.
- **Impact** of the proposed changes on the application and its underlying architecture.
- How **well defined** the requested feature is (some issues gain in popularity rapidly, allowing little time to plan their implementation).
- Our long-term **strategic vision** for the product.

How to Contribute to Feature Development

Influencing Atlassian's release cycle
We encourage our customers to vote on feature requests in JIRA. The current tally of votes is available online in our issue tracking systems, http://jira.atlassian.com and http://studio.atlassian.com. Find out if your improvement request already exists. If it does, please vote for it. If you do not find it, create a new feature or improvement request online.

Extending Atlassian Products
Atlassian products have powerful and flexible extension APIs. If you would like to see a particular feature implemented, it may be possible to develop the feature as a plugin. Documentation regarding the plugin APIs is available. Advice on extending either product may be available on the user mailing-lists, or at our community forums.

If you require significant customisations, you may wish to get in touch with our partners. They specialise in extending Atlassian products and can do this work for you. If you are interested, please contact us.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Patch Policy

Unable to render {include} Couldn't find a page to include called: Support:Atlassian Patch Policy

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Advisory Publishing Policy

Publication of Security Advisories

When a security issue in an Atlassian product is discovered and resolved, Atlassian will inform customers through the following mechanisms:

- A security advisory will be posted in the documentation.
- A copy of the advisory will be sent to the product mailing-lists. These lists are mirrored on our forums.
- If the person who reported the issue wants to publish an advisory through some other agency (for example, CERT), Atlassian will assist in the production of that advisory, and link to it from our own.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Security Patch Policy

Unable to render {include} Couldn't find a page to include called: Support:Security Patch Policy

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Severity Levels for Security Issues

Severity Levels

Atlassian security advisories include a severity level, rating the vulnerability as one of the following:

- Critical
- High
- Moderate
- Low

Below is a summary of the factors which we use to decide on the severity level, and the implications for your installation.

Severity Level: Critical

We classify a vulnerability as critical if most or all of the following are true:

- Exploitation of the vulnerability results in root-level compromise of servers or infrastructure devices.
- The information required in order to exploit the vulnerability, such as example code, is widely available to attackers.
- Exploitation is usually straightforward, in the sense that the attacker does not need any special authentication credentials or knowledge about individual victims, and does not need to persuade a target user, for example via social engineering, into performing any special functions.

Severity Level: High

We give a high severity level to those vulnerabilities which have the potential to become critical, but have one or more mitigating factors that make exploitation less attractive to attackers.

For example, given a vulnerability which has many characteristics of the critical severity level, we would give it a level of high if any of the following are true:

- The vulnerability is difficult to exploit.
- Exploitation does not result in elevated privileges.
- The pool of potential victims is very small.

Note: If the mitigating factor arises from a lack of technical details, the severity level would be elevated to critical if those details later became available. If your installation is mission-critical, you may want to treat this as a critical vulnerability.

Severity Level: Moderate

We give a moderate severity level to those vulnerabilities where the scales are slightly tipped in favour of the potential victim.

The following vulnerabilities are typically rated moderate:

- Denial of service vulnerabilities, since they do not result in compromise of a target.
- Exploits that require an attacker to reside on the same local network as the victim.
- Vulnerabilities that affect only nonstandard configurations or obscure applications.
- Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics.
- Vulnerabilities where exploitation provides only very limited access.

Severity Level: Low

We give a low severity level to those vulnerabilities which by themselves have typically very little impact on an organisation's infrastructure.

Exploitation of such vulnerabilities usually requires local or physical system access. Exploitation may result in client-side privacy or denial of service issues and leakage of information about organisational structure, system configuration and versions, or network topology.

Original ranking compiled by the SANS Institute
Our vulnerability ranking is based on a scale originally published by the SANS Institute.

Further reading

See How to Get Legendary Support from Atlassian for more support-related information.

Troubleshooting

- Finding Known Issues
- Characters in User or Group DN's that will cause problems when using Crowd
- Problems when Importing Users into MySQL
- Troubleshooting LDAP Error Codes
  - Active Directory LDAP Errors
- Troubleshooting SSL certificates and Crowd
- How to Optimise Crowd Client Caching
- Troubleshooting Crowd Performance
- Troubleshooting SSO with Crowd
  - Debugging SSO in environments with Proxy Servers
- Troubleshooting CrowdID

- Troubleshooting your Configuration on Setup

Finding Known Issues

We track the feature requests and bug reports in the Crowd project on our JIRA site. To find a known issue:

1. Browse the list of unresolved bugs and requests.
2. Click the '**Edit**' button on the left.
3. Under '**Text Search**', type keywords for your problem into the '**Query**' field.
4. Click '**View**' and browse the summaries of the unresolved issues.
5. Click an issue key to view the details of the issue and any fixes or workarounds.

Characters in User or Group DN's that will cause problems when using Crowd

At present, the `AbstractEncodingFilter` used by Crowd, JIRA and Confluence silently translates certain 'dangerous' characters. The `AbstractEncodingFilter` exists because Microsoft Word uses some special Unicode characters for text (e.g. curly quotes). Not all fonts on non-Windows systems contain these characters. This causes issues in JIRA and Confluence when users copy and paste text from Word into a page or issue. Users on non-Windows systems will see question marks or other odd characters if their fonts don't have these characters.

http://jira.atlassian.com/browse/CORE-100

Unfortunately, these translations obviously cause problems when querying for users or groups in Crowd which contain these characters.

http://jira.atlassian.com/browse/CWD-1152

Until we are able to resolve this issue, customers should be aware that user or group DN's that contain the following characters will not work in Crowd:

UTF-8

Decimal ASCII value	AbstractEncodingFilter Replacement Value	Description
183	"- "	Middle dot, Georgian comma, Greek middle dot
8211	"-"	En dash
8216	"'"	Left single quotation mark

8217	"'"	Right single quotation mark
8220	"\""	Left double quotation mark
8221	"\""	Right double quotation mark
8230	"..."	Horizontal ellipsis, three dot leader

ISO-8859-1

Decimal ASCII value	AbstractEncodingFilter Replacement Value	Description
133	"..."	Horizontal ellipsis, three dot leader
145	"'"	Left single quotation mark
146	"'"	Right single quotation mark
147	"\""	Left double quotation mark
148	"\""	Right double quotation mark
150	"-"	En dash

Problems when Importing Users into MySQL

If your Crowd installation is using a mySQL database, you may find that the user and group import process does not perform a complete import.

To solve this problem, please check the transaction level in your MySQL startup options, as defined in the `my.cnf` configuration file. See the Crowd MySQL configuration guide for instructions.

Troubleshooting LDAP Error Codes

Useful Links for translating LDAP Error codes:

- LDAP Error Codes
- How LDAP Error Codes Map to JNDI Exceptions
- Active Directory LDAP Errors
- Novell eDirectory or NDS Error Code List

Active Directory LDAP Errors

AD-specific errors appear after the word "data" and before "vece" or "v893" in the actual error string returned to the binding process*

525	user not found
52e	invalid credentials
530	not permitted to logon at this time
531	not permitted to logon at this workstation
532	password expired
533	account disabled
701	account expired
773	user must reset password
775	user account locked

*This information provided by the following IBM support document.

To enable LDAP logging on your AD server, please review this Microsoft guide.

Troubleshooting SSL certificates and Crowd

1. Ensure that you are not using any parameters in the `JAVA_OPTS` variable that refer to your keystore. For example,

```
-Djavax.net.ssl.trustStore="/my/key/store"
```

The JAVA_OPTS variable is normally located in the standalone version of Crowd's apache-tomcat/bin/setenv.sh or setenv.bat file (depending on the OS you are using). Remove these references and restart Crowd.

2. Run this command on the Crowd server, replacing <ip address of LDAP server> with your LDAP server's IP address:

```
openssl s_client -connect <ip address of LDAP server>:636
```

3. Save the certificate (including the BEGIN CERTIFICATE and END CERTIFICATE lines) of the response into a local file called tmp.pem.

4. Run this command on the local tmp.pem file. This should return an MD5 Fingerprint value.

```
openssl x509 -fingerprint -md5 -noout -in tmp.pem
```

5. Run this command on the Crowd server. This assumes you are using the default keystore and the $JAVA_HOME (or for Windows %JAVA_HOME%) variable has been set. If not, please specify the correct keystore path.

```
keytool -list -keystore $JAVA_HOME/jre/lib/security/cacerts
```

6. Ensure that the MD5 Fingerprint from step 3 is listed in your keystore. If it is not, you will need to import the tmp.pem certificate into your keystore.

For additional information on SSL services and a great testing tool called SSLPoke, please visit this guide. Although this guide was written for JIRA, it is still extremely useful for troubleshooting SSL-related Crowd issues.

If you continue to experience issues with your SSL configuration and Crowd, please open a new support issue. Attach the CROWD APPLICATION DIRECTORY/atlassian-crowd.log file and the output of the tests above to the support issue.

How to Optimise Crowd Client Caching

Crowd-integrated applications can store user, group and role data in a local cache. This helps improve the performance of Crowd since these applications do not have to repeatedly request information from Crowd. Generally, it is not necessary to configure application caching, although this depends on the size of your application deployments. But for larger installations, you may need to configure the application caching. Please refer to more information about:

- An overview of the different caching options in Crowd
- Configuring caching for an application.
- Troubleshooting the caching for Atlassian integrated applications.
- Caching of user permissions on the Crowd server.
- Caching for LDAP directories

Troubleshooting Crowd Performance

Please note:
This guide assumes you have already opened a Crowd support issue at http://support.atlassian.com and wish to provide additional information about your Crowd configuration in this issue.

1. The Crowd application is slow!

1. Ensure you are running the latest version of Crowd.
2. Under **Admin -> Logging & Profiling** in Crowd:
    - Change the com.atlassian.crowd package to DEBUG.
    - Enable profiling.
3. Replicate the performance issues you are seeing in Crowd (e.g. log out and log in, browse users, etc.)
4. Attach the resulting CROWD_DIRECTORY/atlassian-crowd.log file to your support ticket.
5. List the directories and applications active in your Crowd instance.
6. Provide rough estimates of the number of users and groups that are available in each LDAP directory configuration.
7. Provide information about the network location of any LDAP servers in respect to the Crowd server (e.g. same subnet, different networks, different states).
8. If using Active Directory, is SSL enabled?

2. JIRA/Confluence is slow!

1. Confirm that data caching is enabled in Crowd.
2. Confirm that the only crowd-integration-client JAR in the JIRA/Confluence WEB-INF/lib directory matches the version of

Crowd you are running (e.g. `crowd-integration-client-1.5.jar`).
3. Confirm that the `crowd-ehcache.xml` file located in the `JIRA/ConfluenceWEB-INF/classes` directory matches the one in the `CROWD/client/conf` directory.
4. If your Crowd installation contains more than 50,000 users, review the guide at Configuring Caching for an Application.

a. JIRA/Confluence still slow?

1. Stop JIRA/Confluence.
2. Temporarily replace the WEB-INF/lib/crowd-integration-client-1.x.JAR file with the appropriate version from this issue.
3. Restart JIRA/Confluence.
4. Under **Admin -> Logging & Profiling** in JIRA/Confluence:
    - Change the com.atlassian package to DEBUG.
    - Enable profiling.
5. Perform actions in JIRA/Confluence that are slow to respond (e.g. log out and log in, browse users, etc).
6. Attach the resulting `JIRA/Confluence logs/catalina.out` or `stdout.log`. If Confluence, also attach the `atlassian-confluence.log` file in the Confluence home directory (specified in the `confluence-init.properties` file at setup).
7. List the directories and applications active in your Crowd instance for the JIRA/Confluence application.
8. Provide rough estimates of the number of users and groups that are available in each LDAP directory configuration for the JIRA/Confluence application.
9. Provide information about the network location of any LDAP servers in respect to the Crowd server (e.g. same subnet, different networks, different states).

b. Using Active Directory?

1. Is SSL enabled?
2. Are you using nested groups (is the Use Nested Groups box checked in Crowd)?
3. If login is slow, please connect to your AD server using Apache Directory Studio and highlight the username used for this login. Provide a screenshot of this user — especially the list of `memberOf` attributes for this account (should contain full DNs).
4. Please also confirm that all domain controllers referenced in these groups are resolvable/reachable from the Crowd server using ping:

```
ping ad1.mycompany.com
ping ad2.mycompany.au
```

RELATED TOPICS

- Overview of Caching
- Configuring Caching for an Application
- Authorisation Caching
- Configuring Caching for an LDAP Directory

Troubleshooting SSO with Crowd

Please follow the steps below to troubleshoot problems with SSO (single sign-on) in Crowd:

1. Ensure that each application is using the same version of the crowd-integration-client JAR file. For example, if you are using Crowd 1.4, the `crowd-integration-client-1.4.jar` file should be located in the `WEB-INF/lib` directory of each Crowd-integrated application. For more information, please review this Knowledge Base article.

2. Confirm that you can log in to each application with the same username and password.
    - In Crowd, click '**Applications**' to view the Application Browser.
    - Click '**View**' next to the application.
    - Click the '**Authentication Test**' tab and follow these instructions.

3. Set each application to use centralised SSO authentication, as follows. Ensure that each Atlassian application's `WEB-INF/classes/seraph-config.xml` file is using the Crowd's com.atlassian.crowd authenticator class. For example in JIRA, instead of this:

    ```
    1.<authenticator class="com.atlassian.seraph.auth.DefaultAuthenticator"/>
    ```

    you should have this:

    ```
    1.<authenticator class="com.atlassian.crowd.integration.seraph.JIRAAuthenticator"/>
    ```

    Please, see our Adding an Application Tutorial page to check the SSO authenticator classes for other applications.

4. Once each application is using centralised authentication, confirm you can log in to each application with the same username and password.

5. Ensure that each application is using the same sub-domain. For example:
    - **JIRA** -> jira.example.com
    - **Confluence** -> confluence.example.com

- **Crowd** -> crowd.example.com

SSO will only work with applications on the same sub-domain. Why? Crowd uses a cookie to manage SSO and your browser only has access to cookies in the same sub domain, e.g. *.example.com.

This is the value that you set in the Domain property (e.g. .example.com) for Crowd to enable SSO. This is covered in the documentation on configuring the domain.

Still having trouble?

If the above steps have not solved your problem, please gather some debugging information as described below before contacting Atlassian support:

1. In Crowd, go to '**Administration**' -> '**Logging & Profiling**'. Change the `com.atlassian.crowd` package to DEBUG.
2. Replicate the SSO problem you are having.
3. Please raise a support issue on our Support System, attaching your `{CROWD}/atlassian-crowd.log` file with the debug information gathered.

RELATED TOPICS

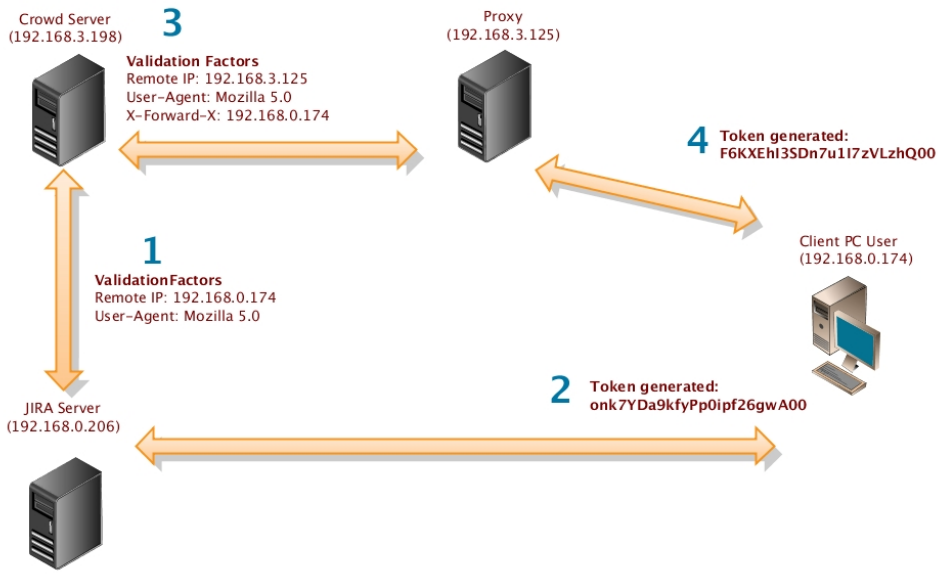
Overview of SSO

Debugging SSO in environments with Proxy Servers

This is an example log file from Crowd 1.6 with Debugging turned On for *com.atlassian.crowd* under Admin > Logging & Profiling. In this example, I've logged into Crowd Console, then attempt to access JIRA.

Example of non-working SSO Configuration

In this example, **admin** signs into Crowd Console, and then visits JIRA. JIRA is being served behind a Apache proxy (mod_proxy for example).



Login to Crowd directly without a proxy
Crowd detects a user logging in for the first time from the IP address 192.168.0.174, with a Mozilla Browser on Linux. A token of **onk7YDa9k** **ValidationFactors** consisting of an IP address, User-Agent, Random Number.

[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Generating Token for principal: admin
[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Adding User-Agent of com.atlassian.crowd.integration.authentication.ValidationFac
Linux i686; en-US; rv:1.8.0.9) Gecko/20070316 CentOS/1.5.0.9-10.el5.centos Firefox/1.5.0.9 pango-text]
[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Adding remote address of **192.168.0.174**
[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Adding **Random-Number** of
com.atlassian.crowd.integration.authentication.ValidationFactor@d171c2[name=Random-Number,value=8162711822532519761]
[crowd.manager.application.ApplicationServiceGeneric] Current Validation Factors:
com.atlassian.crowd.integration.authentication.ValidationFactor@83b064[name=remote_address,value=192.168.0.174]com.atlassian.crowd
User-Agent,value=Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.9) Gecko/20070316 CentOS/1.5.0.9-10.el5.centos Firefox/1.5.0.9 pango-
[crowd.manager.application.ApplicationServiceGeneric] comparing existing token
com.atlassian.crowd.model.token.Token@ca8293[ID=524387,key=onk7YDa9kfyPp0ipf26gwA00,name=admin,secretNumber=81627118225
com.atlassian.crowd.model.token.Token@b1b30c[ID=0,key=**onk7YDa9kfyPp0ipf26gwA00**,name=admin,secretNumber=816271182253251
[crowd.manager.application.ApplicationServiceGeneric] they match
[crowd.manager.application.ApplicationServiceGeneric] user has access to the application crowd

Login to JIRA via proxy

After my visit to the Crowd Console, I then visit JIRA through a proxy. It detects my same User-Agent, but now sees that my IP is **192.168.3.**
doesn't match my existing one: **F6KXEhI3SDn7u1I7zVLzhQ00** as compared to **onk7YDa9kfyPp0ipf26gwA00** and thus, I'm prompted to log
X-Forwarded-For header. It also contains my real IP. The way to fix this is to add **192.168.3.125** to my list of Trusted Proxies.

[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Generating Token for principal: admin
[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Adding **User-Agent** of com.atlassian.crowd.integration.authentication.ValidationFa
Linux i686; en-US; rv:1.8.0.9) Gecko/20070316 CentOS/1.5.0.9-10.el5.centos Firefox/1.5.0.9 pango-text]
[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Adding remote address of **192.168.3.125**
[atlassian.crowd.authentication.TokenKeyGeneratorImpl] Adding Random-Number of
com.atlassian.crowd.integration.authentication.ValidationFactor@1a99a7[name=Random-Number,value=8162711822532519761]
[crowd.manager.application.ApplicationServiceGeneric] Current Validation Factors:
com.atlassian.crowd.integration.authentication.ValidationFactor@5db889[name=remote_address,value=192.168.3.125]com.atlassian.crowd
X-Forwarded-For,value=192.168.0.174]com.atlassian.crowd.integration.authentication.ValidationFactor@31f633[name=User-Agent,value=
Gecko/20070316 CentOS/1.5.0.9-10.el5.centos Firefox/1.5.0.9 pango-text]
[crowd.manager.application.ApplicationServiceGeneric] comparing existing token
com.atlassian.crowd.model.token.Token@417bf8[ID=524387,key=onk7YDa9kfyPp0ipf26gwA00,name=admin,secretNumber=816271182253
com.atlassian.crowd.model.token.Token@f9d0af[ID=0,key=F6KXEhI3SDn7u1I7zVLzhQ00,name=admin,secretNumber=8162711822532519
[crowd.manager.application.ApplicationServiceGeneric] The token keys don't match

Troubleshooting CrowdID

If you are experiencing issues with Crowd's OpenID server (CrowdID), please take the following steps to help diagnose the problem:

Step 1: Change the logging for Crowd's OpenID server and client.

- Change the openid package from INFO to DEBUG in `CROWD/crowd-openidserver-webapp/WEB-INF/classes/log4j.properties`

```
01.#####################################################
02.# CROWD - CLASS-SPECIFIC LOGGING LEVELS
03.#####################################################
04.
05.# Set the following lines to DEBUG to enable logging on incoming, outgoing and fault SOAP
messages
06.
log4j.logger.com.atlassian.crowd.integration.service.soap.xfire.XFireInLoggingMethodHandler=WA
07.
log4j.logger.com.atlassian.crowd.integration.service.soap.xfire.XFireOutLoggingMethodHandler=W
08.
log4j.logger.com.atlassian.crowd.integration.service.soap.xfire.XFireFaultLoggingMethodHandler=
09.
10.log4j.logger.com.atlassian.crowd.openid=DEBUG
11.
12.log4j.logger.com.atlassian.crowd=INFO
```

- Change the openid package from INFO to DEBUG in `CROWD/crowd-openidclient-webapp/WEB-INF/classes/log4j.properties`

```
1.#####################################################
2.# CROWD - CLASS-SPECIFIC LOGGING LEVELS
3.#####################################################
4.
5.log4j.logger.com.atlassian.crowd.openid=DEBUG
6.
7.log4j.logger.com.atlassian.crowd=INFO
```

Step 2: Test CrowdID with the bundled OpenID client:

- `http://<your Crowd URL>:<Crowd port>/openidclient/`

If these tests are not successful, attach the `atlassian-crowd-openid-client.log` and `atlassian-crowd.openid-server.log` files (in the same location specified by this guide) to a support issue at http://support.atlassian.com. Note the username of the account tested.

Step 3: Test CrowdID with your OpenID application:

If these tests are not successful, attach the `atlassian-crowd-openid-client.log` and `atlassian-crowd.openid-server.log` files (in the same location specified by this guide) to a support issue at http://support.atlassian.com. Note the username of the account tested and the OpenID application you are attempting to use.

Crowd Resources

Resources for Evaluators

- Free Trial
- Feature Tour

Resources for Administrators

- Crowd Knowledge Base
- Tips of the Trade
- Guide to Installing an Atlassian Integrated Suite

Downloadable Documentation

- Crowd documentation in PDF, HTML or XML formats

Plugins and Extensions

- Atlassian Plugin Exchange

Support

- Atlassian Support
- Support Policies

Forums

- Crowd Announcements | subscribe
- Crowd General Forum | subscribe
- Crowd Developers Forum | subscribe

Feature Requests

- Issue Tracker and Feature Requests for Crowd

Tips of the Trade

Below are some links to external blog posts and articles containing technical tips and instructions on setting up and using Crowd. This page presents an opportunity for customers and community authors to share information and experiences.

The references here are specific to Crowd and are technical 'how to' guides written by bloggers who use Crowd. For more general information on identity management solutions, best practices and business cases, please refer to the Atlassian website.

Please be aware that these are external blogs and articles.
Most of the links point to external sites, and some of the information is relevant to a specific release of Crowd. Atlassian provides these links because the information is useful and relevant at the time it was written. Please check carefully whether the information is still relevant when you read it, and whether it is relevant to your version of Crowd. Unless explicitly stated, Atlassian does not offer support for third-party extensions or plugins. The information in the linked blog posts has not been tested or reviewed by Atlassian. We recommend that you test all solutions on a **test** server before trying it on your production site.

On this page:

- Three's a Crowd - securing a Grails application with Acegi and Crowd
- SSO for RoundCube Webmail with Atlassian Crowd
- Nexus Crowd Plugin Introduction
- Install Crowd Apache2 Module
- Bulk User Management with Crowd's Remote API
- Crowd Caching in 1.6

- Hammering Crowd

Application Connectors

Three's a Crowd - securing a Grails application with Acegi and Crowd

- By: Kate Ellingburg, on the 'Atlassian Blog'
- About: How to get Grails, Acegi and Crowd going together
- Date and Crowd version: 4 March 2008; Crowd 1.3
- Related documentation: Integrating Crowd with Spring Security

SSO for RoundCube Webmail with Atlassian Crowd

- By: Stefan Reuter, on the 'Stefan Reuter' blog
- About: Integrating a webmail system (RoundCube Webmail 0.2.2) with Crowd
- Date and Crowd version: 24 June 2009; Crowd 1.6
- Related documentation: Creating a Crowd Client for your Custom Application

Nexus Crowd Plugin Introduction

- By: Justin Edelson, on the 'Sonatype Blog'
- About: Using Crowd with Sonatype Nexus, via a new plugin for Nexus
- Date and Crowd version: 28 February 2009; Crowd 1.6
- Related documentation: Creating a Crowd Client for your Custom Application

Install Crowd Apache2 Module

- By: Scott Herdman, on blog 'swherdman.com'
- About: Integrating Crowd with Apache on Debian
- Date and Crowd version: 22 July 2009
- Related documentation: Integrating Crowd with Apache

Remote API

Bulk User Management with Crowd's Remote API

- By: Andreas Knecht, on the 'Atlassian Blog'
- About: Adding multiple users to a group in Crowd, using Crowd's remote API and Ruby
- Date and Crowd version: 11 September 2008; Crowd 1.5
- Related documentation: Managing Group Members

Performance and Load Testing

Crowd Caching in 1.6

- By: Shihab Hamid, on the 'Atlassian Blog'
- About: Caching in Crowd 1.6
- Date and Crowd version: 4 January 2009; Crowd 1.6
- Related documentation:
  - Overview of Caching
  - Configuring Caching for an LDAP Directory

Hammering Crowd

- By: Shihab Hamid, on the 'Atlassian Blog'
- About: Tips for and finding from Crowd performance and load testing
- Date and Crowd version: 30 March 2008; Crowd 1.3
- Related documentation:
  - Overview of Caching
  - Performance Profiling
  - Troubleshooting Crowd Performance

Have you written a technical tip for Crowd?
Add a comment to this page, linking to your blog post or article. We will include it if the content fits the requirements of this page.

Feedback?
Your first port of call should be the author of the linked blog post. If you want to let us know how useful (or otherwise) a linked post is, please add a comment to this page.

Other Sources of Information

Crowd documentation
Atlassian website
Atlassian forums
Atlassian Blog
Crowd plugins