Documentation for Crowd 2.11
Crowd Installation and Upgrade Guide

Configuring Options for an Application ........................................ 198
Managing Users and Groups ..................................................... 199
Using the User Browser ............................................................ 199
Adding a User ........................................................................ 200
Editing a User’s Details and Password ......................................... 201
Deleting or Deactivating a User ................................................ 202
Case Sensitivity of Usernames and Groups ................................. 204
Specifying a User’s Aliases ....................................................... 205
Editing a User’s Group Membership ........................................... 207
Managing Groups .................................................................... 208
Deleting a Group ..................................................................... 209
Adding a Group ....................................................................... 210
Managing Group Members ....................................................... 211
  Automatically Assigning Users to Groups ................................. 214
  Adding Users to a Group ........................................................ 215
  Removing Users from a Group ............................................... 218
Nested Groups in Crowd ............................................................ 221
Adding a Sub-Group .................................................................. 224
Removing a Sub-Group .............................................................. 226
Specifying a User’s Attributes ................................................... 228
Granting Crowd Administration Rights to a User ....................... 229
Granting Crowd User Rights to a User ....................................... 230
Managing a User’s Session .......................................................... 231

System Administration ................................................................. 232
Configuring Server Settings ....................................................... 233
  Deployment Title .................................................................. 233
  Domain ............................................................................... 234
  Session Configuration .......................................................... 236
  Authorization Caching .......................................................... 238
  Licensing ............................................................................ 240
  Finding Your Crowd Support Entitlement Number (SEN) ....... 242
SSO Cookie ............................................................................... 243
Configuring your Mail Server .................................................... 244
Creating an Email Notification Template .................................. 247
Configuring Trusted Proxy Servers ............................................ 250
Viewing Crowd’s System Information ........................................ 251
Backing Up and Restoring Data ................................................ 253
Logging and Profiling ............................................................... 254
  Performance Profiling .......................................................... 259
Configuring the LDAP Connection Pool .................................... 259
Overview of Caching ............................................................... 262
Crowd Release Notes ................................................................. 263
Crowd Release Summary ........................................................... 264
Crowd 2.11.0 Release Notes ...................................................... 266
Crowd 2.10.1 Release Notes ...................................................... 270
Crowd 2.9.5 Release Notes ....................................................... 272
Crowd 2.9.1 Release Notes ....................................................... 273
Installing Crowd ....................................................................... 274
Supported Platforms ................................................................ 275
  Setting JAVA_HOME ............................................................ 277
Installing Crowd and CrowdID ................................................... 278
  Connecting Crowd to a Database ......................................... 280
  Connecting CrowdID to a Database ..................................... 286
Installing Crowd and CrowdID WAR Distribution .................... 296
Specifying your Crowd Home Directory .................................... 301
Running the Setup Wizard ......................................................... 302
Troubleshooting your Configuration on Setup ......................... 311
Configuring Crowd ................................................................... 313
  Important Directories and Files ............................................ 313
  Changing the Port that Crowd uses ....................................... 320
  Configuring Crowd to Work with SSL ................................. 321

Creative Commons Attribution 2.5 Australia License
Troubleshooting SSL certificates and Crowd ........................................... 437
How to Optimize Crowd Client Caching ................................................. 438
Troubleshooting Crowd Performance ....................................................... 438
Troubleshooting SSO with Crowd .............................................................. 439
  Debugging SSO in environments with Proxy Servers .......................... 440
Troubleshooting CrowdID ......................................................................... 442
Provide Crowd Information to Atlassian Support .................................... 443
Contributing to the Crowd Documentation .............................................. 443
Tips of the Trade ...................................................................................... 444
Crowd Documentation in Other Languages ............................................. 446
Crowd Documentation

Crowd 2.11

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 community answers 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 developer documentation and the API documentation. You may also like to see answers from the community.

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

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 a JIRA application and/or Confluence and now wish to set up Crowd for user management in one or both of them.

- If you want to use a JIRA application or Confluence but have not yet installed them, please follow the instructions in Installing JIRA applications and/or Confluence 101 before configuring Crowd.
- If you want to use Crowd with other applications but not a JIRA application 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 ZIP archive file for the Crowd distribution (not EAR-WAR).
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 application 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 TAR.GZ archive file for the Crowd distribution (not EAR-WAR).
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 TAR.GZ Archive file for the Crowd distribution (not EAR-WAR).
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 application 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 and Confluence users are currently all managed via JIRA's internal management, you can use Crowd to simplify and centralize 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 centralize 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
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 flavors, 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 flavors, 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:
   a. **Import your users and groups** into Crowd.
   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.
   a. Choose 'Confluence' as the application type.
   b. In the 'Directories' step, choose the user directory you added for Confluence.
   c. In the 'Authorization' 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.
   a. Choose 'JIRA' as the application type.
   b. In the 'Directories' step, choose the user directory you added for JIRA.
   c. In the 'Authorization' 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 authorized 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 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

---

*Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
• Managing Directories
  • Using the Directory Browser
  • Adding a Directory
    • Configuring an Internal Directory
    • Configuring an LDAP Directory Connector
      • LDAP Object Structures
      • Configuring an SSL Certificate for Microsoft Active Directory
    • Configuring a Remote Crowd Directory
    • Configuring a Custom Directory Connector
    • Configuring a Delegated Authentication Directory
  • Configuring Caching for an LDAP Directory
  • Using Naive DN Matching
  • 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 Confluence 3.4 or earlier
      • Updating Files in a Confluence Evaluation Distribution
    • Integrating Crowd with Atlassian CrowdID
    • Integrating Crowd with Atlassian Crucible
    • Integrating Crowd with Atlassian FishEye
      • Configuring FishEye earlier than 4.0 with Crowd
    • Integrating Crowd with Atlassian JIRA
    • Integrating Crowd with Atlassian JIRA 4.2 or earlier
    • Integrating Crowd with Atlassian Bitbucket Server
    • Integrating Crowd with Acegi Security
      • Integrating AppFuse - a Crowd-Acegi Integration Tutorial
    • Integrating Crowd with Apache
      • Disabling Previous Versions of the Crowd Apache Connector
      • Installing the Crowd Apache Connector on CentOS Linux
      • Installing the Crowd Apache Connector on Red Hat Enterprise Linux
      • Installing the Crowd Apache Connector on Ubuntu Linux
      • Installing the Crowd Apache Connector on Debian
      • Installing the Crowd Apache Connector on Other UNIX-Like Systems
      • Installing the Crowd Apache Connector on Windows
    • 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
    • Integrating Crowd with Atlassian HipChat
    • Integrating Crowd with Atlassian FishEye - DRAFT
  • 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
• Viewing Users in Directories Mapped to an Application
• Specifying which Groups can access an Application
• Effective memberships with multiple directories
• Specifying an Application's Address or Hostname
• Testing a User's Login to an Application
• Enforcing Lower-Case Usernames and Groups 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 and Groups
  • Using the User Browser
  • Adding a User
  • Editing a User's Details and Password
  • Deleting or Deactivating a User
  • Case Sensitivity of Usernames and Groups
  • Specifying a User's Aliases
  • Editing a User's Group Membership
  • Managing Groups
    • Deleting a Group
    • Adding a Group
  • Managing Group Members
    • Automatically Assigning 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
  • Specifying a User's Attributes
  • Granting Crowd Administration Rights to a User
  • Granting Crowd User Rights to a User
  • Managing a User's Session

• System Administration
  • Configuring Server Settings
    • Deployment Title
    • Domain
    • Session Configuration
    • Authorization Caching
    • Licensing
      • Finding Your Crowd Support Entitlement Number (SEN)
    • 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
  • Draft - Troubleshooting and Requesting Technical Support
  • Configuring the LDAP Connection Pool
  • Overview of Caching

• Crowd Security Advisories and Fixes
  • Crowd Security Advisory 2010-07-05
  • Crowd Security Advisory 2010-05-04
  • Crowd Security Advisory 2008-10-14 - Parameter Injection Vulnerability
  • Crowd Security Advisory 2012-05-17
  • Crowd Security Notice 2013-07-01
  • Crowd Security Advisory 2013-07-16
  • Crowd Security Advisory 2014-05-21
  • Crowd Security Advisory 2016-10-19
Getting Started

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

Concepts

Crowd is an application security framework that handles authentication and authorization 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 centralized 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 authorized 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 centralized identity management. Crowd works by dispatching authentication and authorization calls from configured applications to configured directories.

A typical deployment may be similar to the following:

![Diagram of Crowd Architecture](image)

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 management in LDAP.
• A Crowd internal directory for user/group 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 authorization requests to the directory, for all applications that are mapped to that directory. Modification of directory entities (users and groups) 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 Bitbucket Server
• Atlassian Bamboo
• Atlassian FishEye
• Atlassian Crucible
• Google Apps
• Apache
• Subversion
• Jive Forums
• Atlassian CrowdID
• Acegi
• Spring Security
• 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**

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 authorized Crowd administrators.

Authorized 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 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 and applications associated with your username. Refer to the Crowd User Guide for details.

To access the Crowd Administration 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.

Please refer to the link below in order to grant administrators rights to Crowd user(s):

**Granting Crowd Administration Rights to a User**

**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.
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 management in LDAP.
- A Crowd internal directory for user/group 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 authorization requests to the directory, for all applications that are mapped to that directory. Modification of directory entities (users and groups) 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

#### 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 management in LDAP.
- A Crowd internal directory for user/group 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 authorization requests to the directory, for all applications that are mapped to that directory. Modification of directory entities (users and groups) 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'**

<table>
<thead>
<tr>
<th>Name</th>
<th>Active</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd</td>
<td>true</td>
<td>Crowd Internal Directory</td>
<td>View</td>
</tr>
<tr>
<td>Employees</td>
<td>true</td>
<td>Crowd Internal Directory</td>
<td>View</td>
</tr>
</tbody>
</table>

**RELATED TOPICS**

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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 authorization 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 and groups in Crowd. You can use Crowd's flexible and simple group management when the LDAP groups do not suit your requirements. Alternatively, you can have Crowd import users' group memberships from LDAP each time they authenticate.

- **LDAP Directory Connector**
- **Remote Crowd Directory Connector**
  Remote Crowd directories allow Crowd to Crowd connections. In other words, one Crowd server can obtain users and groups from another Crowd server.
- **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)
   - 'Remote Crowd' — see Configuring a Remote Crowd 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.

**Related Topics**

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
• Configuring a Remote Crowd Directory
• Configuring a Custom Directory Connector
• Configuring a Delegated Authentication Directory
• Configuring Caching for an LDAP Directory
• Using Naive DN Matching
• 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
Create Internal Directory

<table>
<thead>
<tr>
<th>Details</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
</tr>
<tr>
<td>Active:</td>
<td></td>
</tr>
<tr>
<td>Password Regex:</td>
<td></td>
</tr>
<tr>
<td>Password Complexity Requirement Message:</td>
<td></td>
</tr>
<tr>
<td>Maximum Invalid Password Attempts:</td>
<td>0</td>
</tr>
<tr>
<td>Maximum Unchanged Password Days:</td>
<td>0</td>
</tr>
<tr>
<td>Password History Count:</td>
<td>0</td>
</tr>
<tr>
<td>Password Encryption:</td>
<td></td>
</tr>
<tr>
<td>Use Nested Groups:</td>
<td></td>
</tr>
</tbody>
</table>

Internal Directory Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>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.</td>
</tr>
<tr>
<td>Description</td>
<td>Details about this specific directory.</td>
</tr>
</tbody>
</table>
| Active      | Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications. If a directory is not marked as 'Active', it is inactive. Inactive directories:  
  - are not included when searching for users, groups or memberships.  
  - are still displayed in the Crowd Administration Console screens. |
| 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. |
<p>| Password Complexity Message | A message shown when a user is resetting a password to explain custom complexity requirements set with Password Regex (since Crowd 2.5.2). |</p>
<table>
<thead>
<tr>
<th>Table Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum Invalid Password Attempts</strong></td>
<td>The maximum number of invalid password attempts before the authenticating account will be disabled. Enter 0 to disable this feature.</td>
</tr>
<tr>
<td><strong>Maximum Unchanged Password Days</strong></td>
<td>The number of days until the password must be changed. This value is in days, enter 0 to disable this feature.</td>
</tr>
<tr>
<td><strong>Password History Count</strong></td>
<td>The number of previous passwords to prevent the user from using. Enter 0 to disable this feature.</td>
</tr>
<tr>
<td><strong>Password Encryption</strong></td>
<td>If you wish to import users into this directory from another Atlassian product, specify 'ATLAS SIAN-SECURITY' in order to ensure password compatibility.</td>
</tr>
<tr>
<td><strong>Use Nested Groups</strong></td>
<td>Enable or disable support for nested groups on the internal user directory.</td>
</tr>
</tbody>
</table>

**Next Step**

See **Specifying Directory Permissions**.

**RELATED TOPICS**

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
    - Configuring a Remote Crowd Directory
    - Configuring a Custom Directory Connector
    - Configuring a Delegated Authentication Directory
  - Configuring Caching for an LDAP Directory
  - Using Naive DN Matching
- 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:

- 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
1. Log in to the Crowd Administration Console.
2. Click the Directories link in the top navigation bar. The Directory Browser opens.
3. Click the Add Directory link. The ‘Select Directory Type’ screen opens.
4. Click the ‘Connector’ button. The ‘Create Directory Connector’ window opens.
5. Complete the configuration information required on each of the tabs to finish setting up the connector.

**General configuration notes**

- **By default, the Cache Enabled setting** on the ‘Details’ tab is selected. We recommend you leave this setting selected. For more information, see Configuring Caching for an LDAP Directory.
- **If you select the Manage Groups Locally setting** on the ‘Connector’ tab (available only if you've selected the Cache Enabled check box), new groups are created and updated in the Crowd database and not propagated to the LDAP server. Memberships of local groups are also stored locally. This makes it possible to augment the group structure with new groups even with a read-only LDAP server. When this option is enabled, only local groups can be created and updated, while groups synchronized from the remote directory cannot be locally modified.
- **If you select the Use the User Membership setting** on the ‘Connector’ tab, Crowd will use the group membership attribute on the user when it retrieves the members of a given group, which will result in a more efficient retrieval.
- **If you select the Use 'memberOf for Group Membership setting** on the ‘Connector’ tab, Crowd will use the ‘memberOf’ attribute when it retrieves the list of groups a user belongs to, which will result in a more efficient retrieval. If you don't select this setting, Crowd will use the members attribute on the group (‘member’ by default) for the search.
- Crowd will synchronize user renames made in the LDAP server, provided that the User Unique Identifier Attribute is set in the ‘Configuration’ tab. If this attribute is not set and a user is renamed in the LDAP server, Crowd will not be able to track the user's identity, and will delete the user with the old name and create a new user with the new name. Crowd does not support group renames.
- If the directory type you're using guarantees the format of DNs, we recommend selecting the Use Naive DN Matching setting on the ‘Connector’ tab to allow Crowd to do a direct, case-insensitive, string comparison when it compares DNs. This setting can significantly improve performance.
- **Specify the Username on the ‘Connector’ tab in the following format:** cn-administrator, cn=users, dc=ad, dc=acmecorp, dc=com.
- **If you specify the User Name RDN attribute**, the DN for each LDAP entry is composed of two parts: the RDN and the location within the RDN directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure.
- **By default the Synchronise group memberships when logging in option** is set to *For newly added users only*. This will synchronize group memberships for users who have been created in the LDAP directory, but not yet synchronized to Crowd. This is recommended for convenience, without sacrificing performance. Other options are to synchronize the memberships *Every time a user logs in*, which was the behaviour in Crowd 2.7, 2.8 and 2.9, and to *Never synchronise the memberships*, which was how Crowd behaved before version 2.7.
- **If you have successfully added your connector, but aren't able to see any data when you browse the LDAP directory, make sure that any non-standard object types and filters are configured correctly.**

By default, the *Active* setting on the ‘Details’ tab is selected. Only clear this setting if you want to prevent all users within the directory from accessing mapped applications. Inactive directories:

- Are not included when Crowd searches for users, groups, or memberships
- Still appear in the Crowd Administration Console screens

You can also configure site-wide LDAP connection pool settings. See Configuring the LDAP Connection Pool.

**Directory-specific configuration notes**

Apache Directory Server (ApacheDS)
There are two known issues with ApacheDS and Crowd:

- ApacheDS 1.0.2 does not support password resets without a restart. This is an ApacheDS limitation.
- 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.

Apple Open Directory

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

Fedora Directory Server

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

Microsoft Active Directory

- If you want to use a secure SSL connection, make sure you configure an SSL Certificate before enabling this setting.
- We recommend selecting the Enable Incremental Sync setting to allow Crowd to retrieve changes made after the last synchronization when possible.
- 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 LDAP structure of your server.
- If you want to use Crowd to add users or change passwords in Microsoft 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. Please read the instructions: Configuring an SSL Certificate for Microsoft Active Directory.
- Crowd will synchronize the user status with Active Directory. If a user account is disabled in Active Directory, the user will be deactivated in Crowd, and reciprocally, if a user is deactivated in Crowd, the user account will be disabled in Active Directory. To prevent this synchronization, use Manage User Status Locally in the 'Connector' tab.
- Users' primary groups in Active Directory will be displayed as regular memberships in Crowd. However, you will not be able to change or remove the user's primary group through Crowd's user interface.
- If you are using a single Active Directory domain, you should disable "Use node referrals" in the directory configuration. If you have a forest, you should read User lookups fail with 'PartialResultExceptions' due to Active Directory 'follow referrals' configuration and ensure your DNS server is configured appropriately.
- 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.

Posix Schema for LDAP or Open LDAP

- Currently, Crowd supports read-only access to the directory based on the Posix schema. You cannot add or update user details. 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 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.

**LDAP Object Structures**

The Crowd LDAP connectors assume that all container objects (groups) have the full DN to the associated member. Currently the membership attributes on a User object are not used by Crowd.

**Supported Object Types**

- groupOfUniqueNames
- inetorgperson
- posixGroup
- posixUser
- zimbraAccount

**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.

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.

**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 Active Directory Certificate Services
- Step 2. Obtain the Server Certificate
- Step 3. Import the Server Certificate

There's a Confluence SSL plugin that facilitates this process.

Updating user, group, and membership details in Active Directory requires that your Atlassian application be running in a JVM that trusts the AD server. To do this, we generate a certificate on the Active Directory server, then import it into Java's keystore.

**Prerequisites**

To generate a certificate, you need the following components installed on the Windows Domain Controller to which you're connecting.
<table>
<thead>
<tr>
<th>Required Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet Information Services (IIS)</td>
<td>This is required before you can install Windows Certificate Services.</td>
</tr>
<tr>
<td>Windows Certificate Services</td>
<td>This installs a certification authority (CA) which is used to issue certificates. Step 1, below, explains this process.</td>
</tr>
</tbody>
</table>

Step 1. Install the Active Directory Certificate Services

If Certificate Services are already installed, skip to step 2, below. The screenshots below are from Server 2008, but the process is similar for Server 2000 and 2003.

1. Log in to your Active Directory server as an administrator.
2. Click Start, point to Administrative Tools, and then click Server Manager.
3. In the Roles Summary section, click Add Roles.

5. On the Select Role Services page, select the Certification Authority check box, and then click Next.
6. On the **Specify Setup Type** page, click **Enterprise**, and then click **Next**.

7. On the **Specify CA Type** page, click **Root CA**, and then click **Next**.
8. On the Set Up Private Key and Configure Cryptography for CA pages, you can configure optional configuration settings, including cryptographic service providers. However, the default values should be fine. Click Next twice.
9. In the **Common name for this CA** box, type the common name of the CA, and then click **Next**.

10. On the **Set Validity Period** page, accept the default values or specify other storage locations for the
certificate database and the certificate database log, and then click **Next**.
11. After verifying the information on the **Confirm Installation Selections** page, click **Install**.

12. Review the information on the results screen to verify that the installation was successful.
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 application 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. For example: `c:\ad2008.ad01.atlassian.com_ad01.crt`.

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

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

You might still fail to be authenticated using the certificate file above. In this case, Microsoft’s [LDAP over SSL (LDAPS) Certificate](https://docs.microsoft.com/en-us/previous-versions/windows-server/ff738648(v=ws.10)) page might help. Note that you need to:

1. Choose “No, do not export the private key” in step-10 of Exporting the LDAPS Certificate and Importing for use with AD DS section
2. Choose “DER encoded binary X.509 (.CER)” in step-11 of Exporting the LDAPS Certificate and Importing for use with AD DS section. This file will be used in the following step.

Step 3. Import the Server Certificate

For an application 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 `jre\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.

Once the certificate has been imported as per the below instructions, you will need to restart the application to
pick up the changes.

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`.

   
   ```
   cd /d 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 -importcert -keystore .\jre\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
   Certificate fingerprints:
   Trust this certificate? [no]: yes
   Certificate was added to keystore
   ```

You may now change `URL` to use LDAP over SSL (i.e. ldaps://<HOSTNAME>:636/) and use the 'Secure SSL' option when connecting your application to your directory server.

UNIX

1. Navigate to the directory in which the Java used by JIRA is installed. If the default JAVA installation is used, then it would be

   ```
   cd $JAVA_HOME
   ```

2. Run the command below, where `server-certificate.crt` is the name of the file from your directory server:

   ```
   sudo keytool -importcert -keystore ./jre/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:
Trust this certificate? [no]: yes
Certificate was added to keystore

You may now change 'URL' to use LDAP over SSL (i.e. ldaps://<HOSTNAME>:636/) and use the 'Secure SSL' option when connecting your application to your directory server.

Mac OS X

1. Navigate to the directory in which Java is installed. This is usually

   cd /Library/Java/Home

2. Run the command below, where server-certificate.crt is the name of the file from your directory server:

   sudo keytool -importcert -keystore ./jre/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:
Trust this certificate? [no]: yes
Certificate was added to keystore

You may now change 'URL' to use LDAP over SSL (i.e. ldaps://<HOSTNAME>:636/) and use the 'Secure SSL' option when connecting your application to your directory server.

RELATED TOPICS

Configuring Crowd to Work with SSL
Configuring a Remote Crowd Directory
Remote Crowd directories allow Crowd to Crowd connections. In other words, one Crowd server can obtain users and groups from another Crowd server.

Two things need to be done in order to configure the local Crowd server to obtain users and groups from a remote Crowd server:

1. The local Crowd server needs to be given access to the remote Crowd server. This is achieved by adding a new application on the remote Crowd server.
2. The new Remote Crowd directory needs to be added to the local Crowd server with details on how to connect to the remote Crowd server.

In our local testing, we found that it took about 4 minutes to sync to external Crowd with 10 000 users, 1 000 groups, and 200 000 memberships.

Roles are not supported in Remote Crowd Directories.

To configure a Remote Crowd directory,

1. Log in to the Crowd Administration Console in the remote Crowd server.
2. Add a new application which will be the local Crowd server. The application name and password entered here will be used later by the local Crowd server to connect to the remote Crowd server.
3. Log in to the Crowd Administration Console in the local Crowd server.
4. Click the 'Directories' tab in the top navigation bar.
5. This will display the Directory Browser. Click 'Add Directory' in the left-hand menu.
6. Click the 'Remote Crowd' button.
7. The 'Details' tab will appear. Enter the 'Name' and 'Description' (see table of fields below).
8. We recommend that you leave 'Cache Enabled' at its default setting (enabled). Remote Crowd directory caching works the same way as it does for LDAP directories. For more information, see Configuring Caching for an LDAP Directory.
9. Click the 'Connection' tab. See screenshot 2 below.
10. Fill in the basic connection information for your remote Crowd server.
11. Click the 'Test Connection' button to verify that Crowd can successfully connect to the directory.
12. Click the 'Permissions' tab to configure the directory's permissions.

### Screenshot 1: Directory details

[Create Remote Crowd Directory]

- **Name:** Chicago Employees
- **Description:** Crowd Server with Chicago Employees
- **Active:**
- **Cache Enabled:**
- **Use Nested Groups:**

*Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>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'.</td>
</tr>
<tr>
<td>Description</td>
<td>Details about this specific directory.</td>
</tr>
<tr>
<td>Active</td>
<td>Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications. If a directory is not marked as 'Active', it is inactive. Inactive directories are:</td>
</tr>
<tr>
<td></td>
<td>• not included when searching for users, groups or memberships.</td>
</tr>
<tr>
<td></td>
<td>• still displayed in the Crowd Administration Console screens.</td>
</tr>
<tr>
<td>Cache Enabled</td>
<td>We recommend that you turn on caching. Remote Crowd directory caching works the same way as it does for LDAP directories. For more information, see Configuring Caching for an LDAP Directory.</td>
</tr>
<tr>
<td>Use Nested Groups</td>
<td>Enable or disable support for nested groups on the Remote Crowd directory.</td>
</tr>
</tbody>
</table>

Configuring Connection Details

Screenshot 2: Connection details
Create Remote Crowd Directory

**Attribute** | **Description**
--- | ---
URL | The connection URL to use when connecting to the directory server. The URL should be in the following format: `http://domainname:port/crowd`.
Application Name | Application name used to authenticate to the remote Crowd server.
Application Password | Application password used to authenticate to the remote Crowd server.
Connection Timeout | The time, in seconds, to wait for a connection to be established. If there is no connection within the specified time period, the connection attempt will be aborted. A value of 0 (zero) means there is no limit.
Max Connections | The maximum number of simultaneous connections to remote Crowd server.
Proxy Host | HTTP proxy server domain name. This field is required if the remote Crowd server is behind a HTTP proxy.
Proxy Port | HTTP proxy server port number. This field is required if the remote Crowd server is behind a HTTP proxy.
<table>
<thead>
<tr>
<th>Proxy Username</th>
<th>HTTP proxy server username. This field is required if the HTTP proxy server requires authentication.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy Password</td>
<td>HTTP proxy server password. This field is required if the HTTP proxy server requires authentication.</td>
</tr>
<tr>
<td>Enable Incremental Sync</td>
<td>We recommend that you turn on incremental synchronization. It will cause only changes since the last synchronization to be queried when synchronizing, thus significantly reducing synchronization resource usage.</td>
</tr>
<tr>
<td>Polling Interval</td>
<td>Crowd will synchronize with the remote Crowd server every x minutes, where ‘x’ is the number specified here. Remote Crowd directory caching works the same way as it does for LDAP directories, for which there is more information in Configuring Caching for an LDAP Directory.</td>
</tr>
</tbody>
</table>

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 the appropriate applications.

RELATED TOPICS
- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring a LDAP Directory Connector
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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 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'

<table>
<thead>
<tr>
<th>Custom Directory Store Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>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.</td>
</tr>
<tr>
<td>Description</td>
<td>Details about this specific directory.</td>
</tr>
</tbody>
</table>
| Active | Only deselect this if you wish to prevent all users within the directory from accessing all mapped applications. If a directory is not marked as ‘Active’, it is inactive. Inactive directories:  
- are not included when searching for users, groups or memberships.  
- are still displayed in the Crowd Administration Console screens. |
| Implementation Class | Implementation of com.atlassian.crowd.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 Caching for an LDAP Directory  
- Using Naive DN Matching
• Specifying Directory Permissions
• Importing Users and Groups into a Directory

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 and groups in Crowd. You can use Crowd's flexible and simple group management when the LDAP groups do not suit your requirements. Alternatively, you can have Crowd import users’ group memberships from LDAP each time they authenticate.

The username must be the same in the Crowd Delegated Authentication directory and in the LDAP directory. If a user is renamed in LDAP, Crowd will automatically rename the user in the Delegated Authentication directory.

Delegated Authentication directories do not allow you to browse the LDAP data. The directory delegates user authentication to LDAP, but to be able to list users and groups, you will need to add them to the directory. See more details in the Next Steps section of this page.

On this page:

• Example of using a Delegated Authentication Directory
• Configuration Steps
• Next Steps

Example of using a Delegated Authentication Directory

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).
Configuration Steps

Before setting up a new Delegated Authentication Directory, please review to the notes on LDAP object structures in the page about LDAP connectors.

To configure a Delegated Authentication directory:

1. Log in to the Crowd Administration Console.
2. Click the Directories link in the top navigation bar. The Directory Browser will open.
3. Click the Add Directory link and then select the Delegated Authentication button.
4. Complete the configuration information required on each of the tabs to finish setting up the directory.

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. If you have enabled the “Synchronize Group Memberships” option, groups and group memberships from LDAP will be automatically imported each time a user authenticates.

Next Steps

After configuring your new directory:
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 by enabling the Synchronize User Details option when you configure the directory.

RELATED TOPICS

- Using the Directory Browser
- Adding a Directory
  - Configuring an Internal Directory
  - Configuring an LDAP Directory Connector
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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

Configuring Caching for an LDAP Directory

Crowd manages a cache of LDAP directory information stored in the Crowd database, to ensure fast recurrent access to user and group data. We call this ‘database-backed LDAP caching’.

This page describes the caching of user and group information in the Crowd database. For a description 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
- Finding the Time Taken to Synchronize
- Manually Synchronizing the Cache
- Notes

Features of LDAP Caching in Crowd

For all LDAP directories with caching enabled, Crowd will keep an up-to-date cache of user and group information retrieved from the LDAP directory. Use of the cache should improve performance particularly in directories which are slow or off site.

Please refer to the notes below, especially regarding the number of users for which the caching is optimized.

Summary of the caching functionality:

- The caches are held in the Crowd database.
When you add the directory connector to Crowd, Crowd will start a synchronization task in the background to copy all the required users, groups and membership information from LDAP to the Crowd database. This task may take a while to complete, depending on the size and complexity of your user base.

Crowd will perform a periodic synchronization to update the database with any changes made to LDAP. The default sync interval, or polling interval, is one hour (60 minutes). You can change the polling interval on the directory connector configuration screen.

You can manually synchronize the database-backed cache if necessary. Whenever an update is made to the users, groups or membership information via Crowd, Crowd will update both the database-backed cache and the LDAP directory immediately.

For all authentication requests, Crowd performs calls to the LDAP directory itself. The Crowd database-backed cache does not store user passwords. Crowd performs all other queries against the database-backed cache.

The diagram below gives a conceptual overview of the caches supported by Crowd, including the LDAP database-backed caching discussed on this page. For a description of the other types of caching offered by Crowd, please refer to the overview of caching.
Supported LDAP Directories

Crowd's database-backed caching is available for all the LDAP directories that Crowd supports. See Configuring an LDAP Directory Connector for the list of supported directories.

Configuring the Cache

Screen snippets: Cache Configuration
These are the configuration options, as shown in the screenshots above:

- **Enable or disable the cache** for each directory on the directory connector's 'Details' tab. See Configuring an LDAP Directory Connector.
- Set the **polling interval** on the directory connector's 'Connector' tab. The polling interval, or sync interval, is the period of time (number of minutes) that Crowd will wait between its requests for updates from LDAP.
  - The length of your polling interval depends on the length of time you can tolerate stale data, the amount of load you want to put on Crowd and the LDAP server, and the size of your user base. 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 LDAP server with requests.
  - If in doubt, we recommend that you start with an interval of 60 minutes (this is the default setting) and reduce the value incrementally. You will need to experiment with your setup.

**Finding the Time Taken to Synchronize**

*Screen snippets: Information about the last synchronization*
The directory connector's 'Details' tab shows information about the last sync operation, including the length of time it took.

Manually Synchronizing the Cache

You can manually synchronize the cache by clicking the 'Synchronize Now' button on the directory connector's 'Details' tab. If a sync operation is already in progress, you cannot start another until the first has finished.

Notes

1. **Be aware of the optimal number of users.** We have optimized the database caching for directories containing approximately 10 000 (ten thousand) users. If your directory is significantly larger, the new
1. **You can reduce the number of LDAP users visible to Crowd.** You can narrow the LDAP user/group filter to control the size of the userbase visible to Crowd.

2. **Delegated Authentication directories are not cached.** Delegated Authentication directories are not cached, because only the authentication is delegated to the directory, and authentication itself is not cached.

3. **Synchronization errors are shown in the logs.** If there are any errors during the synchronization process, they will appear in the logs (not the UI). If one user fails to sync for some reason, the process will write the error to the logs, skip that user, and continue with the remaining users.

### Additional Notes for Active Directory

When Crowd synchronizes with Active Directory, Crowd requests only the changes from the LDAP server rather than the entire user base. This optimizes the synchronization process and gives much faster performance on the second and subsequent requests.

On the other hand, this synchronization method results in a few limitations:

1. **Externally moving objects out of scope or renaming objects causes problems in AD.** If you move objects out of scope, this will result in an inconsistent cache. We recommend that you 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. If you do need to make structural changes to your LDAP directory, manually synchronize the directory cache after you have made the changes to ensure cache consistency.

2. **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 a single directory to different AD servers for syncing.

3. **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.

4. **Obtaining AD 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](https://support.microsoft.com/en-us/help/865013).

### Our Test Results

We performed internal testing of synchronization with an AD server on our local network consisting of 10 000 users, 1000 groups and 200 000 memberships.

We found that the initial synchronization took about 5 minutes. Subsequent synchronizations with 100 modifications on the AD server took a couple of seconds to complete.

Please keep in mind that a number of factors come into play when trying to tune the performance of the synchronization process, including:

- **Size of userbase.** Use LDAP filters to keep this to the minimum that suits your requirements.
- **Type of LDAP server.** We currently support change detection in AD, so subsequent synchronizations are much faster for AD than for other LDAP servers.
- **Network topology.** The further away your LDAP server is from your application server, the more latent LDAP queries will be.
- **Database performance.** As the synchronization process caches data in the database, the performance of your database will affect the performance of the synchronization.
- **JVM heap size.** If your heap size is too small for your userbase, you may experience heavy garbage collection during the synchronization process which could in turn slow down the synchronization.

### RELATED TOPICS

- Overview of Caching
- Authorization Caching
- Configuring Caching for an Application
- Using Naive DN Matching
- Configuring an LDAP Directory Connector
- Managing Directories
Crowd Documentation

Using Naive DN Matching

When configuring an LDAP directory connector in Crowd, you can turn 'naive DN matching' on or off. A 'DN' is a distinguished name. Naive DN matching is also known as 'relaxed DN standardization'. 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 (that is, 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 naive DN matching provides significant performance benefits. For that reason, we recommend enabling it where possible.

Effect of Turning Naive DN Matching On or Off

<table>
<thead>
<tr>
<th>Naive DN Matching in Crowd</th>
<th>Processing in Crowd</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>Crowd will perform the full DN parsing and compare the parsed version.</td>
<td>See below for default settings for each directory type.</td>
</tr>
<tr>
<td>On</td>
<td>Crowd will perform a toLower operation and then do a direct comparison of the two DN strings.</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. The directory server always returns memberDNs in a compact format i.e. the separators are commas without spaces. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Compact format: 'cn=bob,dc=example,dc=com'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Not compact: 'cn=bob, dc=example, dc=com'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. The attribute names in the RDN are always lower case, or all searches for DNs and memberDN attributes are case insensitive.</td>
</tr>
</tbody>
</table>

Default Settings in Crowd

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

<table>
<thead>
<tr>
<th>Directory Type</th>
<th>Naive DN Matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApacheDS 1.0.x</td>
<td>Off</td>
</tr>
<tr>
<td>ApacheDS 1.5.x</td>
<td>Off</td>
</tr>
<tr>
<td>Apple Open Directory</td>
<td>On</td>
</tr>
<tr>
<td>FedoraDS</td>
<td>On</td>
</tr>
<tr>
<td>Generic LDAP</td>
<td>Off</td>
</tr>
<tr>
<td>Microsoft Active Directory</td>
<td>On</td>
</tr>
<tr>
<td>Novell eDirectory</td>
<td>Off</td>
</tr>
<tr>
<td>OpenDS</td>
<td>Off</td>
</tr>
</tbody>
</table>
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 and groups 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, because any user who could log into the Crowd application could change the application-level permissions for the Crowd application anyway.

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

### Directory-Level Permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Group</td>
<td>Allows applications to add groups to the directory.</td>
</tr>
<tr>
<td>Add User</td>
<td>Allows applications to add users to the directory.</td>
</tr>
<tr>
<td>Modify Group</td>
<td>Allows applications to modify groups in the directory.</td>
</tr>
<tr>
<td>Modify User</td>
<td>Allows applications to modify users in the directory.</td>
</tr>
</tbody>
</table>
1. Modify Group Attributes
   Allows applications to modify group attributes in the directory.

2. Modify User Attributes
   Allows applications to modify user attributes in the directory including the active option.

3. Remove Group
   Allows applications to delete groups from the directory.

4. 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.

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](image)

---

**View Directory - CROWD260**

<table>
<thead>
<tr>
<th>Permissions</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Group:</td>
<td></td>
</tr>
<tr>
<td>Add User:</td>
<td></td>
</tr>
<tr>
<td>Modify Group:</td>
<td></td>
</tr>
<tr>
<td>Modify User:</td>
<td></td>
</tr>
<tr>
<td>Modify Group Attributes:</td>
<td></td>
</tr>
<tr>
<td>Modify User Attributes:</td>
<td></td>
</tr>
<tr>
<td>Remove Group:</td>
<td></td>
</tr>
<tr>
<td>Remove User:</td>
<td></td>
</tr>
</tbody>
</table>

Update » Cancel
**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
- Using Naive DN Matching
- 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 organization, and give you a consolidated, centralized 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 organization 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
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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 apachetomcat/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.

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’**
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
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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-SECURITY'. 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`.
     - For PostgreSQL type `org.postgresql.Driver`.
     - For Oracle type `oracle.jdbc.OracleDriver`.
     - For MS SQL Server type `net.sourceforge.jtds.jdbc.Driver`.
     - For HSQLDB type `org.hsqldb.jdbcDriver`.
   - `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.

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
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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'**

<table>
<thead>
<tr>
<th>Import Type</th>
<th>Options</th>
<th>Results</th>
</tr>
</thead>
</table>

To Import Jive Forum users you will need to have the JDBC connection information and the necessary database drivers installed in the Crowd CLASS_PATH.

- **Directory:** Atlassian Internal
- **Product Database URL:** jdbc:mysql://localhost:jive_software?autoReconnect
- **Database Driver:** com.mysql.jdbc.Driver
- **Username:**
- **Password:**

**Next Step**

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

**Related Topics**
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.

How the CSV Importer Handles Data

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.

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:

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
</table>

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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 labeled '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
  - LDAP Object Structures
  - Configuring an SSL Certificate for Microsoft Active Directory
  - Configuring a Remote Crowd Directory
  - Configuring a Custom Directory Connector
  - Configuring a Delegated Authentication Directory
- Configuring Caching for an LDAP Directory
- Using Naive DN Matching
- 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:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSV Header Row</td>
<td>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.</td>
</tr>
<tr>
<td>Sample Row</td>
<td>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.</td>
</tr>
<tr>
<td>Mapping</td>
<td>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.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Crowd field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Name</td>
<td>Required. One of the rows on the screen must map this value to the CSV field containing the users' first names.</td>
</tr>
<tr>
<td>Last Name</td>
<td>Required. One of the rows on the screen must map this value to the CSV field containing the users' last names.</td>
</tr>
<tr>
<td>Email Address</td>
<td>Required. One of the rows on the screen must map this value to the CSV field containing the users' email addresses.</td>
</tr>
<tr>
<td>Username</td>
<td>Required. One of the rows on the screen must map this value to the CSV field containing the usernames.</td>
</tr>
<tr>
<td>Password</td>
<td>If your CSV file contains passwords, map this value to the CSV field containing the passwords.</td>
</tr>
<tr>
<td>None</td>
<td>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.</td>
</tr>
</tbody>
</table>

In the ‘Group Mappings’ section (if present), the ‘Mapping’ drop-down lists contain the following Crowd field names:
<table>
<thead>
<tr>
<th>Crowd field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Name</td>
<td>Required. One of the rows on the screen must map this value to the CSV field containing the names of the groups.</td>
</tr>
<tr>
<td>Username</td>
<td>Required. One of the rows on the screen must map this value to the CSV field containing the usernames.</td>
</tr>
<tr>
<td>None</td>
<td>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.</td>
</tr>
</tbody>
</table>

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'
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'
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
Once you have added a directory, you can import users and groups 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 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 and groups from LDAP into a delegated authentication directory. See CWD-1334.
- The ‘source directory’ is the directory you want to copy users and groups 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 and groups 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 and groups you want to copy.
   • 'Destination Directory' — Select the directory that you want to copy the users and groups 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 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 and groups have been imported into your Crowd directory.
    If the import of any users or groups failed, please check the log files to find out why.

Screenshot: 'Import users from one directory to another'

**Directory Importer**

<table>
<thead>
<tr>
<th>1. Import Type</th>
<th>2. Options</th>
<th>3. Confirmation</th>
<th>4. Results</th>
</tr>
</thead>
</table>

**Which directory do you want to copy your users from? And where do you want them to go?**

**Source Directory:** *Apache DS 1.5.1*  
The directory to import your users and groups from.

**Destination Directory:** *Employees*  
The directory to import your users and groups into.

**Overwrite Destination Directory:**  
Tick this box to delete and replace all details and memberships for users that already exist in the destination directory.

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
    - LDAP Object Structures
    - Configuring an SSL Certificate for Microsoft Active Directory
• Configuring a Remote Crowd Directory
• Configuring a Custom Directory Connector
• Configuring a Delegated Authentication Directory
• Configuring Caching for an LDAP Directory
• Using Naive DN Matching
• 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 Confluence 3.4 or earlier
    • Updating Files in a Confluence Evaluation Distribution
  • Integrating Crowd with Atlassian CrowdID
  • Integrating Crowd with Atlassian Crucible
  • Integrating Crowd with Atlassian FishEye
    • Configuring FishEye earlier than 4.0 with Crowd
  • Integrating Crowd with Atlassian JIRA
    • Integrating Crowd with Atlassian JIRA 4.2 or earlier
  • Integrating Crowd with Atlassian Bitbucket Server
  • Integrating Crowd with Acegi Security
    • Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  • Integrating Crowd with Apache
    • Disabling Previous Versions of the Crowd Apache Connector
    • Installing the Crowd Apache Connector on CentOS Linux
    • Installing the Crowd Apache Connector on Red Hat Enterprise Linux
    • Installing the Crowd Apache Connector on Ubuntu Linux
    • Installing the Crowd Apache Connector on Debian
    • Installing the Crowd Apache Connector on Other UNIX-Like Systems
    • Installing the Crowd Apache Connector on Windows
  • 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
  • Integrating Crowd with Atlassian HipChat
  • Integrating Crowd with Atlassian FishEye - DRAFT
• 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
  • Viewing Users in Directories Mapped to 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

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>appfuse-app</td>
<td>AppFuse-Based Application</td>
<td>View</td>
</tr>
<tr>
<td>crowd</td>
<td>Crowd Console</td>
<td>View</td>
</tr>
<tr>
<td>crowd-openid-server</td>
<td>CrowdID OpenID Provider</td>
<td>View</td>
</tr>
<tr>
<td>demo</td>
<td>Crowd Demo Application</td>
<td>View</td>
</tr>
<tr>
<td>google-apps</td>
<td>Google Applications Connector</td>
<td>View</td>
</tr>
<tr>
<td>lite-app</td>
<td>JIRA Server</td>
<td>View</td>
</tr>
</tbody>
</table>

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.
- '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.
- '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.
<table>
<thead>
<tr>
<th><img src="image" alt="Icon" /></th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>This marks a Bamboo server connected to Crowd.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>This marks a Confluence server connected to Crowd.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>This marks a Crucible server connected to Crowd.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>This marks a Fisheye server connected to Crowd.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>This marks a JIRA server connected to Crowd.</td>
</tr>
</tbody>
</table>

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
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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. It 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 Atlassian Bitbucket Server
- 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
- Integrating Crowd with Atlassian HipChat
1. Log in to the Crowd Administration Console.
2. Consider whether you need to add any directories, users and groups. If so, see the detailed instructions for your application.
3. Click Applications in the top navigation bar.
4. Click Add Application in the left-hand menu to display the first screen for the 'Add Application' wizard for Crowd.
5. Complete the form:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Type</td>
<td>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.</td>
</tr>
<tr>
<td>Name</td>
<td>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.</td>
</tr>
<tr>
<td>Description</td>
<td>A short description of the application. Note: A URL is often helpful.</td>
</tr>
<tr>
<td>Password</td>
<td>The password which the application will use when it authenticates against the Crowd framework as a client.</td>
</tr>
<tr>
<td>Confirm Password</td>
<td>Retype the same password as above, to confirm it.</td>
</tr>
</tbody>
</table>

6. Click Next.
7. Enter the connection details for your application, as described in the table below.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>The URL of your application. For example this may be <a href="http://jira.atlassian.com">http://jira.atlassian.com</a>. Remember to include the port, if you are not using a proxy. After entering the URL for the application, you can click Resolve IP Address. Crowd will attempt to resolve the IP address for your application.</td>
</tr>
</tbody>
</table>
Remote IP Address: This is the IP address of the server where your application exists. To help you work this out, you can click Resolve IP Address once you have entered a URL.

8. Click Next.
9. Now select one or more directories that this application can use for authentication and authorization:

10. Click Next.
11. In the 'Authorization' step you specify the users who are authorized to access the application.

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

- 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 remove next to the authorized groups' names.

12. Click Next.
13. 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 Add Application.

14. 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 Atlassian Bitbucket Server
- 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
- Integrating Crowd with Atlassian HipChat
- Integrating Crowd with Atlassian FishEye - DRAFT

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
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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 centralized 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
- Step 1. Configuring Crowd to Talk to Bamboo
  - 1.1 Prepare Crowd's Directories/Groups/Users for Bamboo
  - 1.2 Define the Bamboo Application in Crowd
  - 1.3 Specify which Users can Log In to Bamboo
  - 1.4 Specify the Address from which Bamboo can Log In to Crowd
- Step 2. Configuring Bamboo to Talk to Crowd
  - 2.1 Install the Crowd Client Libraries into Bamboo
  - 2.2 Edit Bamboo's crowd.properties file
  - 2.3 Configure Bamboo to use Crowd's Authenticator
  - 2.4 Configure External User Management in Bamboo
  - 2.5 (Optional) Enable Single Sign-On
  - 2.6 (Optional) Tune the Cache
- See Crowd in Action

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.

Do not deploy multiple Atlassian applications in a single Tomcat container.

Deploying multiple Atlassian applications in a single Tomcat container is not supported. We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration. See this FAQ for more information.

There are also a number of practical reasons why we do not support deploying multiple Atlassian applications in a single Tomcat container. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

Finally, we recommend not deploying any other applications to the same Tomcat container that runs Crowd, especially if these other applications have large memory requirements or require additional libraries in Tomcat's lib subdirectory.

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 Crowd distribution (not EAR-WAR) (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 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 crowd.properties file. You can find the crowd.properties file in either Bamboo/webapp/WEB-INF/classes/ (Bamboo 3.1 and earlier) or $BAMBOO_HOME/xml-data/configuration (Bamboo 3.2 or later). 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 authorizations 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:
1. 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

Before you begin Step 2

- If you are using **Bamboo 3.2 or later**, please refer to the Bamboo instructions on Integrating Bamboo with Crowd and skip Step 2 (and all sub-steps) on this page.
- 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.
   - **Please**, copy **CROWD/client/crowd-integration-client-X.X.X.jar** into **BAMBOO/webapp/WEB-INF/lib** directory.
   - If you are using the Crowd WAR distribution, then you will need to get the CROWD client libraries from the Crowd distribution, available on our **download site**.
   - Copy the Crowd client libraries and configuration files to Bamboo:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
</table>

---

*Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
CROWD/client/crowd-integration-client-X.X.X.jar | BAMBOO/webapp/WEB-INF/lib (No need to copy any crowd.jar files over for the latest Bamboo versions (after 4.0) as they already contain the needed jar files to work with Crowd)

CROWD/client/conf/crowd.properties | BAMBOO/webapp/WEB-INF/classes (Bamboo 3.1 and earlier) or $BAMBOO_HOME/xml-data/configuration (Bamboo 3.2 or later)

CROWD/client/conf/crowd-ehcache.xml | BAMBOO/webapp/WEB-INF/classes (Bamboo 3.1 and earlier) or $BAMBOO_HOME/xml-data/configuration (Bamboo 3.2 or later)

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 crowd.properties found in BAMBOO/webapp/WEB-INF/classes (Bamboo 3.1 and earlier) or $BAMBOO_HOME/xml-data/configuration (Bamboo 3.2 or later). Change the following properties:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>bamboo</td>
</tr>
<tr>
<td></td>
<td>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).</td>
</tr>
<tr>
<td>application.password</td>
<td>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).</td>
</tr>
<tr>
<td>crowd.server.url</td>
<td><a href="http://localhost:8095/crowd/services/">http://localhost:8095/crowd/services/</a></td>
</tr>
<tr>
<td></td>
<td>If your Crowd server's port is configured differently from the default (8095), set it accordingly.</td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>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.</td>
</tr>
</tbody>
</table>

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 atlassian-user.xml file (found in BAMBOO/webapp/WEB-INF/classes (Bamboo 3.1 and earlier) or $BAMBOO_HOME/xml-data/configuration (Bamboo 3.2 or later)) so that the contents of the file is:
2. At this stage, Bamboo is set up for centralized 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.

Security and Permission

You can change the following security and permission related settings for Bamboo.

<table>
<thead>
<tr>
<th>Change Global Security and Permission Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Enable External User Management?</td>
</tr>
<tr>
<td>Enable this option if you are delegating your user management to another user management system (e.g. Crowd).</td>
</tr>
<tr>
<td>□ Enable Signup?</td>
</tr>
<tr>
<td>This will allow users to sign up for an account to Bamboo.</td>
</tr>
<tr>
<td>□ Enable contact details to be displayed?</td>
</tr>
<tr>
<td>This will allow Bamboo users contact details to be visible. Disabling this option will hide the email address, IM address, and the group the user is in.</td>
</tr>
</tbody>
</table>

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 centralized 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:
<!--<authenticator
class="com.atlassian.bamboo.user.authentication.BambooAuthenticator"-->

Please, uncomment the authenticator "com.atlassian.crowd.integration.seraph.v25.BambooAuthenticator":

<!--
If you're authenticating against a Crowd server you can use this authenticator for single sign-on.
Enable it after configuring your Crowd properties through user management and restart Bamboo.
It does not support Crowd property changes at runtime. If you need to switch back to local users,
revert the change and restart Bamboo again.
-->
<authenticator
class="com.atlassian.crowd.integration.seraph.v25.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 centralized 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 Confluence 3.4 or earlier
Integrating Crowd with Atlassian Confluence

Atlassian's popular Confluence wiki can quickly be configured to use Crowd for user and group management.

On this page:

- Compatibility of Confluence and Crowd Versions
- Prerequisites
- Step 1. Configuring Crowd to Talk to Confluence
  - 1.1 Prepare Crowd's Directories/Groups/Users for Confluence
  - 1.2 Define the Confluence Application in Crowd
  - 1.3 Specify which Users can Log In to Confluence
  - 1.4 Specify the Address from which Confluence can Log In to Crowd
- Step 2. Configuring Confluence to talk to Crowd
  - 2.1 Add a Crowd Directory in Confluence
  - 2.2 Enable SSO integration with Crowd (Optional)
- See Crowd in Action
Compatibility of Confluence and Crowd Versions

Please ensure that your Crowd and Confluence versions are compatible:

<table>
<thead>
<tr>
<th>Confluence Version</th>
<th>Supported Crowd Version</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower than 2.6.2</td>
<td>N/A</td>
<td>Confluence does not support Crowd - please upgrade Confluence.</td>
</tr>
<tr>
<td>Between 2.6.2 and 2.7.4</td>
<td>1.2 and later</td>
<td>Confluence 2.6.1 is not supported - the earliest supported version is 2.6.2</td>
</tr>
<tr>
<td>Between 2.8 and 3.4.8</td>
<td>Between 1.3.2 and 2.2.7</td>
<td>In Confluence 2.8, the interface for atlassian-user changed. Crowd 1.3.2 is the earliest version to support this change.</td>
</tr>
<tr>
<td>Confluence 3.5 and above</td>
<td>Crowd 2.1 or Later</td>
<td>In Confluence 3.5 and above, the communication between Confluence and Crowd has been changed from SOAP to REST.</td>
</tr>
</tbody>
</table>

Prerequisites

Do not deploy multiple Atlassian applications in a single Tomcat container.

Deploying multiple Atlassian applications in a single Tomcat container is not supported. We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration. See this FAQ for more information.

There are also a number of practical reasons why we do not support deploying multiple Atlassian applications in a single Tomcat container. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

Finally, we recommend not deploying any other applications to the same Tomcat container that runs Crowd, especially if these other applications have large memory requirements or require additional libraries in Tomcat's lib subdirectory.

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 Crowd distribution (not EAR-WAR) (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 authorizations 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

The instructions for step 2 below apply to Confluence 3.5 or newer. If you use Confluence 3.4 or older, please follow "Step 2" on Integrating Crowd with Atlassian Confluence 3.4 or earlier instead.

2.1 Add a Crowd Directory in Confluence

Confluence can use Crowd for user authentication simply by adding the 'Atlassian Crowd' user directory.

1. Log in to Confluence Admin as 'confluence-administrator'.
2. Click on the 'User Directories' label of the left bar under the 'Security' tab.
3. Click 'Add Directory'. Then select 'Atlassian Crowd' from the dropdown list. Click 'Next'.
4. Enter connection parameters and save. Now a new Crowd directory should appear on the user directory list.
For more information on configuring a Crowd remote directory in Confluence, check out the Confluence documentation on Connecting to Crowd or JIRA for User Management.

2.2 Enable SSO integration with Crowd (Optional)

1. If Confluence is running, shut it down first.
2. Now, edit the file CONFLUENCE/confluence/WEB-INF/classes/seraph-config.xml Comment out the line:-

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

3. Uncomment the line:-

```xml
<authenticator
class="com.atlassian.confluence.user.ConfluenceCrowdSSOAuthenticator"/>
```

3. Copy the crowd.properties file from CROWD/client/conf/ to CONFLUENCE/confluence/WEB-INF/classes.
4. Edit CONFLUENCE/confluence/WEB-INF/classes/crowd.properties. Change the following properties:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>confluence</td>
</tr>
</tbody>
</table>

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 Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.password</td>
<td>The <strong>application.name</strong> and <strong>application.password</strong> must match the Name and Password that you specified when defining the application in Crowd (see Step 1 above).</td>
</tr>
<tr>
<td>crowd.base.url</td>
<td><a href="http://localhost:8095/crowd/">http://localhost:8095/crowd/</a> If your Crowd server's port is configured differently from the default (i.e. 8095), set it accordingly.</td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>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 to the required number of minutes between validation requests. The recommended default is 2 minutes. Setting this value to 1 or higher will increase the performance of Crowd's integration. Setting this value to 0 will cause the application to perform authentication checks on each request but can cause poor performance, especially with Crowd 2.1 - Crowd 2.3.2 using REST due to CWD-2646.</td>
</tr>
</tbody>
</table>

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

---

**It is possible to define multiple user directories in Confluence.** However, if you enable Crowd SSO integration, you will only be able to authenticate as users from the Crowd server defined in the *crowd.properties* file. To log in using a user from another directory, such as the Confluence Internal Directory, you will need to disable Crowd SSO by reverting back to the default Confluence authenticator. Steps to disable Crowd SSO:

1. Shut down Confluence if it is currently running
2. Edit the file **CONFLUENCE/confluence/WEB-INF/classes/seraph-config.xml**
3. Uncomment the default Confluence authenticator:

   ```xml
   <authenticator
class="com.atlassian.confluence.user.ConfluenceAuthenticator" />
   ```

   Comment out the Crowd SSO Authenticator:

   ```xml
   <!-- <authenticator
class="com.atlassian.confluence.user.ConfluenceCrowdSSOAuthenticator"/> -->
   ```

   Restart Confluence

---

**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 **centralized 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!
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 (A
D) 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 synchronized between AD and Confluence. For example, if a user/group is added/deleted from AD it will be automatically added/deleted from Confluence.

Components

<table>
<thead>
<tr>
<th>Confluence NTLM plugin</th>
<th>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. <strong>Note:</strong> This plugin is not officially supported by Atlassian.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd</td>
<td>Crowd takes care of the synchronization of users/groups between Active Directory and Confluence. <strong>Note:</strong> 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.</td>
</tr>
<tr>
<td>Active Directory (AD) on Windows 2003 Server</td>
<td>Active Directory (AD) on Windows 2003 Server — you must already have an AD instance set up and running with a domain controller.</td>
</tr>
<tr>
<td>Confluence</td>
<td>The machine running Confluence must be part of the Windows domain or installed on the same box as the domain controller.</td>
</tr>
</tbody>
</table>

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) 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 Confluence 3.4 or earlier**

This is an alternate step to “Step 2” defined in Integrating Crowd with Atlassian Confluence for users wanting to integrate Crowd with Confluence 3.4 or earlier.

- If you are using Confluence 3.5 or later, please follow the guide on Integrating Crowd with Atlassian Confluence.
- If you are using Confluence 3.4 or earlier, please complete "Step 1" from Integrating Crowd with Atlassian Confluence before attempting the alternate “Step 2” below.
- Crowd 2.3.3 or later requires Confluence 3.2.1 at minimum. See also CWD-2680.

**Step 2. Configuring Confluence to talk to Crowd**

- **1. Install the Crowd Client Library into Confluence**
- **2. Configure Confluence to use Crowd's Authenticator**
- **3. Enable Confluence's External User Management**
- **4. (Optional) Tune the Cache**

**Step 2. Configuring Confluence to talk to Crowd**

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 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 Crowd distribution, available on our download site.
2. If you are using the Windows Evaluation distribution of Confluence, please see this page on how to update the `crowd.properties` file in Confluence.
3. Copy the Crowd client library and configuration file to Confluence:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/crowd-integration-client-X.X.X.jar</td>
<td>CONFLUENCE/confluence/WEB-INF/lib</td>
</tr>
<tr>
<td>CROWD/client/conf/crowd.properties</td>
<td>CONFLUENCE/confluence/WEB-INF/classes</td>
</tr>
</tbody>
</table>

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.

Be sure that there is only one `crowd-integration-client-x.x.x.jar` file in the `lib` directory. Otherwise, it would cause library incompatibilities.
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.

4. Replace Confluence's cache configuration file:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Replace File</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/conf/crowd-ehcache.xml</td>
<td>CONFLUENCE/confluence WEB-INF/classes/crowd-ehcache.xml</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>confluence</td>
</tr>
<tr>
<td></td>
<td>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).</td>
</tr>
<tr>
<td>application.password</td>
<td>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).</td>
</tr>
<tr>
<td>crowd.server.url</td>
<td><a href="http://localhost:8095/crowd/services/">http://localhost:8095/crowd/services/</a></td>
</tr>
<tr>
<td></td>
<td>If your Crowd server's port is configured differently from the default (i.e. 8095), set it accordingly.</td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>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 to the required number of minutes between validation requests. The recommended default is 2 minutes. Setting this value to 1 or higher will increase the performance of Crowd's integration.</td>
</tr>
</tbody>
</table>

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 content of the file is:

```
Setting this value to 0 will cause the application to perform authentication checks on each request but can cause poor performance, especially with Crowd 2.1 - Crowd 2.3.2 using REST due to CWD-2646.
```
2. At this stage, Confluence is set up for **centralized authentication**. If you wish to enable **single sign-on (SSO)** or if you are using **Confluence 3.2.1 or later**, take the following steps to ensure that Confluence's authentication and access request calls will be performed using Seraph:

```xml
<atlassian-user>
  <repositories>
    <crowd key="crowd" name="Crowd Repository"/>
  </repositories>
</atlassian-user>
```

Make sure the content of the file is only what is indicated above, otherwise you may get this error

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

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

Add a new authenticator, choosing the one relevant to your version of Confluence:

- If you are using Confluence 3.4 or later:

  ```xml
  <authenticator
class="com.atlassian.crowd.integration.seraph.v22.ConfluenceAuthenticator"/>
  ```

- If you are using Confluence 3.3.3 or earlier:

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

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**.

**Note:**

- 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...
### 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).

### Updating Files in a Confluence Evaluation Distribution

This page tells you how to update the `crowd.properties` file in Confluence, if you are using the Windows Evaluation distribution of Confluence.

1. **Download 7-zip**, a program that you can use to unzip a JAR file.
2. **Navigate to your** `C:\Program Files\Atlassian\Confluence Evaluation 3.3.1\lib` **directory** and open the `confluence-3.3.1-war.jar` file in 7zip.
3. **Navigate to the relevant** `../WEB-INF/classes` **directory.**
4. Edit the `crowd.properties` file and save the changes to the zip archive.

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.
   - For the Application type select 'Generic Application'
   - For Name and Password, the values you specify 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.)

Need more help? See the full instructions for the Add application wizard.

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 authorizations 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:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>crowd-openid-server</td>
</tr>
<tr>
<td></td>
<td>The application.name and application.password</td>
</tr>
<tr>
<td></td>
<td>must match the Name and Password that you</td>
</tr>
<tr>
<td></td>
<td>specified when you defined the application in</td>
</tr>
<tr>
<td></td>
<td>Crowd (see Step 1 above).</td>
</tr>
<tr>
<td>application.password</td>
<td>The application.name and application.password</td>
</tr>
<tr>
<td></td>
<td>must match the Name and Password that you</td>
</tr>
<tr>
<td></td>
<td>specified when you defined the application in</td>
</tr>
<tr>
<td></td>
<td>Crowd (see Step 1 above).</td>
</tr>
<tr>
<td>application.login.url</td>
<td><a href="http://localhost:8095/openidserver">http://localhost:8095/openidserver</a></td>
</tr>
<tr>
<td></td>
<td>The application.login.url should point to the</td>
</tr>
<tr>
<td></td>
<td>correct host and port of the CrowdID</td>
</tr>
<tr>
<td></td>
<td>application.</td>
</tr>
</tbody>
</table>
**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.

**After editing these properties**, you must restart your CrowdID container before the changes will take effect.

**See CrowdID in Action**

- Go to [http://localhost:8095/openidserver](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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
    - Installing the Crowd Apache Connector on CentOS Linux
    - Installing the Crowd Apache Connector on Red Hat Enterprise Linux
    - Installing the Crowd Apache Connector on Ubuntu Linux
    - Installing the Crowd Apache Connector on Debian
    - Installing the Crowd Apache Connector on Other UNIX-Like Systems
    - Installing the Crowd Apache Connector on Windows
  - 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
  - Integrating Crowd with Atlassian HipChat
  - Integrating Crowd with Atlassian FishEye - DRAFT
- 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
  - Viewing Users in Directories Mapped to an Application
  - Specifying which Groups can access an Application
  - Effective memberships with multiple directories
  - Specifying an Application’s Address or Hostname
  - Testing a User’s Login to an Application
  - Enforcing Lower-Case Usernames and Groups for an Application
You can use Crowd to provide external authentication and authorization 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 Authorization 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
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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 authorization for Atlassian's FishEye source-repository viewer.

Crowd supports centralized 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
- 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 authorizations 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 authorize 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

Click here for instructions for older versions of FishEye (before 4.0.0).

To set up FishEye to use Crowd authentication, follow the instructions in the FishEye documentation.

If you have groups in the Crowd directory that is mapped to your FishEye application (see Step 1 above), the Crowd groups can be seen in FishEye. You can use those groups to control access to your FishEye repositories. See Permissions in the FishEye documentation for details.

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
- Effective memberships with multiple directories
- Specifying an Application’s Address or Hostname
- Testing a User’s Login to an Application
- Enforcing Lower-Case Usernames and Groups 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 earlier than 4.0 with Crowd

You can use Crowd to provide external authentication and authorization for Atlassian’s FishEye source-repositor
Crowd supports centralized authentication and single sign-on (SSO) for FishEye versions 1.3.1 up to 3.10. If you are using FishEye 4.0 or later, see Integrating Crowd with Atlassian FishEye.

**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 authorization in FishEye (if required)
- **Step 3. Override Crowd default properties (optional)**
- 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.

**Crowd Client JAR**

Please make sure you use the default Crowd client JAR that ships with FishEye. In particular, FishEye is not compatible with the crowd-integration-client-2.0.7.jar that is bundled with Crowd 2.0.7. See the Crowd 2.0.7 Release Notes.

**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 authorizations 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:

[Table]

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 – 3.10.

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 authorization 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 area and click Authentication (under 'Security Settings').
2. Click Edit under 'JIRA/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 Remove 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 authorization in FishEye (if required)

If you have groups in the Crowd directory that is mapped to your FishEye application (see Step 1 above), the Crowd groups can be seen in FishEye. You can use those groups to control access to your FishEye repositories.

See **Permissions** in the FishEye documentation for details.

**Step 3. Override Crowd default properties (optional)**

You set the basic Crowd properties, such as the application name, password and URL, using the FishEye administration screens (described above). You can also fine tune your Crowd integration by overriding the default Crowd properties, such as the session validation interval and SSO cookie name, by manually editing the `config.xml` file in your FishEye installation directory.

**To override the default Crowd properties:**

1. Shutdown the application.
2. Backup and then open the `config.xml` file in your `<FishEye home directory>` (the folder where you installed FishEye).
3. Add a new `<crowd-properties>` element to the file.
4. Override the default values for any of the Crowd properties (described in the `crowd.properties` file) by adding the property in the `<crowd-properties>` section with the desired value.

For example, your `config.xml` file should look like this, if you want to set the session validation interval to 20 minutes:

```xml
<config control-bind="127.0.0.1:8059" version="1.0">
  <crowd-properties>
    session.validationinterval=20
  </crowd-properties>
</config>
```
Note that FishEye 2.8, and later, overrides the Crowd defaults with these values:

<table>
<thead>
<tr>
<th>Property</th>
<th>Crowd Default</th>
<th>FishEye 2.8+</th>
</tr>
</thead>
<tbody>
<tr>
<td>http.timeout</td>
<td>5000 (milliseconds)</td>
<td>5000 (milliseconds)</td>
</tr>
<tr>
<td>socket.timeout</td>
<td>600000 (milliseconds)</td>
<td>20000 (milliseconds)</td>
</tr>
</tbody>
</table>

5. Save the file and restart FishEye.

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
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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 JIRA

Currently Crowd supports centralized 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.

On this page:

- Compatibility of JIRA and Crowd Versions
- Prerequisites
- Step 1. Configuring Crowd to talk to JIRA
  - 1.1 Prepare Crowd's Directories/Groups/Users for JIRA
  - 1.2 Define the JIRA Application in Crowd
  - 1.3 Specify which users can log in to JIRA
  - 1.4 Specify the address from which JIRA can log in to Crowd
- Step 2. Configuring JIRA to talk to Crowd
  - 2.1 Add a Crowd Directory in JIRA
  - 2.2 Configure JIRA to use Crowd's Authenticator to enable SSO (Optional)
  - 2.3 (Optional) Disable the Auto-Complete Function in JIRA's User Picker
- See Crowd in Action
- Known Limitations

Compatibility of JIRA and Crowd Versions

Please ensure that your Crowd and JIRA versions are compatible:

- If you are using JIRA 4.2 please upgrade to Crowd 2.0.7 or later. (watch out for Crowd 2.4 though:
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. OR: 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

If multiple versions of JIRA are being connected to Crowd, ensure you define an application in Crowd for each one

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 authorizations 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:

### JIRA Directory in Crowd

Map your groups, per directory, to the application. If the directory does not allow all to authenticate, then a user must belong to a mapped group in order to access the application.

<table>
<thead>
<tr>
<th>Directory — Group</th>
<th>Status</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>JIRA Directory — jira-administrators</td>
<td>Active</td>
<td>Remove</td>
</tr>
<tr>
<td>JIRA Directory — jira-developers</td>
<td>Active</td>
<td>Remove</td>
</tr>
<tr>
<td>JIRA Directory — jira-users</td>
<td>Active</td>
<td>Remove</td>
</tr>
</tbody>
</table>

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

The instructions for step 2 below apply to JIRA 4.3 or newer. If you use JIRA 4.2 or older, please follow “Step 2” on Integrating Crowd with Atlassian JIRA 4.2 or earlier instead.

2.1 Add a Crowd Directory in JIRA

JIRA can use Crowd for user authentication simply by adding ‘Atlassian Crowd’ as user directory.

1. Login to the administration section of JIRA
2. Click on the ‘User Directories’ label of the left bar under the ‘User management’ tab.
3. Click ‘Add Directory’. Then select ‘Atlassian Crowd’ from the dropdown list. Click ‘Next’.
4. Enter connection parameters and save. If you configure Server URL to use HTTPS, by replacing http:// with https://, communications between JIRA and Crowd will be encrypted.
5. Now a new Crowd directory should appear on the user directory list.

For more information on configuring a Crowd directory in JIRA, check out the JIRA documentation on Connecting to Crowd or Another JIRA Server for User Management.

2.2 Configure JIRA to use Crowd’s Authenticator to enable SSO (Optional)

At this stage, JIRA is set up for centralized authentication. If you wish, you can now enable single sign-on (SSO) to JIRA. This will ensure that JIRA’s authentication and access request calls will be performed using Seraph.

Note: if you are migrating/upgrading a JIRA instance that already uses Crowd, you will need to merge these files (not overwrite them).
1. If JIRA is running, shut it down first.
2. Edit the JIRA/atlassian-jira/WEB-INF/classes/seraph-config.xml file. Comment out the authenticator node:

   <!--<authenticator
   class="com.atlassian.jira.security.login.JiraSeraphAuthenticator"/>
   -->

   Uncomment the line that contains the new authenticator:

   <authenticator
   class="com.atlassian.jira.security.login.SSOSeraphAuthenticator"/>

3. Copy the crowd.properties file from CROWD/client/conf/ to JIRA/atlassian-jira/WEB-INF/classes.
4. Edit JIRA/atlassian-jira/WEB-INF/classes/crowd.properties. Change the following properties:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>jira</td>
</tr>
<tr>
<td></td>
<td>The application name must match the name that you specified when you</td>
</tr>
<tr>
<td></td>
<td>defined the application in Crowd (see Step 1 above).</td>
</tr>
<tr>
<td>application.password</td>
<td>The password must match the one that you specified when you defined</td>
</tr>
<tr>
<td></td>
<td>the application in Crowd (see Step 1 above).</td>
</tr>
<tr>
<td>crowd.base.url</td>
<td>eg. (<a href="http://localhost:8095/crowd/">http://localhost:8095/crowd/</a>)</td>
</tr>
<tr>
<td></td>
<td>If your Crowd server's port is configured differently from the default (i.e. 8095), set it accordingly.</td>
</tr>
<tr>
<td></td>
<td>crowd.base.url must be the same URL used to access Crowd in your</td>
</tr>
<tr>
<td></td>
<td>Browser.</td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>Set to 0, if you want authentication checks to occur on each request.</td>
</tr>
<tr>
<td></td>
<td>Otherwise set to the number of minutes between request to validate if</td>
</tr>
<tr>
<td></td>
<td>the user is logged in or out of the Crowd SSO server. Setting this</td>
</tr>
<tr>
<td></td>
<td>value to 1 or higher will increase the performance of Crowd's</td>
</tr>
<tr>
<td></td>
<td>integration.</td>
</tr>
</tbody>
</table>

   It is possible to define multiple user directories in JIRA. However, if you enable SSO integration, you will only be able to authenticate as users from the Crowd server defined in the crowd.properties file.

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

2.3 (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.

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 centralized authentication in action!
If you have enabled SSO, you can try adding the JIRA Directory in Crowd and the 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

If you are using JIRA 4.2, 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. We recommend upgrading JIRA or deactivating the user's account by removing them from the jira-users group.

If you are using JIRA 4.3 or later, this problem has been resolved, allowing the removal of users that are externally managed, despite existing data associations. When a user managed by an external system such as Crowd is removed, any user associations in JIRA will continue to be associated, with the username acting as a placeholder. This username will not be listed in the User Browser and no profile exists for that user.

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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
    - Installing the Crowd Apache Connector on CentOS Linux
    - Installing the Crowd Apache Connector on Red Hat Enterprise Linux
    - Installing the Crowd Apache Connector on Ubuntu Linux
    - Installing the Crowd Apache Connector on Debian
    - Installing the Crowd Apache Connector on Other UNIX-Like Systems
    - Installing the Crowd Apache Connector on Windows
  - 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
  - Integrating Crowd with Atlassian HipChat
  - Integrating Crowd with Atlassian FishEye - DRAFT
- 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
  - Viewing Users in Directories Mapped to an Application
  - Specifying which Groups can access an Application
  - Effective memberships with multiple directories
  - Specifying an Application's Address or Hostname
  - Testing a User's Login to an Application
  - Enforcing Lower-Case Usernames and Groups for an Application
Integrating Crowd with Atlassian JIRA 4.2 or earlier

This is an alternate step to "Step 2" defined in Integrating Crowd with Atlassian JIRA for users wanting to integrate Crowd with JIRA 4.2 or earlier.

- If you are using JIRA 4.3 or later, please follow the guide on Integrating Crowd with Atlassian JIRA.
- If you are using JIRA 4.2 or earlier, please complete "Step 1" from Integrating Crowd with Atlassian JIRA before attempting the alternate "Step 2" below.

Use the client libraries from Crowd 2.2.7 to integrate with JIRA 4.2 or earlier even when the Crowd server is more recent. The client libraries from Crowd 2.2.7 remain compatible with later releases of the Crowd server.

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 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 Crowd distribution, available on our download site.
2. Copy the Crowd client libraries and configuration files to JIRA:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/crowd-integration-client-X.X.X.jar</td>
<td>JIRA/atlassian-jira/WEB-INF/lib</td>
</tr>
<tr>
<td>CROWD/client/conf/crowd.properties</td>
<td>JIRA/atlassian-jira/WEB-INF/classes</td>
</tr>
</tbody>
</table>

Duplicate Crowd Client libraries in your classpath

JIRA should only have a single copy of crowd-integration-client installed. Therefore you need to delete the existing crowd-integration-client-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:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Replace File</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6. Edit `JIRA/atlassian-jira/WEB-INF/classes/crowd.properties`. Change the following properties:

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

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.

**Note:** if you are migrating/upgrading a JIRA instance that already uses Crowd, you will need to merge these files (not overwrite 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:
1. -- 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:
3. At this stage, JIRA is set up for **centralized authentication**. If you wish, you can now enable **single sign-on (SSO)** to JIRA. This will ensure that JIRA's authentication and access request calls will be performed using Seraph. When authentication or access request calls are performed versus the OSUser framework, the JIRA stack will call the Crowd providers and propertyset implementations.

   **Edit** the JIRA/atlassian-jira/WEB-INF/classes/seraph-config.xml file. **Comment out** the authenticator node:

   ```xml
   <!--<authenticator
      class="com.atlassian.jira.security.login.JiraOsUserAuthenticator"/>
   -->
   
   Add a new authenticator, choosing the one relevant to your version of JIRA:
   - If you are using JIRA 4.2.x:
     ```xml
     <authenticator
        class="com.atlassian.crowd.integration.seraph.v22.JIRAAuthenticator"/>
     ```
   - If you are using JIRA 4.1.2 or earlier:
     ```xml
     <authenticator
        class="com.atlassian.crowd.integration.seraph.JIRAAuthenticator"/>
     ```

2.3 *(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.

ℹ️ 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.

2.4 *(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.

Integrating Crowd with Atlassian Bitbucket Server
You can use Crowd to provide external authentication, and to determine group memberships for authorization, for Atlassian's Bitbucket Server.

On this page:

- Prerequisites
- Step 1. Configuring Crowd to talk to Bitbucket Server
- Step 2. Configuring Bitbucket Server to talk to Crowd

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 Bitbucket Server. Refer to either Getting started (if running the Bitbucket Server installer) or Install Bitbucket Server from an archive file, for detailed information on how to do this. We will refer to the Bitbucket Server root folder as Bitbucket.

Crowd Client JAR

Please make sure you use the default Crowd client JAR that ships with Bitbucket Server.

Step 1. Configuring Crowd to talk to Bitbucket Server

1.1 Prepare Crowd's directories/groups/users for Bitbucket Server

The Bitbucket Server application will need to authenticate users against a directory configured in Crowd. You will need to set up a directory in Crowd for Bitbucket Server. For more information on how to do this, see Adding a Directory. We will assume that the directory is called Bitbucket 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 Bitbucket Directory to house Bitbucket Server users.

If you wish to use Crowd groups to control access to your Bitbucket Server projects, 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 Bitbucket 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 Bitbucket Server application in Crowd

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

1. Log in to the Crowd Administration Console and navigate to Applications > Add Application.
2. Complete the 'Add Application' wizard for the Bitbucket Server application. See the instructions. When prompted for an Application Type, choose Generic Application if the Bitbucket Server option is not available. Note that the Name and Password values you specify in the 'Add Application' wizard must match those for Application Name and Application Password that you will set in Bitbucket Server's 'Configure Atlassian Crowd Server' screen (see Step 2 below).

1.3 Specify which users can log in to Bitbucket Server

Once Crowd is aware of the Bitbucket Server application, Crowd needs to know which users can authenticate (log in) to Bitbucket Server via Crowd. As part of the 'Add Application' wizard, you will set up your directories and the group memberships that Bitbucket Server will use for authorization. If necessary, you can adjust these settings after completing the wizard.

You can either allow entire directories to authenticate, or just particular groups within the directories. If you only specific groups to be able to log in, please see Mapping a Directory to an Application and Specifying which Groups can access an Application.

1.4 Specify the address from which Bitbucket Server can log in to Crowd

As part of the 'Add Application' wizard, you will need to tell Crowd the IP address and/or hostname of the server
that Bitbucket Server is connecting from. See Specifying an Application's Address or Hostname.

Step 2. Configuring Bitbucket Server to talk to Crowd

2.1 Connecting Bitbucket Server to Crowd

To set up Bitbucket Server to use Crowd authentication, follow the instructions in Connecting Bitbucket Server to Crowd.

2.2 Configure group permissions in Bitbucket Server (if required)

If you have created groups in the Crowd directory which is mapped to your Bitbucket Server application (see Step 1 above), the Crowd groups can be seen in Bitbucket Server. Now you can set up group permissions for your Bitbucket Server projects. See Creating projects.

2.3 Configure Bitbucket Server to enable SSO with Crowd (optional)

Once the Crowd directory has been set up, you can enable Crowd SSO integration in Bitbucket Server. See Connecting to Crowd for details.

Integrating Crowd with Acegi Security

Crowd 2.6 removed support for Acegi Security (Upgrade Notes). Please upgrade to Spring Security or use an older release of Crowd.

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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.
2. Since we will be editing the core Acegi configuration, we will need the full source code of the application.

   cd myproject
   mvn appfuse:full-source

3. Build it.

   mvn clean install

4. Run it.

   mvn jetty:run-war -Dmaven.test.skip

5. Play with it.

   http://localhost:8080/

6. Shut it down.

   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:

```xml
<dependencies>
  <dependency>
    <groupId>com.atlassian.crowd</groupId>
    <artifactId>crowd-integration-client</artifactId>
    <version>1.5.1</version>
  </dependency>
  ...
</dependencies>
```

You will also need to create the file `myproject/src/main/resources/crowd.properties`: 
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>appfuse</td>
</tr>
<tr>
<td>application.password</td>
<td>password</td>
</tr>
<tr>
<td>application.login.url</td>
<td><a href="http://localhost:8095/crowd/">http://localhost:8095/crowd/</a></td>
</tr>
<tr>
<td>crowd.server.url</td>
<td><a href="http://localhost:8095/crowd/services/">http://localhost:8095/crowd/services/</a></td>
</tr>
<tr>
<td>session.isauthenticated</td>
<td>session.isauthenticated</td>
</tr>
<tr>
<td>session.tokenkey</td>
<td>session.tokenkey</td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>0</td>
</tr>
<tr>
<td>session.lastvalidation</td>
<td>session.lastvalidation</td>
</tr>
</tbody>
</table>

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 Centralized 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`:

```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 Acegi configuration in `myproject/src/main/webapp/WEB-INF/security.xml`. In order to get centralized 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`:

```xml
<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="crowd UserManager"/>
    <property name="authorityPrefix" value="ROLE_"/>
</bean>
```
2. Add the definition of the RemoteCrowdAuthenticationProvider which will delegate Acegi's authentication requests to Crowd:

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

3. Replace the DaoAuthenticationProvider with our authenticator in the authentication manager:

```xml
<bean id="authenticationManager"
     class="org.acegisecurity.providers.ProviderManager">
    <property name="providers">
        <list>
            <ref local="crowdAuthenticationProvider"/>
            <!--ref local="daoAuthenticationProvider"-->
            <ref local="anonymousAuthenticationProvider"/>
            <ref local="rememberMeAuthenticationProvider"/>
        </list>
    </property>
</bean>
```

4. Now do a:

```bash
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 authorization 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 centralized authentication 😊

**Application-level centralized 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 centralized user management, AppFuse will need to delegate its calls to create, retrieve, update and delete users to Crowd using Crowd's remote API. This will prevent data redundancy and eliminate the hassle of data synchronization. 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:

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

2. Add the definition of the CrowdLogoutHandler:

```xml
<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. You may need to uncomment the logout filter.

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

4. If the logout filter is not defined in the filter invocation list, you will need to add it:
5. Now repeat:

```bash
mvn jetty:run-war -Dmaven.test.skip=true
```

SSO will only work for users that are able to authenticate with both applications and are authorized 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 Basic Authentication requests made to an Apache web server.

The following features are supported:
- Authentication: Use Crowd to password-protect resources on your website.
- Authorization: Configure website locations to restrict access to specific Crowd groups or users.

This version of the connector is supported under Linux. Please see Choosing the Correct Apache Connector for your Operating System for details and alternatives.

**Note:** These instructions apply to Crowd 2.1 and later. We assume some UNIX system and Apache configuration knowledge.

On this page:
- Prerequisites
- Step 1. Disabling any Previous Version of the Crowd Apache Connector
- Step 2. Configuring Crowd to Talk to Apache
- Step 3. Choosing the Correct Apache Connector for your Operating System
- Step 4. Configuring Authentication
- Step 5. Configuring Authorization
- Step 6. Configuring Subversion (Optional)
- Notes

**Prerequisites**

Download and configure Crowd. Refer to the [Crowd installation guide](#) for detailed information on how to do this.

**Step 1. Disabling any Previous Version of the Crowd Apache Connector**

If you are upgrading from a previous version of the Connector, you must disable it by following these instructions.
before proceeding.

Step 2. Configuring Crowd to Talk to Apache

If you are upgrading from an earlier version of the Apache Connector, you will have already completed this step and you can skip it.

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

1. Define Apache as a Crowd-connected application to Crowd.
2. Add and configure the directories visible to Apache.
3. Add and map the groups which are allowed to authenticate with Apache.

Step 3. Choosing the Correct Apache Connector for your Operating System

The installation procedures for Apache and the Crowd Apache connector vary depending on the operating system you are using. Use the links below to find installation instructions for your chosen operating system. If you have not chosen an operating system yet, you will probably find one of the Linux variants easiest to set up.

The 2.x version of the Crowd-Apache connector supports advanced features such as nested groups and single sign-on but is currently only supported for certain operating systems.

### Note about Crowd-Apache 1.4 Connector

This document is for the 2.x version of Crowd-Apache Connector, if you can't find a compatible version of the 2.x Subversion Connector for your O/S in the table below then refer to the 1.4 version of the Crowd-Apache Connector.

Previous version of Crowd-Apache connectors don’t support the SVNParentPath directive. Crowd 2.9.1 is not compatible with version 1.4 of the Crowd-Apache connector.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Crowd-Apache Connector 2.0</th>
<th>Crowd-Apache Connector 1.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat Enterprise Linux</td>
<td><img src="#" alt="✓ 6.0" /> <img src="#" alt="5.5" /> <a href="#">Installation guide</a></td>
<td><img src="#" alt="i" /> <a href="#">Installation guide for other OS versions</a></td>
</tr>
<tr>
<td>CentOS Linux</td>
<td><img src="#" alt="✓ 5.5" /> <a href="#">Installation guide</a></td>
<td><img src="#" alt="i" /> <a href="#">Installation guide for other OS versions</a></td>
</tr>
<tr>
<td>Ubuntu Linux</td>
<td><img src="#" alt="✓ 11.04" /> <img src="#" alt="10.10" /> <img src="#" alt="10.04" /> <a href="#">Installation guide</a></td>
<td><img src="#" alt="i" /> <a href="#">Installation guide for other OS versions</a></td>
</tr>
<tr>
<td>Debian</td>
<td><img src="#" alt="✓ 6.0" /> <a href="#">Installation guide</a></td>
<td><img src="#" alt="i" /> <a href="#">Installation guide for other OS versions</a></td>
</tr>
<tr>
<td>Other UNIX-Like Systems</td>
<td>Not supported*</td>
<td><img src="#" alt="✓" /> <a href="#">Installation guide</a></td>
</tr>
<tr>
<td>Windows</td>
<td>Not supported*</td>
<td><img src="#" alt="✓" /> <a href="#">Installation guide</a></td>
</tr>
</tbody>
</table>

*The Crowd-Apache Connector 2.0 source code is available with instructions to build from source. If you have the expertise to compile the C library and get it working for an environment that is currently not supported then feel free to contribute by adding a comment to this page. We will try to incorporate your fix in the next release of the Crowd-Apache Connector.*

Step 4. Configuring Authentication

In this section, you will tell Apache to use Crowd to authenticate requests for a particular location. Edit the Apache config file and add the following commands to a `<Location>` or `<Directory>` section.
This is the minimum configuration required to password-protect a location with Crowd.

<table>
<thead>
<tr>
<th>Command</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>`&lt;Directory &quot;/var/mysite/&quot;&gt;</td>
<td>See the Apache documentation for the format of the <code>&lt;Directory&gt;</code> and <code>&lt;Location&gt;</code> directives. We have used the directory path of <code>/var/mysite/</code> as the simplest example. You may substitute your own directory path here.</td>
</tr>
<tr>
<td>AuthName &quot;Atlassian Crowd&quot;</td>
<td>Defines the realm of the authentication. This information is typically provided to the user in the dialog box popped up by their browser. This must be a unique name for each Crowd application</td>
</tr>
<tr>
<td>AuthType Basic</td>
<td>Tells Apache to use HTTP Basic authentication. HTTP Digest authentication is not currently supported.</td>
</tr>
<tr>
<td>AuthBasicProvider crowd</td>
<td>Tells Apache to delegate authentication to the Apache Crowd connector.</td>
</tr>
<tr>
<td>CrowdAppName myappname</td>
<td>Set 'myappname' to the application Apache should authenticate as.</td>
</tr>
<tr>
<td>CrowdAppPassword mypassword</td>
<td>Set 'mypassword' to the password for the application.</td>
</tr>
</tbody>
</table>
Require valid-user | Tells Apache that clients must provide a valid username/password to access the location.

The following configuration commands are optional, and can be used to customize your configuration further:

<table>
<thead>
<tr>
<th>Command</th>
<th>Explanation</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CrowdAcceptSSO Off</td>
<td>When set to 'On', the Apache Crowd connector will attempt to validate single sign-on (SSO) tokens provided in requests, avoiding the need for the user to log in if they have already logged in to another application.</td>
<td>On</td>
</tr>
<tr>
<td>CrowdCreateSSO Off</td>
<td>When set to 'On', the Apache Crowd connector will create a single sign-on (SSO) token whenever a user successfully authenticates, avoiding the need for the user to log in to other applications.</td>
<td>On</td>
</tr>
<tr>
<td>CrowdBasicAuthEncoding</td>
<td>Sets the list of character encoding schemes that the Apache Crowd connector will use to decode usernames and passwords. Each is tried in turn, until authentication succeeds. This setting may need to be changed if you have users with non-ASCII characters in their usernames or passwords, as browsers differ in the encoding schemes they use. Note that when an authentication attempt fails with one or more encodings before succeeding with another, the failures may still be counted and logged as failures by the directory.</td>
<td>ISO-8859-1</td>
</tr>
<tr>
<td>CrowdTimeout 5</td>
<td>The maximum number of seconds that the Apache Crowd connector should wait for a response from Crowd. If set to 0, the connector will wait indefinitely.</td>
<td>0</td>
</tr>
<tr>
<td>CrowdCacheMaxAge 120</td>
<td>The maximum number of seconds that a response from Crowd will be cached by the Apache Crowd connector.</td>
<td>60</td>
</tr>
<tr>
<td>CrowdCacheMaxEntries 1000</td>
<td>The maximum number of entries cached at any time by the Apache Crowd connector. If set to 0, caching is disabled.</td>
<td>500</td>
</tr>
</tbody>
</table>

For more detail about Apache configuration, please refer to the [Apache documentation](https://httpd.apache.org/docs/).  

**Step 5. Configuring Authorization**

If you want to restrict access to a certain Apache `<Directory>` or `<Location>`, so that only a subset of Crowd users and/or groups have permissions, add the following lines to your configuration:

```xml
<Location URL_to_restrict>
  .
  .
  .
  Require user johnh kevinr
  Require group developers crowd-administrators
  AuthzUserAuthoritative Off
  .
  .
</Location>
```

Note that you must also remove any `Require valid-user` command from this `<Directory>` or `<Location>` for the new restrictions to take effect.
Documentation for Crowd 2.11

**Command** | **Explanation**
---|---
**Require user johnh kevinr** | Allow the users johnh or kevinr to access the location.  
**Require group developers crowd-administrators** | Allow members of the developers or crowd-administrators groups to access the location.

If both 'Require user' and 'Require group' are used, these are combined using OR rather than AND, so any user that authenticates successfully will be able to access the resource.

If you are using "Require user" and "Require group" directives together, you will need to add the following setting:

**Command** | **Explanation** | **Default**
---|---|---
**AuthzUserAuthoritative Off** | When set to 'On', authorization decisions made by mod_authz_user on the basis of "Require user" directives are final. When set to 'Off', they may be overruled by other Apache authorization providers. | On

If you have configured authorization providers in addition to the Crowd Apache connector, you may need to add the following optional setting:

**Command** | **Explanation** | **Default**
---|---|---
**AuthzCrowdAuthoritative Off** | When set to 'On', authorization decisions made by Crowd are final. When set to 'Off', they may be overruled by other Apache authorization providers. | On

**Step 6. Configuring Subversion (Optional)**

If you are using Subversion under Apache, Crowd's Subversion connector allows you to password-protect a Subversion repository and provide fine-grained access control by group or user.

Follow the instructions on integrating Crowd with Subversion.

**Notes**

- Typically, only one of the Require user or Require group commands is needed for a particular location. You can define both. If you do, then access is granted if either is satisfied.
- If the CrowdCacheMaxEntries setting is missing or set to a non-zero value, then requests to Crowd are cached in order to increase performance. This means that changes to passwords, group membership and session expiry in Crowd may not be reflected immediately in user access.
- 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.

**CrowdURL** https://localhost:8095/crowd/

For information on how to secure Crowd connections, refer to the documentation on configuring Crowd to work with SSL.

- If you are using Crowd 2.0 or earlier, you need to follow the instructions for Crowd 2.0.

**RELATED TOPICS**

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Crowd Documentation

Disabling Previous Versions of the Crowd Apache Connector

This page provides instructions on how to disable older versions (1.3 or earlier) of the Crowd Apache Connector in preparation for installation of version 2.0 of the Connector. These instructions are part of the guide to integrating Crowd with Apache.

Procedure

1. Locate your Apache configuration file(s). On most systems, you will find these in /etc/httpd/conf, and possibly also in /etc/httpd/conf.d.
2. Open each of the configuration files in an editor, and place a hash character (#) at the beginning of any line that starts with one of the following phrases:
   - `PerlAuthenHandler Apache::CrowdAuth`
   - `PerlSetVar CrowdAppName`
   - `PerlSetVar CrowdAppPassword`
   - `PerlSetVar CrowdSOAPURL`
   - `PerlSetVar CrowdCacheEnabled`
   - `PerlSetVar CrowdCacheLocation`
   - `PerlSetVar CrowdCacheExpiry`
3. Save your changes to the Apache configuration files.

Now that the previous version has been disabled, the next step is to install the new Crowd Apache Connector packages.

RELATED TOPICS

Integrating Crowd with Apache

Installing the Crowd Apache Connector on CentOS Linux

This page provides instructions on how to install the Crowd Apache connector on a computer using CentOS Linux. These instructions are part of the guide to integrating Crowd with Apache.

The intent of these instructions is to take you from a default OS installation to a working Apache/Subversion/Crowd integration as easily as possible. We assume a fresh installation. If you are an experienced Linux system administrator you need not follow these instructions to the letter.

1. Determine which Package You Need

Identify the package you require by looking up your version of CentOS Linux and your processor architecture in the table below.

If you are unsure of your processor architecture, you can determine it by entering the command "uname -p" in a terminal.

<table>
<thead>
<tr>
<th>CentOS Linux Version</th>
<th>i386</th>
<th>x86_64</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.5</td>
<td>Build from source</td>
<td>rhel5.5/mod_authnz_crowd-2.2.2-1.x86_64.rpm</td>
<td>Build from source</td>
</tr>
<tr>
<td>other</td>
<td>Build from source</td>
<td>Build from source</td>
<td>Build from source</td>
</tr>
</tbody>
</table>

* ‘Build from source’ means that there is no binary package available for your platform. Rather than following the instructions on this page, you should follow the instructions for installing the Crowd Apache Connector on other UNIX-like systems.
2. Install the Crowd Apache Connector Packages

1. Download the package by entering the following command at a terminal, substituting PACKAGE_RELATIVE_URL with the appropriate relative URL from the table in step 1:

   ```
   wget http://downloads.atlassian.com/software/crowd/downloads/cwdapache/packages/PACKAGE_RELATIVE_URL
   ```

2. Start installation of the package by entering the following command at a terminal, substituting PACKAGE_FILE with the filename component of the package URL:

   ```
   su -c "yum -y --nogpgcheck install PACKAGE_FILE"
   ```

3. When prompted, enter the root user password.
4. Everything you need should now be installed and Apache should restart. If Apache fails to start, check the /var/log/httpd/error_log file.

Now that the software is installed, the next step is to configure Apache authentication.

Installing the Crowd Apache Connector on Red Hat Enterprise Linux

This page provides instructions on how to install the Crowd Apache connector on a computer using Red Hat Enterprise Linux. These instructions are part of the guide to integrating Crowd with Apache.

The intent of these instructions is to take you from a default OS installation to a working Apache/Subversion/Crowd integration as easily as possible. We assume a fresh installation. If you are an experienced Linux system administrator you need not follow these instructions to the letter.

1. Determine which Package You Need

Identify the package you require by looking up your version of Red Hat Enterprise Linux and your processor architecture in the table below.

If you are unsure of your processor architecture, you can determine it by entering the command "uname -p" in a terminal.

<table>
<thead>
<tr>
<th>Red Hat Enterprise Linux Version</th>
<th>i386</th>
<th>x86_64</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Build from source*</td>
<td>rhel6/mod_authnz_crowd-2.2.2-1.el6.x86_64.rpm</td>
<td>Build from source*</td>
</tr>
<tr>
<td>5.5</td>
<td>Build from source*</td>
<td>rhel5.5/mod_authnz_crowd-2.2.2-1.x86_64.rpm</td>
<td>Build from source*</td>
</tr>
<tr>
<td>other</td>
<td>Build from source*</td>
<td>Build from source*</td>
<td>Build from source*</td>
</tr>
</tbody>
</table>

*Build from source* means that there is no binary package available for your platform. Rather than following the instructions on this page, you should follow the instructions for installing the Crowd Apache connector on other UNIX-like systems.

2. Subscribe to the Red Hat Network

Ensure that your system has an active subscription to the Red Hat Network. This is required so that packages upon which the Apache Connector depends can be downloaded from Red Hat and installed.
If it has an active subscription, the system will appear in the list of Red Hat Network systems.
If your system does not have an active subscription, you can register it by entering the command "su -c rhn_register" in a terminal on the affected system. Enter the root password when prompted and follow the instructions that appear.

3. (Red Hat Enterprise Linux 6 only) Subscribe to the Optional Software Channel

   This step is not required for Red Hat Enterprise Linux 5.5.

This is required for installation of some of the packages upon which the Apache Connector depends.

1. Visit the page for your system on the Red Hat Network by clicking its name in the list of systems.
2. Click the 'Alter Channel Subscriptions' link.
3. If the checkbox for 'RHEL Server Optional' is not already checked, check it and click the 'Change Subscriptions' button.

4. Install the Crowd Apache Connector Packages

1. Download the package by entering the following command at a terminal, substituting PACKAGE_RELATIVE_URL with the appropriate relative URL from the table in step 1:

   ```
   wget https://downloads.atlassian.com/software/crowd/downloads/cwdapache/packages/PACKAGE_RELATIVE_URL
   ```

2. Start installation of the package by entering the following command at a terminal, substituting PACKAGE_FILE with the filename component of the package URL:

   ```
   su -c "yum -y --nogpgcheck install PACKAGE_FILE"
   ```

3. When prompted, enter the root user password.
4. Everything you need should now be installed and Apache should restart. If Apache fails to start, check the file /var/log/httpd/error_log.

Now that the software is installed, the next step is to configure Apache authentication.

Installing the Crowd Apache Connector on Ubuntu Linux

The following instructions have been tested on Ubuntu 11.04. Other versions may require variations to this procedure.

Procedure

1. Open a terminal on the system, change to a suitable working directory, and enter the following command:

   ```
   sudo apt-get -y install autoconf libtool apache2-threaded-dev libsvn-dev libcurl4-gnutls-dev libxml2-dev apache2-mpm-worker libapache2-svn automake
   ```

2. Enter your password when prompted.
3. Enter the following commands:
If you get this error after running ./configure:

```
configure: error: Could not locate Apache configuration file
```

Refer to this KB for resolution: "Could not locate Apache configuration file" Error When Attempting to Install the Crowd Apache Connector on Ubuntu Linux

4. Enter your password if prompted.
5. Everything you need should now be installed and Apache should restart. If Apache fails to start, check the file. /var/log/apache2/error.log

Now that the software is installed, the next step is to configure Apache authentication.

Installing the Crowd Apache Connector on Debian

The following instructions have been tested on Debian 6.0.1a. Other versions may require variations to this procedure.

**Procedure**

1. Open a terminal on the system, change to a suitable working directory, and enter the following command:

```
su -c "apt-get -y install autoconf libtool apache2-threaded-dev libsvn-dev libcurl4-dev libxml2-dev apache2-mpm-worker libapache2-svn make"
```

2. Enter the root password when prompted.
3. Enter the following commands:

```
wget http://downloads.atlassian.com/software/crowd/downloads/cwdapache/mod_authnz_crowd-2.2.2.tar.gz
tar xzf mod_authnz_crowd-2.2.2.tar.gz
cd mod_authnz_crowd-2.2.2
autoreconf --install
./configure
make
su -c "make install"
```

4. Enter the root password when prompted.
5. Everything you need should now be installed and Apache should restart. If Apache fails to start, check the file. /var/log/apache2/error.log file.

Now that the software is installed, the next step is to configure Apache authentication.
Installing the Crowd Apache Connector on Other UNIX-Like Systems

The following instructions have been tested on Red Hat Enterprise Linux 6 Server. Other platforms may require variations to this procedure.

Procedure

1. Open a terminal on the system, change to a suitable working directory, and enter the following command:

   ```
   su -c "yum -y install autoconf automake gcc httpd-devel
   libcurl-devel libtool libxml2-devel mod_dav_svn subversion-devel"
   ```

2. Enter the root password when prompted.
3. Enter the following commands:

   ```
   wget http://downloads.atlassian.com/software/crowd/downloads/cwdapache/mod_authnz_crowd-2.2.2.tar.gz
   tar xzf mod_authnz_crowd-2.2.2.tar.gz
   cd mod_authnz_crowd-2.2.2
   autoreconf --install
   ./configure
   make
   su -c "make install"
   ```

4. Enter the root password when prompted.
5. Everything you need should now be installed and Apache should restart. If Apache fails to start, check the file
/var/log/httpd/error_log

Now that the software is installed, the next step is to configure Apache authentication.

Installing the Crowd Apache Connector on Windows

Version 2.0 of the Crowd Apache Connector is not yet available for Windows platforms.

Want to stay informed? Please log in to this Confluence site (click ‘Log In’ or ‘Sign Up’ at the top right of this page) and ‘watch’ this page (open the ‘Tools’ menu and select ‘Watch’) to be notified when version 2.0 is made available for Windows.

Until that time, you can continue to use version 1.3 of the Crowd Apache Connector with Crowd 2.1 by following these instructions from the Crowd 2.0 documentation.

Integrating Crowd with Jive Forums

Jive Forums allows you to specify an implementation that provides authentication and authorization 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 centralized 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 Forum 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 authorizations 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 summarized below:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/crowd-integration-client-X.X.X.jar</td>
<td>JIVEFORUMS/WEB-INF/lib</td>
</tr>
<tr>
<td>CROWD/client/lib/log4j-1.2.8.jar</td>
<td>JIVEFORUMS/WEB-INF/lib/</td>
</tr>
<tr>
<td>CROWD/client/lib/ehcache-1.2.3.jar</td>
<td>JIVEFORUMS/WEB-INF/lib/</td>
</tr>
<tr>
<td>CROWD/client/conf/crowd.properties</td>
<td>JIVEFORUMS/WEB-INF/classes/</td>
</tr>
<tr>
<td>CROWD/client/conf/crowd-ehcache.xml</td>
<td>JIVEFORUMS/WEB-INF/classes/</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/xfire*.jar</td>
<td>JIVEFORUMS/WEB-INF/lib</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>jiveforums</td>
</tr>
<tr>
<td>application.password</td>
<td>set a password</td>
</tr>
</tbody>
</table>

   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:

   ```xml
   <jive>
   <!-- When setup is false, you can access the setup tool. -->
   <setup>false</setup>
   ...  
   <!-- Allow SSO login for admins -->
   <admin>
     <tryAlternativeLogin>true</tryAlternativeLogin>
   </admin>
   </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':

![Screenshot of Jive Forums Setup]

7. You should be at the 'Custom User System' screen. Enter the following details which specify Crowd as the custom authenticator:

![Screenshot of Custom User System]

UserManager implementation:

```java
com.atlassian.crowd.integration.jive.CrowdUserManager
```

GroupManager implementation:

If you would like Crowd to manage your user groups, add the following group manager:

```java
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:

```
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**


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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
    - Installing the Crowd Apache Connector on CentOS Linux
    - Installing the Crowd Apache Connector on Red Hat Enterprise Linux
    - Installing the Crowd Apache Connector on Ubuntu Linux
    - Installing the Crowd Apache Connector on Debian
    - Installing the Crowd Apache Connector on Other UNIX-Like Systems
    - Installing the Crowd Apache Connector on Windows
  - 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
Integrating Crowd with Atlassian HipChat
Integrating Crowd with Atlassian FishEye - DRAFT

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
  Viewing Users in Directories Mapped to an Application
  Specifying which Groups can access an Application

Effective memberships with multiple directories
Specifying an Application's Address or Hostname
Testing a User's Login to an Application
Enforcing Lower-Case Usernames and Groups 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 authorized to authenticate. This user also needs to NOT be on any user/IP ban lists within the Jive application. The Crowd integration will honor 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 utilize Crowd's
"remember me", so that "remember me" is centralized.

- 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 honored 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 recognizes 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 Confluence 3.4 or earlier
  - Updating Files in a Confluence Evaluation Distribution
- Integrating Crowd with Atlassian CrowdID
- Integrating Crowd with Atlassian Crucible
- Integrating Crowd with Atlassian FishEye
  - Configuring FishEye earlier than 4.0 with Crowd
- Integrating Crowd with Atlassian JIRA
  - Integrating Crowd with Atlassian JIRA 4.2 or earlier
- Integrating Crowd with Atlassian Bitbucket Server
- Integrating Crowd with Acegi Security
  - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
- Integrating Crowd with Apache
  - Disabling Previous Versions of the Crowd Apache Connector
  - Installing the Crowd Apache Connector on CentOS Linux
  - Installing the Crowd Apache Connector on Red Hat Enterprise Linux
  - Installing the Crowd Apache Connector on Ubuntu Linux
  - Installing the Crowd Apache Connector on Debian
  - Installing the Crowd Apache Connector on Other UNIX-Like Systems
  - Installing the Crowd Apache Connector on Windows
- 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
- Integrating Crowd with Atlassian HipChat
- Integrating Crowd with Atlassian FishEye - DRAFT
- 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
  - Viewing Users in Directories Mapped to an Application
  - Specifying which Groups can access an Application
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups for an Application
- Managing an Application's Session
- Deleting or Deactivating an Application
- Configuring Caching for an Application
- Overview of SSO
Integrating Crowd with Spring Security

Crowd provides **centralized 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 authorization for J2EE applications.

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

**Spring, Acegi and Crowd versions**
Spring Security was formerly known as Acegi. There is a separate tutorial for integrating Acegi with Crowd. The connector is developed and tested with **Spring Security 3.1 from Crowd 2.5 and later**. Please use a previous supported release of Crowd if you require compatibility 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 2.5 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 a 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:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/crowd-integration-client-X.X.X.jar</td>
<td>SpringSecApp/WEB-INF/lib</td>
</tr>
<tr>
<td>CROWD/client/lib/*.jar</td>
<td>SpringSecApp/WEB-INF/lib</td>
</tr>
</tbody>
</table>

2.1.2 Adding the Crowd Spring Security Connector as a Maven Dependency

Follow either 2.1.1 or 2.1.2 (not both).

Add to your pom.xml:

```xml
<properties>
    <crowd.version>2.5.0</crowd.version>
    <spring.version>3.1.0.RELEASE</spring.version>
</properties>

<dependencies>
    ...

    <dependency>
        <groupId>com.atlassian.crowd</groupId>
        <artifactId>crowd-integration-springsecurity</artifactId>
        <version>${crowd.version}</version>
        <scope>runtime</scope>
    </dependency>

    <!-- Crowd needs at runtime -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>${spring.version}</version>
        <scope>runtime</scope>
    </dependency>

    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>${spring.version}</version>
        <scope>runtime</scope>
    </dependency>

    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-beans</artifactId>
        <version>${spring.version}</version>
        <scope>runtime</scope>
    </dependency>

    ...
</dependencies>
```

Ensure you have dependencies on the spring- modules to pick up the versions of Spring required by Crowd rather than the possibly lower version specified by Spring Security.

2.2 Adding the Cache Configuration File
Copy the following file into your application's classpath:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/conf/crowd-ehcache.xml</td>
<td>SpringSecApp/WEB-INF/classes/crowd-ehcache.xml</td>
</tr>
</tbody>
</table>

This file can be tweaked to change the cache behavior.

### 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:

<table>
<thead>
<tr>
<th>Copy From</th>
<th>Copy To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CROWD/client/conf/crowd.properties</td>
<td>SpringSecApp/WEB-INF/classes</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>Same as application name defined when adding the application to Crowd in Step 1.</td>
</tr>
<tr>
<td>application.password</td>
<td>Same as application password defined when adding the application to Crowd in Step 1.</td>
</tr>
<tr>
<td>crowd.server.url</td>
<td><a href="http://localhost:8095/crowd/services/">http://localhost:8095/crowd/services/</a></td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>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.</td>
</tr>
</tbody>
</table>

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:

- **Centralized 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 centralized user repository for retrieving user details as well as performing authentication.
- **Single sign-on**: In addition to centralized 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 behavior 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-CrowdRestClient.xml` Spring configuration file, found in `crowd-integration-client-rest.jar`. For example, if you are configuring Spring using a context listener, you can add the following parameter in your Spring Security application's `WEB-INF/web.xml`:
3.1 Configuring Centralized 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:

   ```
   <bean id="crowdUserDetailsService"
        class="com.atlassian.crowd.integration.springsecurity.user.CrowdUserDetailsServiceImpl">
      <property name="crowdClient" ref="crowdClient"/>
      <property name="authorityPrefix" value="ROLE_"/>
   </bean>
   ```

2. Add the definition of the RemoteCrowdAuthenticationProvider:

   ```
   <bean id="crowdAuthenticationProvider"
        class="com.atlassian.crowd.integration.springsecurity.RemoteCrowdAuthenticationProvider">
      <constructor-arg ref="crowdClient"/>
      <constructor-arg ref="crowdHttpAuthenticator"/>
      <constructor-arg ref="crowdUserDetailsService"/>
   </bean>
   ```

Controlling granted authority names

Rather than taking the group name and setting a prefix, you can define a mapping to grant specific authorities when a user belongs to Crowd groups:

```
<util:map id="groupToAuthorityMappings">
  <beans:entry key="crowd-administrators" value="ROLE_crowd-administrators" />
  <beans:entry key="some-other-group" value="specific-authority-for-other-group" />
</util:map>
```

and then set it on the `crowdUserDetailsService`:
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 centralized user management
Single sign-on is optional. If you wish to configure SSO you must first configure centralized 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.
   c. Include `custom-filters` for login and logout.
      You should end up with `http` elements similar to this:
2. Change the default processing filter to Crowd's SSO filter by adding the following bean definitions:

```xml
<http auto-config="false"
entry-point-ref="crowdAuthenticationProcessingFilterEntryPoint">
  <custom-filter position="FORM_LOGIN_FILTER"
  ref='authenticationProcessingFilter'/>
  <custom-filter position="LOGOUT_FILTER"
  ref='logoutFilter'/>

  <intercept-url pattern="/admin/*"
  access="ROLE_application-administrators"/>
  <intercept-url pattern="/passwordHint.html"
  access="ROLE_ANONYMOUS,ROLE_ADMIN,ROLE_USER"/>
  <intercept-url pattern="/**/*.html*"
  access="IS_AUTHENTICATED_FULLY"/>
</http>
```
Add the definition of the CrowdLogoutHandler and add in a LogoutFilter that references it:

3. Add the definition of the CrowdLogoutHandler and add in a LogoutFilter that references it:
Step 4. Restarting your Spring Security Application

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

Authorization

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 authorization features, users in Crowd will have their Spring Security roles specified by their group names.

The authorities granted will use the authorityPrefix specified on crowdUserDetailsService. If no suffix is specified, the authorities will append the Crowd group name.

For example if user 'admin' is in the 'crowd-admin' group, then the user 'admin' will be authorized to view pages restricted to the 'ROLE_crowd-admin' role in Spring Security.

    <http>
    ...  
    <intercept-url pattern="/console/secure/**" access="ROLE_crowd-administrators"/>
    <intercept-url pattern="/console/info/**" access="ROLE_crowd-users"/>  
    <intercept-url pattern="/console/user/**" access="IS_AUTHENTICATED_FULLY"/>  
    ...
    </http>

If authoritySuffix is also specified, any user in the mapped groups configured in crowd will be granted "authorityPrefix + authoritySuffix" (for example, ROLE_ADMIN).
<beans:bean id="crowdUserDetailsService" ...>
  ...
  <beans:property name="authorityPrefix" value="ROLE_"/>
  <beans:property name="authorityPrefix" value="ADMIN"/>
</beans:bean>

<http>
  ...
  <intercept-url pattern="/console/secure/**" access="ROLE_ADMIN"/>
  <intercept-url pattern="/console/user/**" access="IS_AUTHENTICATED_FULLY"/>
  ...
</http>

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
- Effective memberships with multiple directories
- Specifying an Application’s Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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.

   ```
   mvn archetype:create -DarchetypeGroupId=org.appfuse.archetypes \
   -DarchetypeArtifactId=appfuse-basic-struts \
   -DremoteRepositories=http://static.appfuse.org/releases \
   -DarchetypeVersion=2.0.2 \
   -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.

   ```
   cd myproject
   mvn appfuse:full-source
   ```

3. Build it.

   ```
   mvn clean install
   ```

4. Run it.

   ```
   mvn jetty:run-war -Dmaven.test.skip
   ```

5. Play with it.

   ```
   http://localhost:8080/
   ```

6. Shut it down.

   ```
   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:

```xml
<dependencies>
  <dependency>
    <groupId>com.atlassian.crowd</groupId>
    <artifactId>crowd-integration-client</artifactId>
    <version>1.6</version>
  </dependency>
  ...
</dependencies>
```
You will also need to create the file `myproject/src/main/resources/crowd.properties`:

```
application.name                        appfuse
application.password                    password
application.login.url                   http://localhost:8095/crowd/
crowd.server.url
http://localhost:8095/crowd/services/
session.isauthenticated                 session.isauthenticated
session.tokenkey                        session.tokenkey
session.validationinterval              0
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 Centralized 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 centralized 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="crowd UserManager"/>
  <beans:property name="authorityPrefix" value="ROLE_"/>
</beans:bean>
```
2. Add the definition of the RemoteCrowdAuthenticationProvider that delegates Spring Security authentication requests to Crowd:

```xml
<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:

```xml
<!--
    <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
```


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 authorization 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 centralized authentication 😊

Applicati**on-level centralized user management**

One quirky 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 centralized user management, AppFuse will need to delegate its calls to create, retrieve, update and delete users to Crowd using Crowd's remote API. This will prevent data redundancy and eliminate the hassle of data synchronization. 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`:

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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:

     ```
     <http lowercase-comparisons="false"
     entry-point-ref="crowdAuthenticationProcessingFilterEntryPoint"/>
     <!-- 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>
     ```

2. 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>

3. 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>

4. Now repeat:

mvn jetty:run-war -Dmaven.test.skip=true
SSO will only work for users that are able to authenticate with both applications and are authorized 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

The Crowd Apache/Subversion Connector will still be available, but will not be supported by Atlassian beyond 31 December 2014.

This page provides guidance on how to use Subversion's authorization module to integrate Crowd and Subversion.

If you were already using the Crowd Apache/Subversion Connector before 31 December 2014 and have been experiencing problems, we recommend switching to use Subversion's authentication module: Alternative setup for existing Crowd Apache/Subversion connector users.

If you have integrated Crowd with Apache you can then use Crowd to password-protect a Subversion repository and provide fine-grained access by group or user.

The following features are supported:

- Authentication: Use Crowd to password-protect your Subversion repository.
- Authorization: Provide fine-grained access by group or user.

**Step 1. Integrating Crowd with Apache**

First you will need to integrate Crowd with Apache. Please follow the instructions on integrating Crowd with Apache.

Note that you do not need to define Subversion as an application in Crowd. Subversion and Apache will both use the same Crowd application.

**Step 2. 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 (instructions) then you can use the same configuration method to delegate Subversion’s user authentication to Crowd.

**Example:**
<Location /svn>

AuthName "Atlassian Crowd"
AuthType Basic
AuthBasicProvider crowd

CrowdAppName myappname
CrowdAppPassword mypassword
CrowdURL http://localhost:8095/crowd/

CrowdCreateSSO off  # Improves performance when using Subversion
clients that don't store cookies

DAV svn

# Set this to the path to your repository
SVNPath /var/lib/svn

# The following three lines allow anonymous read, but make
# committers authenticate themselves.
<LimitExcept GET PROPFIND OPTIONS REPORT>
  Require valid-user
</LimitExcept>

</Location>

If your Subversion clients do not store cookies, you will achieve much better performance by disabling single
sign-on with the "CrowdCreateSSO off" directive. This will avoid a request being made to Crowd for every single
request to Subversion.

You will also need to edit your Apache configuration file to load Subversion's authorization module mod_authz_
svn.

LoadModule authz_svn_module modules/mod_authz_svn.so

Note that you will need to restart Apache before any changes to its configuration files will take effect.

*Step 3. Configuring Crowd Authorization for Subversion*

To restrict Subversion repository access to certain groups and/or users, you can add the Require group and
Require user directives, described in the page on integrating Crowd with Apache.

For more fine-grained access, the AuthzSVNAccessFile directive allows you to define path-based access
rules.

*Example:*
<Location /svn>
AuthName "Atlassian Crowd"
AuthType Basic
AuthBasicProvider crowd

CrowdAppName myappname
CrowdAppPassword mypassword
CrowdURL http://localhost:8095/crowd/

CrowdCreateSSO off  # Improves performance when using Subversion clients that don't store cookies

DAV svn

# Set this to the path to your repository
SVNPath /var/lib/svn

AuthzSVNAccessFile /etc/apache2/dav_svn.authz
Require valid-user
</Location>

The `AuthzSVNAccessFile` setting lets you define a file where you can configure group and user access at directory level.

Here is a short example:
[groups]
# Groups referred to in other sections must be listed here, but group membership is obtained from Crowd.
bazdevelopers =
foodevelopers =

# 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 =

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.
- If you make use of the SVNParentPath directive to configure multiple repositories, repository paths should be specified in the form repository:path, for example:
Mixing Authenticated and Anonymous Access

A common requirement for Subversion access is to have a combination of anonymous access (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 line to the Apache configuration file:

```
AuthzSVNAccessFile /etc/apache2/dav_svn.authz
AuthzSVNNoAuthWhenAnonymousAllowed On
Satisfy Any
Require valid-user
```

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 AuthzSVNAccessFile file. For example, if you wanted to allow anonymous read access to most of a repository but require authentication for a private section, the AuthzSVNAccessFile file would look like this:

```
[groups]
developers=

# 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.

Additional Configuration Options

You may customize your configuration further with some optional commands. See the Subversion documentation for more information.

Step 4. Getting memberships from Crowd
Next you need to take the AuthzSVNAccessFile file and expand it with memberships from Crowd. This is done with a python script. The script takes your definitions in terms of Crowd groups, expands it out and generates the access-file-expanded.authz file (containing the list of users and groups from your Crowd instance).

1. Download the python script from Bitbucket.
2. In the Apache httpd.conf file edit the AuthzSVNAccessFile setting to point to where the access-file-expanded.authz file will be located (this is the file that will be generated / updated by the script).
3. Copy your crowd.properties file to Apache's Config directory.
4. Create a script with the following:

   ```bash
   if generate-authz-svn-access-file.py --config crowd.properties --check-event-token access-file-expanded.authz; then
     : # Do nothing; file is current
   else
     generate-authz-svn-access-file.py --config crowd.properties access-file-template.authz >access-file-expanded.authz.tmp && mv access-file-expanded.authz.tmp access-file-expanded.authz
   fi
   
   Edit the example above so that the file paths are correct for your environment.
5. Run the script and check that the access-file-expanded.authz file is updated.
6. Create a scheduled task, using cron or some alternative, to run the script regularly, for example hourly.

Notes

- Note that group memberships specified in the [groups] section of the file described in the Subversion documentation are ignored by the Crowd Apache connector, because group memberships come from Crowd. However, any groups referred to in other sections must be named here.
- Subversion will convert all group names specified in this file to lower case before comparing them to group names supplied by Crowd. For this reason, you must enable the "Lower Case Output" option for your Subversion application.

Alternative setup for existing Crowd Apache/Subversion connector users

The Crowd Apache/Subversion Connector will not be supported by Atlassian beyond 31 December 2014.

If you are experiencing problems with your existing setup, we recommend moving to the following alternative by using Subversion's authentication module to generate a AuthzSVNAccessFile and populate it with memberships from Crowd.

A python script takes your definitions, in terms of Crowd groups, from the AuthzSVNAccessFile, expands it out and generates the access-file-expanded.authz file, containing the list of users and groups from your Crowd instance. You can then automate running this script so that the mapping files used by Subversion are kept up to date.

1. Download the python script from Bitbucket
2. Edit your Apache httpd.conf configuration file to load mod_authz_svn (instead of the Atlassian Crowd Apache Subversion Connector)

   ```bash
   LoadModule authz_svn_module modules/mod_authz_svn.so
   
   3. In the Apache httpd.conf file edit the AuthzSVNAccessFile setting to point to where the access-file-expanded.authz file will be located (this is the file that will be generated / updated by the script).
4. Copy your crowd.properties file to Apache's Config directory.
5. Create a script with the following:
if generate-authz-svn-access-file.py --config crowd.properties --check-event-token access-file-expanded.authz; then
    : # Do nothing; file is current
else
    generate-authz-svn-access-file.py --config crowd.properties access-file-template.authz >access-file-expanded.authz.tmp && mv access-file-expanded.authz.tmp access-file-expanded.authz
fi

Edit the example above so that the file paths are correct for your environment.
6. Run the script and check that the access-file-expanded.authz file is updated.
7. Create a scheduled task, using cron or some alternative, to run the script regularly, for example hourly.

RELATED TOPICS

- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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 REST APIs.

For assistance, please see the developer's guide to 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's classpath, typically WEB-INF/lib. These files will be in Crowd's client folder, with a name similar to crowd-integration-client-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/class es 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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
    - Installing the Crowd Apache Connector on CentOS Linux
    - Installing the Crowd Apache Connector on Red Hat Enterprise Linux
    - Installing the Crowd Apache Connector on Ubuntu Linux
    - Installing the Crowd Apache Connector on Debian
    - Installing the Crowd Apache Connector on Other UNIX-Like Systems
    - Installing the Crowd Apache Connector on Windows
  - 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
  - Integrating Crowd with Atlassian HipChat
  - Integrating Crowd with Atlassian FishEye - DRAFT
- 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
  - Viewing Users in Directories Mapped to an Application
  - Specifying which Groups can access an Application
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
Crowd Documentation

Integrating Crowd with Atlassian HipChat

You can connect HipChat Server to Atlassian Crowd or to a JIRA application (version 4.3 or later) for management of users and for authentication (verification of a user's login).

Connecting HipChat Server to Crowd

You can connect HipChat Server to Atlassian Crowd for user management and authentication. For more information on Crowd, see the Crowd Administration Guide.

When to use this option: Connect to Crowd if you want to import and synchronize users from and authenticate those users against Crowd.

To connect HipChat Server to Crowd:

1. Go to your Crowd Administration Console and define the HipChat Server to Crowd. See the Crowd documentation: Adding an Application.
2. Browse to your server's fully qualified domain name, for example https://hipchat.yourcompany.com/.
3. Log into the HipChat Server web user interface (UI) using your administrator email and password.
4. Click Server admin > Directory.
5. Add a directory and select type 'Atlassian Crowd'. Enter the settings as described below.
6. Save the directory settings.
7. Define the directory order by clicking the blue up and down arrows next to each directory on the 'Use r Directories' screen. Here is a summary of how the directory order affects the processing:
   - The order of the directories is the order in which they will be searched for users and groups.
   - Changes to users and groups will be made only in the first directory where the application has permission to make changes.

Settings in HipChat Server for the Crowd Directory Type

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A meaningful name that will help you to identify this Crowd server amongst your list of directory servers. Examples:</td>
</tr>
<tr>
<td></td>
<td>• Crowd Server</td>
</tr>
<tr>
<td></td>
<td>• Example Company Crowd</td>
</tr>
</tbody>
</table>
### Server URL
The web address of your Crowd console server. Examples:
- `http://www.example.com:8095/crowd/`
- `http://crowd.example.com`

### Application Name
The name of your application, as recognized by your Crowd server. Note that you will need to define the application in Crowd too, using the Crowd administration Console. See the Crowd documentation on adding an application.

### Application Password
The password which the application will use when it authenticates against the Crowd framework as a client. This must be the same as the password you have registered in Crowd for this application. See the Crowd documentation on adding an application.

### Crowd Permissions

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| Read Only     | The users, groups and memberships in this directory are retrieved from Crowd and can only be modified via Crowd. You cannot modify Crowd users, groups or memberships via the application administration screens.  
**Note:** There are no user group structures within HipChat Server. Group data is synchronized, but not used. |
| Read/Write    | Not applicable to HipChat Server. |

### Advanced Crowd Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| Enable Nested Groups          | Not applicable to HipChat Server.  
When you sync your users, you'll sync every user that exists in the directory. |
| Synchronization Interval      | Synchronization is the process by which the application updates its internal store of user data to agree with the data on the directory server. The application will send a request to your directory server every x minutes, where ‘x’ is the number specified here. The default value is 60 minutes. |

### Connecting HipChat Server to JIRA applications
Subject to certain limitations, you can connect a number of Atlassian applications to a single JIRA application for centralized user management.

**When to use this option:** You can connect to a server running **JIRA 4.3** or later, **JIRA Software 7.0** or later, **JIRA Core 7.0** or later, or **JIRA Service Desk 3.0** or later. Choose this option as an alternative to Atlassian Crowd, for simple configurations with a limited number of users.

You can't filter users. When you sync users, you'll sync all users in the JIRA application. HipChat Server marks inactive users from the JIRA application as deactivated users and doesn't count them towards your license.

### To connect HipChat Server to a JIRA application:
1. Configure the JIRA application to recognize HipChat Server:
   a. Log in to the JIRA application as a user with the ‘**JIRA Administrators**’ global permission.  
   b. Choose **crowd/** in the application's settings.
1. b. c. d. e. f. Configure HipChat Server to connect to a JIRA application:
   a. Browse to your server's fully qualified domain name, for example https://hipchat.yourcompany.com/.
   b. Log into the HipChat Server web user interface (UI) using your administrator email and password.
   c. Click Server admin > Directory.
   d. Add a directory and select type ‘Atlassian JIRA’.
   e. Enter the settings as described below. When asked for the application name and password, enter the values that you defined in the settings on the JIRA application.
   f. Save the directory settings.
   g. Define the directory order by clicking the blue up- and down-arrows next to each directory on the ‘User Directories’ screen.

Here is a summary of how the directory order affects the processing:
- The order of the directories is the order in which they will be searched for users.
- Changes to users will be made only in the first directory where the application has permission to make changes.

For details see Managing Multiple Directories.

### Settings for the JIRA Directory Type

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A meaningful name that will help you to identify this JIRA server in the list of directory servers. Examples:</td>
</tr>
<tr>
<td></td>
<td>• JIRA Service Desk Server</td>
</tr>
<tr>
<td></td>
<td>• My Company JIRA</td>
</tr>
<tr>
<td>Server URL</td>
<td>The web address of your JIRA server. Examples:</td>
</tr>
<tr>
<td></td>
<td>• <a href="http://www.example.com:8080">http://www.example.com:8080</a></td>
</tr>
<tr>
<td></td>
<td>• <a href="http://jira.example.com">http://jira.example.com</a></td>
</tr>
<tr>
<td>Application Name</td>
<td>The application name that HipChat Server will use when accessing the JIRA application.</td>
</tr>
<tr>
<td>Application Password</td>
<td>The password that HipChat Server will use when accessing the JIRA server that acts as user manager.</td>
</tr>
</tbody>
</table>

### Permissions for the JIRA Directory Type

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Only</td>
<td>The users, groups and memberships in this directory are retrieved from the JIRA server that is acting as user manager. They can only be modified via that JIRA server.</td>
</tr>
</tbody>
</table>

**Note:** There are no user group structures within HipChat Server. Group data is synchronized, but not used.
### Setting

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>Not applicable to HipChat Server.</td>
</tr>
<tr>
<td>Synchronization Interval (minutes)</td>
<td>Synchronization is the process by which the application updates its internal store of user data to agree with the data on the directory server. The application will send a request to your directory server every x minutes, where ‘x’ is the number specified here. The default value is 60 minutes.</td>
</tr>
</tbody>
</table>

#### Diagrams of Some Possible Configurations

![Diagram: HipChat Server connecting to Crowd for user management.](image)

*Diagram above: HipChat Server connecting to Crowd for user management.*
Diagram above: HipChat Server connecting to a JIRA application for user management. The JIRA application does the user management, storing the user data in its internal directory.
Diagram above: HipChat Server connecting to a JIRA application for user management, with the JIRA application in turn connecting to an LDAP server.

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:
- Background
- Prerequisites
- Step 1. Configuring the Crowd Application, Directory and Group Details
- Step 2. Generating your SSO Keys
- Step 3. Configuring Google Apps to Recognize 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

Background

When people refer to Single sign-on (SSO) they are usually referring to two things.

- Authentication - which userid and which password is used to confirm that a user is who they say they are
- SSO - authenticating for one application means that you don't have to authenticate to access another application (at least for a while and if you are using the same browser)

So in the case of Google Apps, the authentication is your Google Apps userid and password. Google Apps handles the SSO part with its Google Accounts site. When you log into Gmail in the morning you are actually being asked to enter your authentication information at the Google Accounts site, which then redirects your browser back to Gmail if you successfully authenticate. The Google Accounts remembers that you successfully authenticated this morning so that when you go to a Google Docs page later on that day, the redirect happens again without needing to reauthenticate. This all happens unseen by most Google Apps users.

Now Google Apps has the ability to change where it goes for its SSO functionality. A Google Apps administrator can configure just their Google Apps instance to use a different SSO. This could be Crowd, or any other SSO service. Crowd then becomes the master SSO service instead of Google Accounts. This means that logging into Gmail in the morning will take you to a Crowd authentication screen, not the Google Accounts. The redirection back to Gmail after a successful authentication happens just as before.

However this is not how OnDemand integrates with Google Apps. In that case the SSO functionality remains with Google Accounts.

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. Click the link on the 'google-apps' application name.
4. If required, 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 that 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
   - On the Groups tab, use the Add button to select one or more groups of users.
7. Click the Permissions tab and set the directory permissions for the application.
8. If required, you can change the application options on the Options tab:
   - **Lower Case Output** — See Enforcing Lower-Case Usernames and Groups 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. In the Crowd Application Browser, as described in Step 1 above, click the Configuration tab for the Google Apps application.
2. Click Generate New Keys.

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.’
Step 3. Configuring Google Apps to Recognize 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 recognize Crowd:

1. Log in to your **Google Apps Dashboard** as a **Google Apps administrator**.
2. In Google Apps, click **Security**.
3. Click **Advanced Settings**.
4. Click **Set up single sign-on (SSO)**.
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 **Browse** under '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, check **Use a domain specific issuer** and enter any required network masks in Google Apps. Please refer to the Google Apps documentation for guidance on these settings.
8. Save your changes in Google Apps.
Step 4. Verifying that a User can Log in to Google Apps

It is a good idea now to check that your users can log in to Google Apps.

To test a user's authentication to Google Apps:

1. 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 behavior.
- 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.
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 authorizing a user's access request. Read more about users and groups.

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. Click View for the application you wish to map.
4. Click the Directories tab.
5. Select the new directory from the drop-down list and click Add.
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. Now choose how group memberships are aggregated over the directories you've just mapped. Group memberships are used to determine authorization permissions. See Effective memberships with multiple directories for more information.
On the Directories tab check Aggregate group memberships across directories to use the 'aggregating membership' scheme. When the checkbox is clear 'non-aggregating membership' is used.
8. 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 Update.
   OR:
   - To allow only specific groups of users within the directory to authenticate against the application, see Specifying which Groups can access an Application.
9. Next, you should define the application's ability to add or update users in the directory. Click the Permissions tab and set the directory permissions for the application.
   Note that updates are handled differently, depending on the membership aggregation scheme you selected in step 7. See Directory update operations.

Screenshot: ‘Application — Map Directories’

<table>
<thead>
<tr>
<th>Directory</th>
<th>Directory Order</th>
<th>Allow All to Authenticate</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customers</td>
<td>▼</td>
<td>False</td>
<td>Remove</td>
</tr>
<tr>
<td>Employees</td>
<td>▲</td>
<td>False</td>
<td>Remove</td>
</tr>
</tbody>
</table>

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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
• Integrating Crowd with Atlassian CrowdID
• Integrating Crowd with Atlassian Crucible
• Integrating Crowd with Atlassian FishEye
  • Configuring FishEye earlier than 4.0 with Crowd
• Integrating Crowd with Atlassian JIRA
  • Integrating Crowd with Atlassian JIRA 4.2 or earlier
• Integrating Crowd with Atlassian Bitbucket Server
• Integrating Crowd with Acegi Security
  • Integrating AppFuse - a Crowd-Acegi Integration Tutorial
• Integrating Crowd with Apache
  • Disabling Previous Versions of the Crowd Apache Connector
  • Installing the Crowd Apache Connector on CentOS Linux
  • Installing the Crowd Apache Connector on Red Hat Enterprise Linux
  • Installing the Crowd Apache Connector on Ubuntu Linux
  • Installing the Crowd Apache Connector on Debian
  • Installing the Crowd Apache Connector on Other UNIX-Like Systems
  • Installing the Crowd Apache Connector on Windows
• 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
• Integrating Crowd with Atlassian HipChat
• Integrating Crowd with Atlassian FishEye - DRAFT
• 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
  • Viewing Users in Directories Mapped to an Application
  • Specifying which Groups can access an Application
• Effective memberships with multiple directories
• Specifying an Application's Address or Hostname
• Testing a User's Login to an Application
• Enforcing Lower-Case Usernames and Groups 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 priority order. The directory order is used for the following:

**Authentication**

Authentication only relies on the groups you mapped to the application. Users are authenticated if they to a group mapped to the application in the first directory where they exist, or if that directory is mapped to the application using the Allow all users to authenticate option.

**Authorization**

When multiple directories are mapped to an integrated application, and duplicate user names and group names are used across those directories, the effective group memberships for authorization are determined on the basis of the membership aggregation scheme that has been applied.

In particular, the 'non-aggregating' membership scheme depends on the directory order to determine access permissions for a user.
See Effective memberships with multiple directories for more information.

Directory updates

When a user is added to a group, they are only added to the first writeable directory available, in priority order. This applies for both the aggregating and non-aggregating membership schemes.

When a user is removed from a group, the behavior depends on the membership scheme:

- With non-aggregating membership, the user is only removed from the group in the first directory the user is found in.
- With aggregating membership, the user is removed from the group in all directories the user is found in.

See Directory update operations for an explanation of the membership aggregation schemes.

Specify the directory order

1. Log in to the Crowd Administration Console.
2. Click the Applications tab in the top navigation bar to display the Application Browser.
3. Click View for the application.
4. Click the Directories tab to 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:

Specify 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 and groups 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, because any user who could log into the Crowd application could change the application-level permissions for the Crowd application anyway.

For details on directory-level permissions, refer to the instructions on specifying directory permissions. Below are instructions on setting the application-level directory permissions.

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Group</td>
<td>Allows the application to add groups to the selected directory.</td>
</tr>
<tr>
<td>Add User</td>
<td>Allows the application to add users to the selected directory.</td>
</tr>
<tr>
<td>Modify Group</td>
<td>Allows the application to modify groups in the selected directory.</td>
</tr>
<tr>
<td>Modify User</td>
<td>Allows the application to modify users in the selected directory.</td>
</tr>
<tr>
<td>Remove Group</td>
<td>Allows the application to delete groups from the selected directory.</td>
</tr>
<tr>
<td>Remove User</td>
<td>Allows the application to delete users from the selected directory.</td>
</tr>
</tbody>
</table>

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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
    - Installing the Crowd Apache Connector on CentOS Linux
    - Installing the Crowd Apache Connector on Red Hat Enterprise Linux
    - Installing the Crowd Apache Connector on Ubuntu Linux
    - Installing the Crowd Apache Connector on Debian
    - Installing the Crowd Apache Connector on Other UNIX-Like Systems
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.

   ![View Directory - Customers](image)

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.

### 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**

**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 authorization 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: CW D-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

<table>
<thead>
<tr>
<th>Name</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arthur Dent</td>
<td>Username: adent</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:adent@example.com">adent@example.com</a></td>
</tr>
<tr>
<td></td>
<td>Alias: arthur</td>
</tr>
<tr>
<td>Ford Prefect</td>
<td>Username: ford</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:ford@example.com">ford@example.com</a></td>
</tr>
<tr>
<td>Marvin the Paranoid Android</td>
<td>Username: marvin</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:marvin@example.com">marvin@example.com</a></td>
</tr>
<tr>
<td>Slartibartfast Designer of Planets</td>
<td>Username: slartibartfast</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:slart@example.com">slart@example.com</a></td>
</tr>
<tr>
<td>Patricia MacMillan</td>
<td>Username: trillian</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:trillian@example.com">trillian@example.com</a></td>
</tr>
<tr>
<td>Zaphod Beeblebrox</td>
<td>Username: zaphod</td>
</tr>
<tr>
<td></td>
<td>Email: <a href="mailto:zaphod@example.com">zaphod@example.com</a></td>
</tr>
</tbody>
</table>

RELATED TOPICS

Specifying a User's Aliases
Managing Applications
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).
This setting will override any permissions configured in a client application. For example, even if the test-users group is given the Can Use permission in Confluence, if they aren’t a mapped group as specified on this page, they will be unable to authenticate. This does not prevent usernames and groups from appearing in the client application.

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’

<table>
<thead>
<tr>
<th>Directory</th>
<th>Group</th>
<th>Status</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crow</td>
<td>crowd-admins</td>
<td>Active</td>
<td>Remove</td>
</tr>
</tbody>
</table>

- Managing Users and Groups
- 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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Acegi Security
    - Integrating AppFuse - a Crowd-Acegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
Effective memberships with multiple directories

This page describes how Crowd determines group memberships for an integrated application that is mapped to multiple directories, where duplicate user names and group names are used across those directories.

If an application uses only one directory, then this page does not apply.

In Crowd 2.8, and later versions, there are two different schemes that Crowd can use with multiple directories. We have called these ‘aggregating membership’ and ‘non-aggregating membership’, and have described them below.

These schemes are only used to determine the group memberships that the integrated application uses for authorization purposes. Authentication is determined on the basis of the group mappings for an integrated application. See Specifying which Groups can access an Application.

When you map multiple directories to an integrated application (with duplicate user names), non-aggregating membership is applied by default. You can easily configure the application to use aggregating membership, as described below.

Non-aggregating membership

This is the ‘masking’ case – Crowd sees the directories mapped to an application from the top down, and will determine the effective group memberships based on the first instance of a user that is detected, working from the highest priority directory down to the lowest. Occurrences of the user in a lower priority directory are ‘masked’, and are not effective.

For example, Crowd will only consider Sam to be a member of the Admin group if that membership exists in the first directory in which Sam is found. If Sam is not a member of the Admin group in the first directory in which Sam is found, Crowd will not consider that membership even if Sam is a member of Admin in a lower directory.

Note that the priority order of the mapped directories is essential to this scheme. See Specifying the Directory
Order for an Application.

In the diagram of non-aggregating membership below, users or groups in a directory 'mask' themselves in lower priority directories.

For non-aggregating memberships:
A user effectively belongs to only those groups where the user is a group member in the highest directory that contains the user.

So, in the diagram:
- User A is only a member of Group A.
- User B is only a member of Group A.
- User C is only a member of Group B.

A group's members are all the 'top-most' users that belong to the group. If a group contains the user in a 'lower' directory, that membership is 'masked out' and is not effective.

So, in the diagram:
- Group A contains Users A and B.
- Group B only contains User C.

Aggregating membership

This is the 'blending' case - Crowd sees across all the directories mapped to an application, and will determine the effective group memberships based on a union of the directories.

For example, if Sam is a member of the Admin group in any directory, Crowd will consider Sam to be a member of the Admin group.

In the diagram of aggregating membership below, effective membership is a blend of the memberships in all mapped directories.

For aggregating memberships:
A user effectively belongs to all the groups of which the user is a member, in any directory.

So, in the diagram:
- User A is a member of Groups A and B.
- User B is a member of Groups A and B.
- User C is a member of Group B.

A group effectively contains all the users that belong to that group in any directory.

So, in the diagram:
- Group A contains Users A and B.
- Group B contains Users A, B, and C.
Configure the aggregation scheme for an application

When you map multiple directories to an integrated application in Crowd, non-aggregating membership is applied by default.

You can change the aggregation scheme for each integrated application by checking, or clearing, the **Aggregate group memberships**... checkbox on the **Directories** tab for the application:

![Aggregate group memberships](image)

The aggregation scheme applies across all the directories mapped to the application.

Directory update operations

When a user is added to a group, they are only added to the first writeable directory available, in priority order. This applies for both the aggregating and non-aggregating membership schemes.

When a user is removed from a group, the behavior depends on the membership scheme:

- With non-aggregating membership, the user is only removed from the group in the first directory the user is found in.
- With aggregating membership, the user is removed from the group in **all** directories the user is found in.

Inactive users

The membership schemes described above are not used when Crowd determines if a user should be able to authenticate.

Crowd only checks if the user is active in the first (highest priority) directory in which they are found when determining authentication.

For example, an application in Crowd is mapped to two directories: Crowd Internal Directory (primary) and an AD Delegated Authentication Directory (secondary).

- User A is inactive in the primary directory
- User A is active in secondary directory

Result: Crowd rejects access (authentication), because user A is first found in the primary directory, and the user is inactive there.

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.

<table>
<thead>
<tr>
<th>IP address and/or host name?</th>
</tr>
</thead>
<tbody>
<tr>
<td>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 authorization request unless Crowd recognizes the host name.</td>
</tr>
</tbody>
</table>

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.
   - Note: If an application running on the same server needs to access Crowd, you may need to add 'localhost' as well as '127.0.0.1' to the list of IP addresses and hostnames.
6. The new address will be added to the bottom of the list.

Screenshot: Application addresses

<table>
<thead>
<tr>
<th>Address</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.10.16</td>
<td>Remove</td>
</tr>
<tr>
<td>myhost.com</td>
<td>Remove</td>
</tr>
<tr>
<td>localhost</td>
<td>Remove</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>Remove</td>
</tr>
</tbody>
</table>

Notes

- **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 the application must be active. Ensure the 'Active' check box is ticked on the application 'Details' tab.

**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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
Integrating Crowd with Atlassian Bitbucket Server
Integrating Crowd with Acegi Security
  • Integrating AppFuse - a Crowd-Acegi Integration Tutorial
Integrating Crowd with Apache
  • Disabling Previous Versions of the Crowd Apache Connector
  • Installing the Crowd Apache Connector on CentOS Linux
  • Installing the Crowd Apache Connector on Red Hat Enterprise Linux
  • Installing the Crowd Apache Connector on Ubuntu Linux
  • Installing the Crowd Apache Connector on Debian
  • Installing the Crowd Apache Connector on Other UNIX-Like Systems
  • Installing the Crowd Apache Connector on Windows
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
Integrating Crowd with Atlassian HipChat
Integrating Crowd with Atlassian FishEye - DRAFT
  • 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
    • Viewing Users in Directories Mapped to an Application
    • Specifying which Groups can access an Application
      • Effective memberships with multiple directories
      • Specifying an Application's Address or Hostname
      • Testing a User's Login to an Application
      • Enforcing Lower-Case Usernames and Groups 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**

![Authentication Test Screen](image)

**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
Enforcing Lower-Case Usernames and Groups for an Application

In some cases you may wish to convert usernames and group 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
- Group 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 behavior 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 behavior 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 labeled 'Lower Case Output'.
6. Click the 'Update' button.

Screenshot: Application Options

RELATED TOPICS

- Case Sensitivity of Usernames and Groups
- Using the Application Browser
- Adding an Application
- Configuring the Google Apps Connector
- Mapping a Directory to an Application
- Effective memberships with multiple directories
- Specifying an Application's Address or Hostname
- Testing a User's Login to an Application
- Enforcing Lower-Case Usernames and Groups 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 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'**

<table>
<thead>
<tr>
<th>Username</th>
<th>Initialization</th>
<th>Last Accessed</th>
<th>Action</th>
</tr>
</thead>
</table>

**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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
Integrating Crowd with Atlassian JIRA 4.2 or earlier
Integrating Crowd with Atlassian Bitbucket Server
Integrating Crowd with Acegi Security
Integrating AppFuse - a Crowd-Acegi Integration Tutorial
Integrating Crowd with Apache
Disabling Previous Versions of the Crowd Apache Connector
Installing the Crowd Apache Connector on CentOS Linux
Installing the Crowd Apache Connector on Red Hat Enterprise Linux
Installing the Crowd Apache Connector on Ubuntu Linux
Installing the Crowd Apache Connector on Debian
Installing the Crowd Apache Connector on Other UNIX-Like Systems
Installing the Crowd Apache Connector on Windows
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
Integrating Crowd with Atlassian HipChat
Integrating Crowd with Atlassian FishEye - DRAFT
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
Viewing Users in Directories Mapped to an Application
Specifying which Groups can access an Application
Example of Directory Permissions
Viewing Users in Directories Mapped to an Application
Specifying which Groups can access an Application
Specifying an Application's Address or Hostname
Testing a User's Login to an Application
Enforcing Lower-Case Usernames and Groups 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 Confluence 3.4 or earlier
    - Updating Files in a Confluence Evaluation Distribution
  - Integrating Crowd with Atlassian CrowdID
  - Integrating Crowd with Atlassian Crucible
  - Integrating Crowd with Atlassian FishEye
    - Configuring FishEye earlier than 4.0 with Crowd
  - Integrating Crowd with Atlassian JIRA
    - Integrating Crowd with Atlassian JIRA 4.2 or earlier
  - Integrating Crowd with Atlassian Bitbucket Server
  - Integrating Crowd with Aegi Security
    - Integrating AppFuse - a Crowd-Aegi Integration Tutorial
  - Integrating Crowd with Apache
    - Disabling Previous Versions of the Crowd Apache Connector
    - Installing the Crowd Apache Connector on CentOS Linux
Crowd Documentation

Configuring Caching for an Application

Caching is used to store run-time authentication and authorization 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 \( \text{diskPersistent} = \text{true} \) this is the location on the file system where Ehcache will store its caching information. By default it uses \text{java.io.tmpdir} which is Java's default temporary file location.
  • **defaultCache**: This property has many configurable options. Please read the document ation 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 \( \text{isCacheEnabled} \). 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.

```xml
<ehcache>
  <diskStore path="java.io.tmpdir"/>
  <defaultCache maxElementsInMemory="4096" eternal="false" overflowToDisk="false"
    timeToIdleSeconds="300"
    timeToLiveSeconds="300"
    diskPersistent="false"
    diskExpiryThreadIntervalSeconds="120"/>
</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:

  \( \text{(Number of users x 2) + (number of groups x 2)} \)

Important Client Caches

The default \( \text{maxElementsInMemory} \) value of 4096 should be sufficient for most Crowd-integrated applications. However, for larger installations please ensure that the \( \text{maxElementsInMemory} \) matches the recommended
size calculation listed below:

<table>
<thead>
<tr>
<th>Name of Cache:</th>
<th>Size Calculation:</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.atlassian.crowd.integration-user</td>
<td>The number of users in your system.</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration-group</td>
<td>The number of groups in your system.</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration-parentgroup</td>
<td>The number of groups in your system.</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration-group-membership</td>
<td>The number of users multiplied by the number of groups.</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration-all-memberships</td>
<td>The number of users in your system.</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration-all-group-members</td>
<td>The number of groups in your system.</td>
</tr>
</tbody>
</table>

RELATED TOPICS

- Overview of Caching
- Configuring Caching for an LDAP Directory
- Authorization Caching
- Backing Up and Restoring Data
- Configuring Server Settings
  - Authorization Caching
  - Deployment Title
  - Domain
  - Finding Your Crowd Support Entitlement Number (SEN)
  - Licensing
  - Session Configuration
  - SSO Cookie
- Configuring the LDAP Connection Pool
- Configuring Trusted Proxy Servers
- Configuring your Mail Server
- Creating an Email Notification Template
- Draft - Troubleshooting and Requesting Technical Support
- 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 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:

<table>
<thead>
<tr>
<th>Application</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd</td>
<td>crowd.mydomain.com</td>
</tr>
<tr>
<td>JIRA</td>
<td>jira.mydomain.com</td>
</tr>
<tr>
<td>Confluence</td>
<td>confluence.mydomain.com</td>
</tr>
<tr>
<td>FishEye</td>
<td>fisheye.mydomain.com</td>
</tr>
<tr>
<td>FishEye in different domain</td>
<td>fisheye.example.com</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>Application</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd</td>
<td>mydomain.com/crowd</td>
</tr>
<tr>
<td>JIRA</td>
<td>mydomain.com/jira</td>
</tr>
<tr>
<td>Confluence</td>
<td>mydomain.com/confluence</td>
</tr>
<tr>
<td>FishEye</td>
<td>mydomain.com/fisheye</td>
</tr>
<tr>
<td>FishEye in different domain</td>
<td>example.com/fisheye</td>
</tr>
</tbody>
</table>

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 authorization 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 authorized 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.

**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:

<table>
<thead>
<tr>
<th>Security Framework or Custom Code</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework: Atlassian Seraph</td>
<td>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.</td>
</tr>
</tbody>
</table>
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.

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.

Crowd provides a custom integration with FishEye and/or Crucible, including SSO. See the Crowd documentation.

When integrating your own web application with Crowd, you can use the Crowd API to implement SSO.

- We recommend that you use the Crowd REST APIs for long-term compatibility.
- If you have a Java application, you can use the Java Integration 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 Marketplace.

Below are the configuration settings which affect SSO:

<table>
<thead>
<tr>
<th>Short Description</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set your SSO domain</td>
<td>Set the domain via the Crowd Administration Console, as described in the documentation.</td>
</tr>
<tr>
<td>Optional: Configure Trusted Proxy Servers</td>
<td>Configure Crowd to trust a proxy's IP address, if you are running applications behind one or more proxy servers. See the documentation.</td>
</tr>
<tr>
<td>Optional: Enforce a secure connection, such as SSL, for all SSO requests</td>
<td>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.</td>
</tr>
</tbody>
</table>

When integrating an application with Crowd, you will configure the application to use Crowd as a centralized 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 Atlassian Bitbucket Server
- Integrating Crowd with Acergi 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
Integrating Crowd with Atlassian HipChat
Integrating Crowd with Atlassian FishEye - DRAFT

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 application on the 'Options' tab. Click the links below for information about each option:

- Lower Case Output
- Enable Aliasing

Screenshot: Application Options

RELAT​ED TOPICS
Managing Applications
Managing Users and Groups

In Crowd, users are referred to as user entity objects or just users. 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 section describes how to add and edit users and groups using the Crowd Administration Console. Note that the ability to do this depends on the permissions of the directory which contains the users and groups.

Managing Users and Groups

- Using the User Browser
- Adding a User
- Editing a User's Details and Password
- Deleting or Deactivating a User
- Case Sensitivity of Usernames and Groups
- Specifying a User's Aliases
- Editing a User's Group Membership
- Managing Groups
  - Deleting a Group
  - Adding a Group
- Managing Group Members
  - Automatically Assigning 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
- Specifying a User's Attributes
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing a User's Session

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 organization.)
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'
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. 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.
   - **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 **Create** to add the user.

After creating the user, you will be able to specify the user's attributes and group membership. If you wish, you can also verify that the user can log in to appropriate applications.
Crowd Documentation

Editing a User’s Details and Password

Crowd administrators can edit a user’s details, rename users, mark a user as active or inactive, and change or reset a user’s password.

To edit a user’s details:

1. Log in to the Crowd Administration Console.
2. Click Users in the top navigation bar. This will display the User Browser.
3. Select the relevant directory, search for the user you want to update, and click the link on the user’s name.
4. Edit the details as required, then click Update.

To rename a user:

1. Log in to the Crowd Administration Console.
2. Click Users in the top navigation bar. This will display the User Browser.
3. Select the relevant directory, search for the user you want to update, and click the link on the user’s name.
4. Edit the Username field as required, then click Update.
Note: Not all directories support user rename. Crowd will inform you in the event where a user cannot be renamed.

To change a user’s password:

1. Log in to the Crowd Administration Console.
2. Click Users in the top navigation bar.
3. Select the relevant directory, search for the user you want to update, and click the link on the user’s name.
4. You can either:
   - Click Reset Password in the left-hand menu. Crowd will generate a random, unique URL and email it to the user. The user can then click the link and choose their own new password.
   - Enter a new password then click Update. Crowd will not email the user in this case.

View User – adent

<table>
<thead>
<tr>
<th>Details</th>
<th>Attributes</th>
<th>Groups</th>
<th>Roles</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username:</td>
<td>adent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Directory:</td>
<td>Atlassian Crowd — Crowd Internal Directory</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Email:</td>
<td><a href="mailto:adent@example.com">adent@example.com</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Active:</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Name:</td>
<td>Arthur</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Last Name:</td>
<td>Dnt</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Password:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Confirm Password:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes

- You will need to configure an email server so that Crowd can send the user an email notification when you reset their password.
- You can edit the email notification template to determine the content of the email sent to the user.
- Users can update their own profiles. Authorized 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

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 organization.

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.

For Microsoft Active Directory servers, LDAP Connector Directories and Delegated Authentication Directories will synchronize the status of users with the remote server, if the 'Synchronize User Details' option has been enabled. In other words, if a user account is disabled in Active Directory, it will be deactivated in Crowd on the next synchronization. Likewise, if a user is deactivated through Crowd, the user account will be disabled in Active Directory. If you want to prevent this synchronization, enable the 'Manage User Status Locally' option in the directory configuration.

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.

Deleting a User

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.
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

<table>
<thead>
<tr>
<th>Applications</th>
<th>Users</th>
<th>Groups</th>
<th>Roles</th>
<th>Directories</th>
<th>Administration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add User</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import Users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset Password</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remove User</td>
<td>Remove User – adent</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Are you sure you wish to remove this user?
Username: adent
Directory: Atlassian Crowd

Continue » Cancel

RELATED TOPICS

Managing Users and Groups

Crowd Documentation

Case Sensitivity of Usernames and Groups

This page summarizes the way Crowd handles case sensitivity for usernames and group 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 and group 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 and groups 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 and group names in mixed case.
- Support case-insensitive matching and searching.

Importing Users and Groups into Crowd Internal Directories
When you import user information into a Crowd Internal or Delegated Authentication directory, the case of usernames and group names will be preserved.

Enforcing Lower-Case Usernames and Groups for an Application

In some cases you may wish to convert user and group 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 and Groups 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

**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 authorized 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.

ℹ️ 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 labeled '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.
  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**

<table>
<thead>
<tr>
<th>Application</th>
<th>Alias</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>demo</td>
<td></td>
<td>Remove Alias</td>
</tr>
<tr>
<td>crowd-asend-server</td>
<td>Application aliasing disabled</td>
<td></td>
</tr>
<tr>
<td>confluence</td>
<td>arthur</td>
<td>Remove Alias</td>
</tr>
</tbody>
</table>

**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.
LOG IN just once, to any application.
Crowd's SSO will log you in to all others using the appropriate alias.

**RELATED TOPICS**
- Managing Users and Groups
- Crowd Documentation
- Editing a User's Group Membership

Within any given directory, you can choose the groups to which each user belongs.
When you add a user to a group, that user will be authorized to use any applications that use the group to control access.
Furthermore, that group membership may be used to determine authorization permissions – see Effective memberships with multiple directories.

**Groups**

The Crowd Administration Console provides two ways of adding users to, or removing them 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 a user to one or more groups at a time.
Full instructions are in Adding Users to a Group and Removing Users from a Group.

**RELATED TOPICS**

- Using the User Browser
- Adding a User
- Editing a User's Details and Password
-Deleting or Deactivating a User
- Case Sensitivity of Usernames and Groups
- Specifying a User's Aliases
- Editing a User's Group Membership
- Managing Groups
  - Deleting a Group
  - Adding a Group
- Managing Group Members
  - Automatically Assigning 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
- Specifying a User's Attributes
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing a User's Session

Crowd Documentation

**Managing Groups**

This page introduces you to groups in Crowd.

**About Groups**

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.

Support for roles, previously deprecated, has been removed in Crowd 2.5. The implementation of roles in Crowd was identical to the implementation of groups and did not provide any extra functionality.

**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**

The Group Browser allows you to search, view, add and edit the various groups 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

<table>
<thead>
<tr>
<th>Name</th>
<th>Active</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>crowd-administrators</td>
<td>true</td>
<td>View</td>
</tr>
<tr>
<td>my-team</td>
<td>true</td>
<td>View</td>
</tr>
<tr>
<td>team2</td>
<td>true</td>
<td>View</td>
</tr>
<tr>
<td>team3</td>
<td>true</td>
<td>View</td>
</tr>
</tbody>
</table>

Screenshot 2: Viewing and updating group details

View Group – my-team

<table>
<thead>
<tr>
<th>Details</th>
<th>Direct Members</th>
<th>Nested Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>my-team</td>
<td></td>
</tr>
<tr>
<td>Directory:</td>
<td>Atlassian Crowd — Crowd Internal Directory</td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td>My Team</td>
<td></td>
</tr>
<tr>
<td>Active: <strong>[ ]</strong></td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

Update >  Cancel

RELATED TOPICS
Managing Users and Groups
Crowd Documentation
Deleting a Group

Deleting a group removes it completely from the relevant directory.

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
Managing your user profile
Crowd Documentation
Adding a Group

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.

Adding a Group via the Administration Console

To add a group,

1. Log in to the Crowd Administration Console.
2. Click the ‘Groups’ link in the top navigation bar.
3. This will display the Group Browser. Click ‘Add Group’ 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. If your directory supports nested groups, you can now add sub-groups.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The unique name of the group. Within a given directory, the Name must be unique. Note that the Name cannot be changed once the group is created.</td>
</tr>
<tr>
<td>Description</td>
<td>A short description of the group.</td>
</tr>
<tr>
<td>Directory</td>
<td>The directory to which the group will be added. Note that the group cannot be moved to a different directory after it is created.</td>
</tr>
<tr>
<td>Active</td>
<td>Only deselect this if you wish to deny access to all members of the group.</td>
</tr>
</tbody>
</table>

Screenshot 1: ‘Group Browser’

<table>
<thead>
<tr>
<th>Name</th>
<th>Active</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>True</td>
<td>View</td>
</tr>
<tr>
<td>crowd-administrators</td>
<td>True</td>
<td>View</td>
</tr>
</tbody>
</table>

Screenshot 2: ‘Add Group’
Importing Groups from Other Applications

You can also add groups via Crowd's migration tools. See Importing Users and Groups into a Directory.

Group Authorization

See Specifying which Groups can access an Application.

Roles have been Removed

Support for roles, previously deprecated, has been removed in Crowd 2.5. The implementation of roles in Crowd was identical to the implementation of groups and did not provide any extra functionality.

**RELATED TOPICS**

- Using the User Browser
- Adding a User
- Editing a User's Details and Password
- Deleting or Deactivating a User
- Case Sensitivity of Usernames and Groups
- Specifying a User's Aliases
- Editing a User's Group Membership
- Managing Groups
  - Deleting a Group
  - Adding a Group
- Managing Group Members
  - Automatically Assigning 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
- Specifying a User's Attributes
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing a User's Session

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**

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Active</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>crowd-administrators</td>
<td>true</td>
<td>View</td>
</tr>
<tr>
<td>my-team</td>
<td>true</td>
<td>View</td>
</tr>
<tr>
<td>team2</td>
<td>true</td>
<td>View</td>
</tr>
<tr>
<td>team3</td>
<td>true</td>
<td>View</td>
</tr>
</tbody>
</table>
```

**Screenshot 2: Viewing the direct members of 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 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
Crowd Documentation
Automatically Assigning 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 user that logs in becomes a member of these groups automatically.

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, users will not be added automatically into the group when they log in. Existing users will remain members of the group.

Screenshot: Default groups for a directory
ADDING USERS TO A GROUP

When you add a user to a group, that user will be authorized to use any applications that use this group to control access.

Furthermore, that group membership may be used to determine authorization permissions – see Effective memberships with multiple directories.

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

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 organization.)
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

<table>
<thead>
<tr>
<th>View Group – team2</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Details] [Direct Members] [Nested Members]</td>
</tr>
</tbody>
</table>

Groups in this Group

There are no group members in the "team2" group.

Add Groups

Users in this Group

<table>
<thead>
<tr>
<th>Username</th>
<th>Email</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>jay</td>
<td><a href="mailto:jay@example.com">jay@example.com</a></td>
<td>true</td>
</tr>
</tbody>
</table>

Add Users [Remove Users]

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 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.
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
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 organization.)
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

<table>
<thead>
<tr>
<th>View Group – my-team</th>
</tr>
</thead>
<tbody>
<tr>
<td>Details</td>
</tr>
</tbody>
</table>

Groups in this Group

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>team2</td>
<td>Team 2</td>
<td>true</td>
</tr>
</tbody>
</table>

Users in this Group

<table>
<thead>
<tr>
<th>Username</th>
<th>Email</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td><a href="mailto:admin@example.com">admin@example.com</a></td>
<td>true</td>
</tr>
<tr>
<td>admin</td>
<td><a href="mailto:smadcox@atlassian.com">smadcox@atlassian.com</a></td>
<td>true</td>
</tr>
<tr>
<td>trillion</td>
<td><a href="mailto:trillion@example.com">trillion@example.com</a></td>
<td>true</td>
</tr>
</tbody>
</table>

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
This page describes the way Crowd handles nested groups, i.e. groups that contain other groups or that belong to 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 authorization 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 attributes, where the DN can be that of a user or that of a group elsewhere in the LDAP tree:

```plaintext
member=CN=John Smith,OU=Users,OU=OrgUnitA,DC=sub,DC=domain
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 customization 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, Bitbucket Server, 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
Adding a Group
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. First, enable nested groups via the directory configuration screen, the 'Add Groups' button will not appear if nested groups are not enabled for your directory.
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, 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**
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,

**Screenshot: Popup for adding sub-groups**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crowd-administrator</td>
<td></td>
</tr>
<tr>
<td>team3</td>
<td>Team 3</td>
</tr>
</tbody>
</table>
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.
RELATED TOPICS
Nested Groups in Crowd
Adding a Sub-Group
Managing Groups
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, even though they are stored locally on the Crowd server.

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.
  - Enter the name of the new attribute (e.g. phone) in the 'Attribute' field at the bottom of the screen.
  - Enter the value of the new attribute (e.g. 0123456789) in the 'Value' field at the bottom of the screen.
  - 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.

> 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.
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**
**View User – adent**

These are the groups the user is a member of:

<table>
<thead>
<tr>
<th>Group</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>confluence-users</td>
<td>Remove</td>
</tr>
<tr>
<td>my-team</td>
<td>Remove</td>
</tr>
</tbody>
</table>

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
- Editing a User's Details and Password
- Deleting or Deactivating a User
- Case Sensitivity of Usernames and Groups
- Specifying a User's Aliases
- Editing a User's Group Membership
- Managing Groups
- Managing Group Members
- Specifying a User's Attributes
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing a User's Session

**Crowd Documentation**

**Granting Crowd User Rights to a User**

This page tells you how to authorize users to access Crowd, without giving them Crowd administration rights. Only Crowd administrators can authorize other users to access Crowd.

**Administrators and Non-Administrators**

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

Authorized 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 authorization details. But they will not be able to view or update other user profiles, nor perform any Crowd administration functions.

**Authorizing Non-Administrators to Use the Crowd Self-Service Console**
To authorize 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 authorized 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 a User's Session**

**Number of Sessions**

For Crowd 2.0.4 and newer versions, a single session is allowed for each user in a machine accessing an application integrated to Crowd. So, for instance, if you are accessing JIRA and then open a new Browser model and try to login to the same application, two sessions will be created in the issue tracker, however a single session will be created in Crowd. If one of the sessions is terminated in JIRA, all the sessions will be terminated.

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'*

<table>
<thead>
<tr>
<th>Username</th>
<th>Directory</th>
<th>Initialization</th>
<th>Last Accessed</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>Atlassian Internal</td>
<td>2/20/2008 09:32:25</td>
<td>2/20/2008 10:10:08</td>
<td>View</td>
</tr>
</tbody>
</table>

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
- Editing a User's Details and Password
- Deleting or Deactivating a User
- Case Sensitivity of Usernames and Groups
- Specifying a User's Aliases
- Editing a User's Group Membership
- Managing Groups
- Managing Group Members
- Specifying a User's Attributes
- Granting Crowd Administration Rights to a User
- Granting Crowd User Rights to a User
- Managing a User's Session

**Crowd Documentation**

**System Administration**
- Configuring Server Settings
  - Deployment Title
  - Domain
  - Session Configuration
  - Authorization Caching
• Licensing
• Finding Your Crowd Support Entitlement Number (SEN)
• 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
• Draft - Troubleshooting and Requesting Technical Support
• Configuring the LDAP Connection Pool
• Overview of Caching

Other Related Security Resources

• Security Advisory Publishing Policy
• Severity Levels for Security Issues
• How to Report a Security Issue
• Security Patch Policy
• Crowd Security Advisories and Fixes
• Crowd Security Notice 2013-07-01
• Crowd Administration Guide
• System Administration

Configuring Server Settings

You can alter the settings which were specified when your Crowd server was installed:

• Deployment Title
• Domain
• Session Configuration
• Authorization Caching
• Licensing
• Finding Your Crowd Support Entitlement Number (SEN)
• SSO Cookie

RELATED TOPICS

• Configuring Server Settings
  • Deployment Title
  • Domain
  • Session Configuration
  • Authorization Caching
  • Licensing
  • Finding Your Crowd Support Entitlement Number (SEN)
  • 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
• Draft - Troubleshooting and Requesting Technical Support
• Configuring the LDAP Connection Pool
• 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'

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment Title</td>
<td>The name of this Crowd instance.</td>
</tr>
<tr>
<td>SSO Domain</td>
<td>The SSO domain for this Crowd deployment. For example, somecorp.com. If you want to support single sign-on across multiple hosts, ensure the period (.) in front of the domain. Leave this field empty if you want cookies to be set to the domain that requests are made to.</td>
</tr>
<tr>
<td>Secure SSO Cookie</td>
<td>If checked, the &quot;Secure&quot; flag is set on the cookie. This will break SSO for applications not accessed over SSL/TLS (https://), potentially making logging into Crowd impossible.</td>
</tr>
<tr>
<td>Enable Authorisation Caching</td>
<td>If checked, Crowd will cache a user’s authentication and per-application permissions for a specified period. Recommended setting: Enabled, for vastly better performance. Disable only if you need immediate results when removing users or their permissions.</td>
</tr>
<tr>
<td>GZip Compression</td>
<td>Tick the box to enable Gzip compression of responses from the Crowd Security Server.</td>
</tr>
<tr>
<td>Token Seed</td>
<td>A key used to generate authentication tokens in your Crowd deployment. The tokens are used when authenticating applications and users.</td>
</tr>
</tbody>
</table>

RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- Overview of Caching
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.

On this page:
- Overview
- Setting the SSO Domain
- Setting the SSO Domain when Crowd is behind a Proxy Server
- Notes

### Overview

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:

<table>
<thead>
<tr>
<th>Application</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd</td>
<td>crowd.mydomain.com</td>
</tr>
<tr>
<td>JIRA</td>
<td>jira.mydomain.com</td>
</tr>
<tr>
<td>Confluence</td>
<td>confluence.mydomain.com</td>
</tr>
<tr>
<td>FishEye</td>
<td>fishey.mydomain.com</td>
</tr>
<tr>
<td>FishEye in different domain</td>
<td>fisheye.example.com</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>Application</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd</td>
<td>mydomain.com/crowd</td>
</tr>
<tr>
<td>JIRA</td>
<td>mydomain.com/jira</td>
</tr>
<tr>
<td>Confluence</td>
<td>mydomain.com/confluence</td>
</tr>
<tr>
<td>FishEye</td>
<td>mydomain.com/fishey</td>
</tr>
<tr>
<td>FishEye in different domain</td>
<td>example.com/fisheye</td>
</tr>
</tbody>
</table>

You can find information the comparison of host name strings in RFC 2965 (pages 2 and 3).

**i** When developing on your local machine, you should set the domain to localhost.

### Setting the SSO Domain

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

```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" />
```

Notes

- **Avoiding problems with old cookie versions.** In order to avoid problems with hosts or domains defined in old cookie versions, after setting the SSO Domain in Crowd, log out of Crowd and the integrated applications and delete all the web browser cookies.

- **SSO domain.** The 'SSO Domain' field will accept only values based on the domain that is used to access the Crowd console. For instance, if you are using 'www.mycrowd.com/crowd/console' to access the console in the web browser, this field will accept the following values:
  - Empty
  - mycrowd.com
  - .mycrowd.com

  If you enter any other value, Crowd will show an error message: *The supplied domain is invalid.*

- **IP addresses.** SSO will not operate when sites are accessed using IP addresses rather than domain names. This is a limitation of the cookie technology implemented in web browsers.
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 on Timeout' field, then click the 'Update' button.

**Require Consistent Client IP Address**

(Available since Crowd 2.5.2.)

Authenticated sessions can be tied to the IP address they were created from. This means that an attempt to use that session from another machine will fail, which will force mobile clients to reauthenticate when their IP address changes.

This setting can be disabled to relax that requirement, so a session can be used from any IP address. Note that changing this setting will invalidate any existing sessions, so **you will be logged out after making this change**.

To allow sessions to be used from any IP address,

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. Check or uncheck 'Require Consistent Client IP Address' as required, 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:

<table>
<thead>
<tr>
<th>Number of Tokens:</th>
<th>100</th>
<th>500</th>
<th>1000</th>
<th>5000</th>
<th>10000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database -&gt; Memory</td>
<td>0.1s</td>
<td>0.7s</td>
<td>1.2s</td>
<td>4.2s</td>
<td>8.2s</td>
</tr>
<tr>
<td>Memory -&gt; Database</td>
<td>1.2s</td>
<td>4.8s</td>
<td>9.2s</td>
<td>45s</td>
<td>90s</td>
</tr>
</tbody>
</table>

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'

**In-memory cache size**

The size of the in-memory token cache is defined in the `crowd-webapp/WEB-INF/classes/crowd-ehcache.xml` file. The default should be acceptable for most cases. If you require more than 2048 concurrent sessions in memory you may increase the size of the '-hash-cache' caches.

**RELATED TOPICS**

- Managing an Application's Session
- Managing a User's Session
- Configuring Server Settings
  - Deployment Title
  - Domain
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- Overview of Caching

Crowd Documentation

Authorization Caching
Caching is used to store run-time authentication and authorization 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 Authorization 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 ‘authorization 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.
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 (isCach
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
- Minimizing 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.)

It is important to avoid exceeding the license limit, because once the user limit has been exceeded, no one can log in (including administrators).

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 console exceeds the user license limit, no one will be able to log in to any applications. When you try to access Crowd, you’ll be redirected to a page where you can update your license key. On this page, you can either:

- Enter a new license key for a higher user count. (Purchase one here.)
- Enter a 30-day evaluation license key. (Create one here.) The 30-day evaluation key lets an unlimited number of users sign in during the trial period. You can use the 30 days to clean up the users in the system or purchase a license key for a higher user count.

Minimizing 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 labeled '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
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- Overview of Caching

Finding Your Crowd Support Entitlement Number (SEN)

There are three ways to find your Support Entitlement Number (SEN).

Method 1: View the license details in the Crowd Administration Console

Select 'Administration', 'License Details'. The SEN is shown:
Method 2: Log in to my.atlassian.com as the account holder or technical contact

Your Support Entitlement Number is available at http://my.atlassian.com:

Method 3: Look at your Atlassian Invoice

Your Support Entitlement Number (SEN) appears on the third page of your Atlassian invoice.

More information

See Finding Your Support Entitlement Number in the support space for more general information about how Atlassian Support uses this number.

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
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- Overview of Caching
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 requests a password reset or a server event occurs.

On this page:
- Accessing the Mail Configuration Screen
- Mail Server Option 1: SMTP
- Mail Server Option 2: JNDI Location

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 message 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.
- **Use Secure Sockets Layer (SSL)** — Select this check-box if you want to access your mail server over SSL (Secure Sockets Layer). This ensures that all email communications between Crowd and your mail server are encrypted, provided your mail server supports SSL.

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.

**Mail Server Option 2: JNDI Location**

<table>
<thead>
<tr>
<th>Mail Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Notification Email Address</strong>: <a href="mailto:john@example.com">john@example.com</a></td>
</tr>
<tr>
<td><strong>From Email Address</strong>: <a href="mailto:john@example.com">john@example.com</a></td>
</tr>
<tr>
<td><strong>Subject Prefix</strong>: [Crowd - Atlassian Crowd]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mail Server Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>JNDI Location</strong></td>
</tr>
<tr>
<td><strong>JNDI Location</strong>: <code>java:comp/env/mail/CrowdMailServer</code></td>
</tr>
</tbody>
</table>

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 the Crowd distribution (not EAR-WAR)), 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>`
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
```

**Notes**

- To customize the password notification message, see the page about email notification templates.

**RELATED TOPICS**

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- Overview of Caching

Crowd Documentation
Creating an Email Notification Template

Crowd uses an email template to build the content of an email message that Crowd sends to a user. Crowd provides the following email templates:

- **Password Resets**: A template for the email sent when an administrator asks a user to reset their password and when a user asks to reset their own forgotten password.
- ** Forgotten usernames**: A template for the email sent when a user requests their forgotten username.

Email Template for Password Resets (Forgotten Passwords)

This is a template for the email sent when an administrator asks a user to reset their password and when a user asks to reset their own forgotten password.

To edit the email template for password resets,

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. In the 'Forgotten Password Template' text box, 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:
   - **$username** – The username of the person who will receive the email.
   - **$firstname** – The user's first name.
   - **$lastname** – The user's last name.
   - **$deploymenttitle** – The title of your Crowd site, as defined in Deployment Title.
   - **$date** – The date/time of the message event.
   - **$resetlink** – The automatically-generated URL that the user can click, allowing them to choose a new password.

   *Note:* To ensure that the password reset URL will include the correct domain for your Crowd server, please set the `crowd.base.url` variable to the correct domain value in the `<Crowd-Home-Directory>/crowd.properties` file:

```
crowd.base.url=http://your_domain.com
```

5. Click 'Update'.

* Earlier releases of Crowd supplied the `$password` macro to represent the user's new password, automatically generated by Crowd. Crowd no longer generates a new password, but instead generates a link that the user can click to choose their own new password. For backwards compatibility, if your email template contains the `$password` macro, Crowd will now replace it with the text 'available at (link)'. The '(link)' will be the same as now available in the `$resetlink` macro.

Email Template for Forgotten Usernames

This is a template for the email sent when a user requests their forgotten username.

To edit the email template for forgotten usernames,
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. In the 'Forgotten Username(s) Template' text box, 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.
   - $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.
   - $email – The email address that the user entered when requesting forgotten usernames. This is the address to which the email message is sent.
   - $admincontact – The email address of the Crowd administrator.
5. Click 'Update'.

Screenshot: Mail Templates

**Mail Template**

**Forgotten Password Template:**

```
Hello $firstname $lastname,

You (or someone else) have requested to reset your password for $deploymenttitle on $date.

If you follow the link below you will be able to personally reset your password.
$resetlink

This password reset request is valid for the next 24 hours.
```

**Forgotten Username(s) Template:**

```
Hello $firstname $lastname,

You have requested the username for your email: $email.

Your username is: $username

If you think it was sent incorrectly please contact one of the administrators at $admincontact.

$deploymenttitle Administrator

```

Related Topics:

- Requesting Forgotten Usernames
- Resetting Forgotten Passwords
  - Configuring Server Settings
Configuring Trusted Proxy Servers

If you are running applications behind one or more proxy servers then you may find it useful to configure Crowd to trust the proxies' addresses. When a proxy server forwards an HTTP request, Crowd will recognize the request as coming from the request's originator, not from 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 iterate through client IP address and IP addresses in the X-Forwarded-For header from right to left and pick the first IP address that is not a trusted proxy. The address is then used as 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 or the host name of the proxy server. Possible values are:
   - A full IP address, e.g. 192.168.10.12 (IPv4) or 2001:db8:85a3:0:0:8a2e:370:7334 (IPv6).
   - An IPv4 subnet using wildcard notation, e.g. 192.168.*.*.
   - An IPv4 or IPv6 subnet, using CIDR notation, e.g. 192.168.10.1/16 (IPv4) or 2001:db8:85a3::/64 (IPv6). For more information, see the introduction to CIDR notation on Wikipedia and RFC 4632.
   - A host name, e.g. proxy.example.org. All IP addresses bound to the given host name will be trusted.
   
   Using host names will cause DNS requests to be sent, which might affect Crowd performance.

5. Click the 'Add' button.

Screenshot: Trusted Proxy Servers
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

<table>
<thead>
<tr>
<th>Date</th>
<th>Wednesday, 20 Feb 2007</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>13:29:54</td>
</tr>
<tr>
<td>Timezone</td>
<td>Eastern Standard Time (New South Wales)</td>
</tr>
<tr>
<td>Java Version</td>
<td>1.6.0_04</td>
</tr>
<tr>
<td>Java Vendor</td>
<td>Sun Microsystems Inc.</td>
</tr>
<tr>
<td>JVM Version</td>
<td>10.0-b19</td>
</tr>
<tr>
<td>JVM Vendor</td>
<td>Sun Microsystems Inc.</td>
</tr>
<tr>
<td>JVM Runtime</td>
<td>Java HotSpot(TM) Client VM</td>
</tr>
<tr>
<td>Username</td>
<td>smaddox</td>
</tr>
<tr>
<td>Operating System</td>
<td>Windows XP 5.1</td>
</tr>
<tr>
<td>Architecture</td>
<td>x86</td>
</tr>
</tbody>
</table>

## Crowd Information

Home Directory: `C:\data\crowd-home-beta2`

## JVM Statistics

- Total Memory: 47 MB
- Used Memory: 28 MB
- Free Memory: 20 MB

## Database Information

- JDBC URL: `jdbc:hsql:db:/data/crowd-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: 1.0-APSHOT
- Build Number: 212
- Build Date: Nov 30, 2007

## License Information

License Server ID: AGZS:AGZS:AGZS:AGZS

### RELATED TOPICS

- Configuring Server Settings
- Deployment Title
You can back up your Crowd data by exporting it to an XML file in the Crowd Home directory. 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

We recommend that you back up your data regularly, especially before any significant configuration changes and before upgrading Crowd. You should also perform regular backups of your database and your Crowd Home directory.

#### To schedule daily backups of 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. In the 'Schedule Backup' panel, select the 'Enable scheduled backups to XML' and choose the time.
5. Click the 'Update' button.

When scheduled backups are enabled, Crowd will create a daily backup in the /backups directory under your Crowd Home directory. Backup files will be rotated to retain the most recent fifty daily backups.

#### To manually 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. In the 'Manual Backup' panel, 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. Enter an appropriate 'Backup File Name'. This will be the name of the XML file that Crowd will create. When the backup process has finished, you will find the backup file in the /backups directory under your Crowd Home directory.
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.

### RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- Overview of Caching

### 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](example.com).

### 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:

<table>
<thead>
<tr>
<th>Level</th>
<th>Type of Message Written to the Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEBUG</td>
<td>Used to troubleshoot SSO problems only. These are low-level details that most people never need to know about. NOTE: This setting may cause user passwords to be logged.</td>
</tr>
<tr>
<td>INFO</td>
<td>Informational messages about what Crowd is doing. Usually not interesting.</td>
</tr>
<tr>
<td>WARN</td>
<td>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.</td>
</tr>
<tr>
<td>ERROR</td>
<td>Indications that something has gone wrong in Crowd. The person responsible for configuring Crowd should be notified.</td>
</tr>
<tr>
<td>FATAL</td>
<td>Indications that something has gone wrong so badly that the system cannot recover.</td>
</tr>
<tr>
<td>ALL</td>
<td>All possible log messages.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Installation Type</th>
<th>Location of Log File</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd Standalone edition</td>
<td><strong>Crowd 2.0.3 and older versions</strong>: In the root directory of your Crowd application, e.g. <code>atlassian-crowd-2.0.0/atlassian-crowd.log</code></td>
</tr>
<tr>
<td></td>
<td><strong>Crowd 2.0.4 and newer versions</strong>: In the Crowd application Home Directory, e.g. <code>Crowd-Home-Directory/logs/atlassian-crowd.log</code></td>
</tr>
<tr>
<td>Crowd Standalone running as a Windows service</td>
<td><code>C:\Windows\system32\atlassian-crowd.log</code></td>
</tr>
<tr>
<td>Crowd WAR edition</td>
<td>The directory from which you start the application server, e.g. <code>apache-tomcat-6.0.16/bin/atlassian-crowd.log</code></td>
</tr>
</tbody>
</table>

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 behavior 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**

<table>
<thead>
<tr>
<th>Class/Package Name</th>
<th>Current Level</th>
<th>New Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.atlassian.crowd</td>
<td>INFO</td>
<td>INFO</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration.service.soap.fire.XFireFaultLoggingMethodHandler</td>
<td>WARN</td>
<td>WARN</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration.service.soap.fire.XFireInLoggingMethodHandler</td>
<td>WARN</td>
<td>WARN</td>
</tr>
<tr>
<td>com.atlassian.crowd.integration.service.soap.fire.XFireOutLoggingMethodHandler</td>
<td>WARN</td>
<td>WARN</td>
</tr>
<tr>
<td>com.atlassian.crowd.license</td>
<td>ERROR</td>
<td>ERROR</td>
</tr>
<tr>
<td>com.atlassian.crowd.startup</td>
<td>INFO</td>
<td>INFO</td>
</tr>
<tr>
<td>root</td>
<td>WARN</td>
<td>WARN</td>
</tr>
</tbody>
</table>
### Description of the loggers:

<table>
<thead>
<tr>
<th>Logger</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.atlassian.crowd</td>
<td>This is the parent of the <code>crowd</code> package loggers. Any children which do not have a level assigned to them will inherit the level from their parent. This logger should be set to <code>DEBUG</code> only if you are investigating SSO issues.</td>
</tr>
<tr>
<td>com.atlassian.crowd....XFireFaultLoggingMethodHandler</td>
<td>Can be helpful if a Crowd SOAP service fault is thrown. It is best to enable <code>DEBUG</code> for all three XFire classes simultaneously when troubleshooting Crowd's SOAP service.</td>
</tr>
<tr>
<td>com.atlassian.crowd....XFireOutLoggingMethodHandler</td>
<td>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.</td>
</tr>
<tr>
<td>com.atlassian.crowd....XFireInLoggingMethodHandler</td>
<td>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.</td>
</tr>
<tr>
<td>com.atlassian.crowd.license</td>
<td>Useful for troubleshooting certain licensing issues in Crowd.</td>
</tr>
<tr>
<td>com.atlassian.crowd.startup</td>
<td>Can be helpful for troubleshooting startup errors in Crowd.</td>
</tr>
<tr>
<td>root</td>
<td>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.</td>
</tr>
</tbody>
</table>

### 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 behavior is defined in the following properties file:

- **For Crowd installations:** `{CROWD-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 **Crowd installations of CrowdID**: `{CROWDID-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
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
- Draft - Troubleshooting and Requesting Technical Support
- Configuring the LDAP Connection Pool
- 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:

```
[15ms] - AOP: SecurityServer.findPrincipalByToken()
[15ms] - AOP: SecurityServer.isValidPrincipalToken()
[15ms] - AOP: SecurityServer.isValidPrincipalToken()
[15ms] - AOP: SOAPService.validateSOAPService()
[16ms] - AOP: SecurityServer.getDomain()
[16ms] - AOP: SOAPService.validateSOAPService()
```

RELATED TOPICS
Logging and Profiling
Configuring the LDAP Connection Pool

When connection pooling is enabled, the LDAP service provider maintains a pool of connections and assigns them as needed. When a connection is closed, LDAP returns the connection to the pool for future use. This can improve performance significantly.

This page describes the site-wide settings for LDAP connection pooling in Crowd.

To configure the LDAP connection pooling in Crowd,

1. Log in to the Crowd Administration Console.
2. Click the 'Administration' tab in the top navigation bar.
3. Click 'LDAP Connection Pool' in the left-hand menu.
4. The 'LDAP Connection Pool' screen appears. Enter the details for each setting, as described in the table below.
5. Click the 'Update' button.
6. Restart Crowd to put the changes into effect.

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
<table>
<thead>
<tr>
<th>Connection Pool Setting</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Pool Size</td>
<td>The number of LDAP connections created when initially connecting to the pool.</td>
<td>1</td>
</tr>
<tr>
<td>Preferred Pool Size</td>
<td>The optimal pool size. LDAP will remove idle connections when the number of connections grows larger than this value. A value of 0 (zero) means that there is no preferred size, so the number of idle connections is unlimited.</td>
<td>10</td>
</tr>
<tr>
<td>Maximum Pool Size</td>
<td>The maximum number of connections. When the number of connections reaches this value, LDAP will refuse further connections. As a result, requests made by an application to the LDAP server will be blocked. A value of 0 (zero) means that the number of connections is unlimited.</td>
<td>0</td>
</tr>
<tr>
<td>Pool Timeout</td>
<td>The length of time, in seconds, that a connection may remain idle before being removed from the pool. When the application is finished with a pooled connection, the connection is marked as idle, waiting to be reused. A value of 0 (zero) means that the idle time is unlimited, so connections will never be timed out.</td>
<td>30</td>
</tr>
<tr>
<td>Pool Protocol</td>
<td>Only these protocol types are allowed to connect to LDAP. If you want to allow multiple protocols, enter the values separated by a space. Valid values are: plain ssl (Both plain and ssl)</td>
<td>plain ssl</td>
</tr>
<tr>
<td>Pool Authentication</td>
<td>Only these authentication types are allowed to connect to LDAP. If you want to allow multiple authentication types, enter the values separated by a space. See RFC 2829 for details of LDAP authentication methods. Valid values are: none simple DIGEST-MD5</td>
<td>simple</td>
</tr>
</tbody>
</table>

Screenshot: LDAP Connection Pool
# LDAP Connection Pool

You can configure the settings used for pooling of LDAP server connections below. These settings are system wide and will be used to create a new connection pool for each configured LDAP server.

## Current Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Pool Size</td>
<td>1</td>
</tr>
<tr>
<td>Preferred Pool Size</td>
<td>10</td>
</tr>
<tr>
<td>Maximum Pool Size</td>
<td>0</td>
</tr>
<tr>
<td>Pool Timeout (seconds)</td>
<td>30</td>
</tr>
<tr>
<td>Pool Protocol</td>
<td>plain ss1</td>
</tr>
<tr>
<td>Pool Authentication</td>
<td>simple</td>
</tr>
</tbody>
</table>

## Update Settings

⚠️ Changes to these settings will not be active until the server has been restarted.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Pool Size</td>
<td></td>
</tr>
<tr>
<td>Preferred Pool Size</td>
<td>10</td>
</tr>
<tr>
<td>Maximum Pool Size</td>
<td>0</td>
</tr>
<tr>
<td>Pool Timeout (seconds)</td>
<td>30</td>
</tr>
<tr>
<td>Pool Protocol</td>
<td>plain ss1</td>
</tr>
<tr>
<td>Pool Authentication</td>
<td>simple</td>
</tr>
</tbody>
</table>

## RELATED TOPICS

- Configuring Server Settings
  - Deployment Title
  - Domain
  - Session Configuration
  - Authorization Caching
  - Licensing
  - Finding Your Crowd Support Entitlement Number (SEN)
  - 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
Caching is used to store run-time authentication and authorization 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 authorization 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 [Authorization Caching](#).

- **LDAP directory caches in the Crowd database** – The Crowd database keeps an up-to-date cache of all user and group information from the LDAP directory. 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
- Authorization Caching
- Configuring Caching for an Application

Crowd Documentation

Crowd Installation and Upgrade Guide

- Crowd Release Notes
- Installing Crowd
- Upgrading Crowd
- Migrate to Another Database
• Migrating Crowd Between Servers
• Migrating from OnDemand to a Crowd installed site

Crowd Release Notes

Crowd 2.11.0 has now been released — see the Crowd 2.11.0 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 2.11.0 Release Notes
• Crowd 2.10.1 Release Notes
• Crowd 2.9.5 Release Notes
• Crowd 2.9.1 Release Notes
• Crowd 2.8.8 Release Notes
• Crowd 2.8.4 Release Notes
• Crowd 2.8.3 Release Notes
• Crowd 2.8.2 Release Notes
• Crowd 2.8 Release Notes
• Crowd 2.7.2 Release Notes
• Crowd 2.7.1 Release Notes
• Crowd 2.7 Release Notes
• Crowd 2.6.5 Release Notes
• Crowd 2.6.4 Release Notes
• Crowd 2.6.3 Release Notes
• Crowd 2.6.2 Release Notes
• Crowd 2.6.1 Release Notes
• Crowd 2.6 Release Notes
• Crowd 2.5.5 Release Notes
• Crowd 2.5.4 Release Notes
• Crowd 2.5.3 Release Notes
• Crowd 2.5.2 Release Notes
• Older releases

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 2.10 — 6 September 2016

• Major performance improvements
• Improved group synchronization behavior
• Improved UI for LDAP configuration testing
• Many bug fixes and other improvements - see the Crowd 2.10.1 Release Notes

Crowd 2.9 — 10 May 2016

• Oracle 12c is now supported
• Additional functions have been added to the REST API

Crowd 2.8 — 10 Nov 2014

• Full UI makeover
• Option to merge group memberships from multiple directories
• New experimental lightweight OpenID server
• Upgrades to Spring LDAP 2.0, Apache HttpClient 4.3
• PostgreSQL 9.3 is now supported

Crowd 2.7 — 23 Sept 2013
• User rename
• Scheduled backups
• Improved Microsoft Active Directory integration
• Simpler setup
• More in the Crowd 2.7 Release Notes

Crowd 2.6 — 11 Feb 2013
• Faster LDAP & Active Directory synchronization
• Local groups management for LDAP connectors
• More in the Crowd 2.6 Release Notes

Crowd 2.5 — 6 Aug 2012
• JDK 7 support
• Crowd console performance improved with large numbers of groups
• Database drivers are now bundled for PostgreSQL, Microsoft SQL Server and Oracle
• More in the Crowd 2.5 Release Notes

Crowd 2.4 — 11 Jan 2012
• Remote Crowd directories are synchronized incrementally when possible
• The previously selected directory is selected by default on new pages
• More in the Crowd 2.4 Release Notes

Crowd 2.3 — 20 July 2011
• Host name support for trusted proxies
• More in the Crowd 2.3.1 release notes

Crowd 2.2 — 10 March 2011
• Crowd Query Language
• Simple, Quick Configuration of JIRA and Confluence
• Connection to Other Crowd Servers
• Improvements to the LDAP Connector
• More in the Crowd 2.2.2 release notes

Crowd 2.1 — 1 December 2010
• REST API
• Improved Apache and Subversion Connectors
• Database-Backed Caching for All LDAP Directories
• LDAP Connection Pooling
• Secure Password Resets
• More in the Crowd 2.1 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 Authorization 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 2.11.0 Release Notes
12 January 2017
The Atlassian Crowd team is pleased to bring you Crowd 2.11.

This release brings significant performance improvements, as well as many bug fixes.

We recommend you read the upgrade notes for important information about this release.

Thank you for your feedback:

- 80 votes satisfied
- 17 feature and improvement requests fulfilled

Download latest version

Highlights of this release:

- Improved license counting
- Get support easily with the Support Tools Plugin
- Performance improvements
- Support for newer databases versions

Improved license counting

We've improved license counting so that duplicated usernames no longer consume more than one license.

See an example...

This table shows some examples of the changes in the Crowd 2.11 license system. Generally, a consumed license is a unique username which can authenticate against one or more applications in Crowd.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Crowd version &lt; 2.11 license count</th>
<th>Crowd version &gt;= 2.11 license count</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Application1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• directory1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• user1 inactive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• directory2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• user1 active</td>
<td>License count: 1, but user1 cannot authenticate, because was seen inactive in directory1.</td>
<td>0</td>
</tr>
<tr>
<td>• Application1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• directory1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• user1 active</td>
<td>2, but in fact only user1 from directory1 is able to authenticate against application1.</td>
<td>1</td>
</tr>
<tr>
<td>• Application1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• directory1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• user1 active</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>• Application2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• directory2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• user1 active</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Get support easily with the Support Tools Plugin

With Crowd 2.11 you can easily check your logs for common errors and use our support resources directly from Crowd, using the new Support Tools functionality.

Performance improvements

We’ve improved LDAP user search speed, which provides a better experience during searches and synchronisation.

Example of average initial synchronisation durations of 10,000 user base for Crowd 2.9.1, 2.10.1 and 2.11.0.
Support for newer databases versions

We've added support for MySQL 5.6 and 5.7, Postgres 9.5 and Microsoft SQL Server 2014. For a full list of supported platforms, you can check out our supported platforms page.

Complete list of improvements and fixes

<table>
<thead>
<tr>
<th>P</th>
<th>T</th>
<th>Key</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CWD-4484</td>
<td>spring-ldap:2.x causes slow search queries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-3826</td>
<td>Delegated Authentication directories do not track last authenticated time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4446</td>
<td>Official support for Microsoft SQL Server 2014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4429</td>
<td>Provide Support Zip tool on Crowd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4794</td>
<td>Postgres 9.5 support for Crowd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4190</td>
<td>Synchronization to AD directory fails intermittently or is taking a much longer time to finish after upgrading to Crowd 2.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4163</td>
<td>Plugin upgrade tasks not running after Atlassian Plugins upgrade</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-1779</td>
<td>Users can be duplicated in license counts if Crowd is setup with multiple links to the same external (LDAP) directory.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4569</td>
<td>Upgrade task that generates externalids (823) takes a long time to execute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4617</td>
<td>Add PostgreSQL 9.4 to the lists of Crowd supported databases</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4820</td>
<td>Own freemarker templates won't render in 2.10.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4735</td>
<td>Crowd license count same user in different directory as only 1 user license.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4845</td>
<td>Support MySQL 5.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4801</td>
<td>Provide a feature for Crowd to always re-evaluate group memberships in Active Directory with Incremental Sync</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4776</td>
<td>JNDI injection in SMTP configuration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CWD-4819</td>
<td>User rename should return 400 (Bad Request) when renaming to a user that already exists (returns 500 now)</td>
</tr>
</tbody>
</table>
CWD-4795  Update application-links to fix APL-1327
CWD-4796  Crowd HTTP client ignores SSL system properties
CWD-3076  Don’t count same user in multiple directories multiple times
CWD-4310  Disable "Follow Referrals" in the AD Connector by default
CWD-4798  Support for Mysql 5.7
CWD-4823  User updated timestamp should also be updated when user attributes are updated
CWD-4829  Remove ASAP dependencies from Crowd
CWD-4812  Concurrent REST calls attempting to create the same user can return a 500 error
CWD-4672  Logs indicate that an incremental sync is attempted even when it’s disabled
CWD-4766  Retry authentication attempt when authenticating via UI or REST
CWD-4809  Remove trusted applications support
CWD-4378  Stacktrace in the logs when requesting a password reset if the mail server hasn’t been set up yet
CWD-4576  Invalid requests to /appmanagement cause NPE and HTTP error 500
CWD-4375  In the CSV import screen, the 'User file' field should be marked as required
CWD-4376  In the Atlassian Importer screen, some fields should be marked as required
CWD-4787  Provide more tunable logging settings in the Crowd console
CWD-4773  User list shows next button in the wrong place
CWD-4597  As a client of the REST API, I want the response of POST /group to contain the group entity
CWD-4372  Lack of submission feedback in some administrative screens
CWD-4827  Improve attributes synchronization behavior
CWD-4657  Rename Stash to Bitbucket to match product’s rebrand
CWD-4846  Check Crowd WARs for duplicate JARs during the build

38 issues

Crowd 2.10.1 Release Notes

13 September 2016
The Atlassian Crowd team is pleased to bring you Crowd 2.10.1.
This release brings significant performance improvements, as well as many bug fixes.
We recommend you read the upgrade notes for important information about this release.

Thank you for your feedback:
⭐ 83 votes satisfied
⭐ 3 feature and improvement requests fulfilled

Download latest version

Highlights of this release:
- Faster than ever
• Improved group membership synchronization behavior
• Improved UI for LDAP configuration testing
• Complete list of improvements and fixes

Faster than ever

We’ve improved how Crowd performs with aliases and applications with many directories without aggregated memberships.

See an example...

One of the configurations we tested with was an application with disabled membership aggregation and the following directory configuration:

• 2 directories - one internal and one database cached LDAP directory
• 2402 users in 401 groups in the internal directory (1 default group and 200 parent groups each with 1 child group)
• 4000 users in 401 groups in the database cached LDAP directory
• 2400 shadowing users

In this example, it took approximately 26 seconds to return all groups with their users (via /rest/usermanagement/1/group/membership), as opposed to approximately 30 minutes prior to this change.

We think you’ll notice a significant difference in your Crowd instance.

Improved group membership synchronization behavior

We’ve added a new option that synchronizes group memberships for users who have been created in an LDAP directory, but have not yet been synchronised to Crowd. This gives you convenience, without sacrificing performance.

Improved UI for LDAP configuration testing

We’ve improved the look and usability of LDAP configuration testing to ensure better experience when verifying your connection parameters and filters.

Complete list of improvements and fixes

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>P</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWD-2650</td>
<td>Crowd doesn’t create new users automatically in other directories when he first login, but after sometime</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-4344</td>
<td>Significantly Slower Sync to Confluence or JIRA in Crowd 2.8 due to /rest/usermanagement/1/group/membership</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-4197</td>
<td>Improve authentication performance when copying groups on login</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-3255</td>
<td>Incremental sync fails after authentication if default groups have been added to a remote Crowd directory or the user has been synced via authentication</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-3809</td>
<td>ApplicationServiceGeneric#searchNestedGroupRelationships performs n+1 queries in order to shadow users</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-3906</td>
<td>Performance drop considerably when Alias is enabled</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4648</td>
<td>Local Crowd admin account is unable to login to Crowd when LDAP directory is locked out</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4790</td>
<td>CVE-2016-6496: LDAP Java Object Injection in Crowd</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-4745</td>
<td>Crowd application does not startup when license contains SEN</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4689</td>
<td>Improve isCanonical check in InMemoryNonAggregatingSearchStrategy</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-4666</td>
<td>Update struts to a version &gt;= 2.3.26</td>
<td></td>
<td>RESOLVED</td>
</tr>
</tbody>
</table>
Showing 30 out of 32 issues

Crowd 2.9.5 Release Notes

12th October 2016

The Atlassian Crowd team is pleased to bring you Crowd 2.9.5 with a number of fixes and improvements for Crowd 2.9.

Thank you for your feedback:

⭐ 36 votes satisfied

Download latest version

Highlights of this release:

- Complete list of improvements and fixes

Complete list of improvements and fixes

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>P</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWD-2650</td>
<td>Crowd doesn't create new users automatically in other directories when he first login, but after sometime</td>
<td></td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-4790</td>
<td>CVE-2016-6496: LDAP Java Object Injection in Crowd</td>
<td></td>
<td>CLOSED</td>
</tr>
</tbody>
</table>
Crowd 2.9.1 Release Notes

10 May 2016
The Atlassian Crowd team is pleased to bring you Crowd 2.9.1.

This release includes support for Oracle 12c, allows the use of Gravatars, and adds additional REST API functionality.

We recommend you read the upgrade notes for important information about this release.

Thank you for your feedback:

🌟 20 votes satisfied
🌟 Over 38 feature and improvement requests fulfilled

Download latest version

Highlights of this release:

- Oracle 12c support
- REST API improvements
- Complete list of improvements and fixes

Oracle 12c support

We've included support for Oracle 12c, which brings Crowd into line with other Atlassian applications such as JIRA, Confluence and Bitbucket Server.

REST API improvements

We've created new REST API functionality, and released new REST API documentation that includes the new functionality.

Complete list of improvements and fixes

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>P</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWD-3216</td>
<td>Crowd creates NULL DB values for any blank text box in a directory configuration</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4206</td>
<td>LDAP user unable to Login to application due to membership in restricted group</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3051</td>
<td>Require Java 7</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3735</td>
<td>Require Java 8</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4495</td>
<td>Upgrade to httpcomponents' httpclient 4.5</td>
<td></td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3512</td>
<td>For Internal and Caching directories, when searching for memberships you should be able to apply a search restriction</td>
<td></td>
<td>RESOLVED</td>
</tr>
</tbody>
</table>
### Installing Crowd

You can download Crowd [here](https://www.atlassian.com/software/crowd/download).  

**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  

<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWD-4651</td>
<td>Support for openJDK 1.8</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4271</td>
<td>As an admin, I want to bulk-expire SSO sessions</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3569</td>
<td>Support for Oracle 12c</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3292</td>
<td>Upgrade to Struts 2</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3222</td>
<td>Ability to expand on attributes of users for search in the REST endpoint</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3217</td>
<td>Unable to create XML backup from a fresh vanilla Crowd installation hooked with Oracle DB (NullPointerException)</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-2923</td>
<td>Creating new users using Crowd REST ignores &lt;attributes&gt; element</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4475</td>
<td>Retrieve avatars from LDAP for uncached directories</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4154</td>
<td>Create user operation fails to talk to new Crowd server</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4688</td>
<td>Session hijacking due to token identifier_hash collision</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4614</td>
<td>Base URL no longer resolved properly in Crowd 2.8.4</td>
<td>CLOSED</td>
</tr>
<tr>
<td>CWD-4595</td>
<td>Enable X-FRAME-Option in HTTP response headers in order to provide clickjacking protection</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4504</td>
<td>CrowdRestClient does not propagate user creation or updated dates to clients.</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4493</td>
<td>Race condition on user deletion allows a password reset token to be created for a nonexistent user</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4381</td>
<td>Clicking “Synchronize now” button when viewing a remote Crowd directory switches to viewing a LDAP connector</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4230</td>
<td>Expiring a password should invalidate the SSO token for that user</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3380</td>
<td>OpenIDServer vulnerable to clickjacking</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3718</td>
<td>Disabling a user should invalidate the SSO token for that user</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3664</td>
<td>Changing a password for a user should invalidate the SSO token for that user and all sessions</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3201</td>
<td>Return correct HTTP status for failure to create an application through /rest/appmanagement/1/application</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4466</td>
<td>XSRF protection bypass</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4287</td>
<td>Directory periodic synchronisation is not automatically re-started on Crowd restart</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-3225</td>
<td>Improve performance for paginated queries across multiple directories</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>CWD-4692</td>
<td>Documentation tasks for 2.9</td>
<td>RESOLVED</td>
</tr>
</tbody>
</table>

Showing 30 out of 96 issues
Supported Platforms

This page describes the supported platforms and hardware requirements for Crowd 2.11.

**Key:** ✅ = Supported. ❌ = Not Supported ❗️ = Deprecated

<table>
<thead>
<tr>
<th>Java Version</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle JDK (1)</td>
<td>✅ 1.8</td>
</tr>
<tr>
<td></td>
<td>Please note that JDK versions older than 8u65 may have problems connecting to LDAP servers over SSL - see CWD-4444 - Secure LDAP connections are broken when using Java 1.8u51, 1.8u60, 1.7.0_85+ and 1.6.0_101+ RESOLVED for more details.</td>
</tr>
<tr>
<td>OpenJDK</td>
<td>✅ 1.8</td>
</tr>
</tbody>
</table>

**Operating Systems**

- Microsoft Windows (2)
- Linux / Solaris (2)
- Apple Mac OS X (2)

**Application Servers**

- Apache Tomcat (3) ✅ 7.0.x (Crowd ships with Apache Tomcat 7.0.69)

**Databases**

- MySQL (4) ✅ 5.7
  ✅ 5.6
  ✅ 5.5
- Oracle ✅ 12c
### PostgreSQL

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>9.5</strong></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td><strong>9.4</strong></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td><strong>9.3</strong></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td><strong>9.2</strong></td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>

### Microsoft SQL Server

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2014</strong></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td><strong>2012</strong></td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>

**HSQLDB (5)**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>(For evaluation only.)</td>
</tr>
</tbody>
</table>

### Web Browsers

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chrome</strong></td>
<td>✔</td>
</tr>
<tr>
<td><strong>Microsoft Internet Explorer (Windows)</strong></td>
<td>✔</td>
</tr>
<tr>
<td><strong>Mozilla Firefox (all platforms)</strong></td>
<td>✔</td>
</tr>
<tr>
<td><strong>Safari</strong></td>
<td>✔</td>
</tr>
</tbody>
</table>

### 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 Oracle 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. Tomcat: Deploying multiple Atlassian applications in a single Tomcat container is not supported. We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration. See this FAQ for more information.

   There are also a number of practical reasons why we do not support deploying multiple Atlassian applications in a single Tomcat container. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

   Finally, we recommend not deploying any other applications to the same Tomcat container that runs Crowd, especially if these other applications have large memory requirements or require additional libraries in Tomcat's **lib** subdirectory.

4. MySQL: Please ensure that you set transaction isolation to 'read-committed' instead of the default 'repeatable-read', as described in the database configuration guide.

5. 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.

### 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

**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 favorite 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 favorite 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
    - 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

Installing Crowd and CrowdID

The instructions below tell you how to install the Crowd distribution (not EAR-WAR), 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. Prerequisites

- **Java.** You will need to install a Java Development Kit (JDK) on your operating system before proceeding with a Crowd installation. Please note that Crowd requires the full installation of a JDK. It is not enough to run Crowd on a Java Runtime Environment (JRE) alone. For instructions on installing the Sun JDK and setting `JAVA_HOME`, please refer to Supported Platforms.

2. Install Crowd

1. **Download Crowd** (not the EAR-WAR distribution).
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. (Note however that the CrowdID database will be in the installation directory, not the Home directory.) To specify the Crowd Home directory:
   - **Open the** `crowd-init.properties` **file. This is found at** `<crowd_install_directory>/crowd-webapp/WEB-INF/classes/crowd-init.properties`
   - **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
       ```
     - **Important**
       Please, ensure that the Crowd Home directory will not match the Crowd installation directory **AND** it is writable by the user executing the initialization script.

- **Save the** `crowd-init.properties` **file.**

3. Prepare your Database

   **For evaluators** This step applies to **production** installations. 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

4. 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

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 Crowd distribution (not EAR-WAR) 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

**HSQLDB**

The Crowd distribution (not EAR-WAR) 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.


**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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

MS SQL Server

Supported Versions
Crowd supports the versions of MS SQL Server listed on the Supported Platforms page.

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. Additionally, you should create the database with case sensitive collation.

   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
   4. Ensure that the new database was set to use Read Committed with Row Versioning as its isolation level.

   You can apply the new isolation by executing the following query:

   ```sql
   ALTER DATABASE <database name>
   SET READ_COMMITTED_SNAPSHOT ON
   WITH ROLLBACK IMMEDIATE;
   ```

   To verify the changes, use this query which should result in '1':
2. Copy the SQL Server Driver to your Application Server

This step is only required if you have downloaded the WAR distribution of Crowd.

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 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

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.

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:

   ```sql
   create database crowd character set utf8 collate utf8_bin;
   ```

3. Ensure that the user has permission to connect to the database, and create and populate tables:

   ```sql
   GRANT ALL PRIVILEGES ON crowd.* TO 'crowduser'@'localhost'
   IDENTIFIED BY 'crowdpass';
   ```

4. Edit the `my.cnf` file (often named `my.ini` on Windows operating systems) in your MySQL server. Locate the `[mysqld]` section in the file, and add or modify the following parameters (Refer to MySQL Option Files for detailed instructions on editing `my.cnf` and `my.ini`).:
   - Specify the default character set to be UTF-8:

     ```ini
     [mysqld]
     ...
     character-set-server=utf8
     collation-server=utf8_bin
     ...
     ```
   - Set the default storage engine to InnoDB:

     ```ini
     [mysqld]
     ...
     default-storage-engine=INNODB
     ...
     ```
   - Set `transaction-isolation = READ-COMMITTED`.

     ```ini
     [mysqld]
     ...
     transaction-isolation = READ-COMMITTED
     ```

   **Notes:**
   - 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-5.x.x-bin.jar) to the following directory:
   - For Crowd 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-5.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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

**Oracle**

**Supported Versions**
Crowd supports the versions of Oracle listed on the Supported Platforms page.

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**

   This step is only required if you have downloaded the WAR distribution of Crowd.

2. Add the Oracle JDBC driver jar to the following directory:
   - For Crowd 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

PostgreSQL

**Supported Versions**

Crowd supports the versions of PostgreSQL listed on the Supported Platforms page.

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 (for example, `crowduser`).
   2. Create a database for Crowd to store data in (for example, `crowddb`).
   3. Ensure that the user has permission to login to the database and can create database objects.

2. **Copy the PostgreSQL Driver to your Application Server**

   This step is only required if you have downloaded the WAR distribution of Crowd.

   1. Download the PostgreSQL JDBC driver from [http://jdbc.postgresql.org/download.html](http://jdbc.postgresql.org/download.html) and save it locally for later use.

   Internet Explorer may rename the file extension from `.jar` to `.zip` when you download it. If you are using Internet Explorer, please rename the file so that it has a `.jar` extension after downloading it.
If you have installed JDK 6.x, get JDBC4 Postgresql Driver, Version 8.4-701.
If you have JDK 5.x, get JDBC3 Postgresql Driver, Version 8.4-701.

2. Add the PostgreSQL JDBC driver jar to the following directory:
   - For Crowd 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

**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 distribution (not EAR-WAR) 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 distribution (not EAR-WAR) 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.
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
  - 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
- Setting Crowd to Start Automatically on Mac OS X
- Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

HSQLDB should not be used as a production database. It is included for evaluation purposes only.
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 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 customize the username, password, driverClassName and url parameters for the Datasource.
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 Crowd distribution, not Crowd EAR-WAR) 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
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:
        ```sql
        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 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 customize the `username`, `password`, `driverClassName` and `url` parameters for the Datasource.

```xml
<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/crowdiddb?autoReconnect=true&useUnicode=true&characterEncoding=utf8"
      [ 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 crowdiddb 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 Crowd distribution, not Crowd EAR-WAR) 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:
Documentation for Crowd 2.11

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.

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
    - 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

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


2. Add the Oracle JDBC driver jar to the following directory:
   - For Crowd 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 customize the `username`, `password`, `driverClassName` and `url` parameters for the Datasource.

   ```xml
   <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="oracle.jdbc.driver.OracleDriver"
      url="jdbc:oracle:thin:@localhost:1521:crowdidb"
      [ delete the minEvictableIdleTimeMillis, 
      timeBetweenEvictionRunsMillis and maxActive params here ]
      />

   <Manager className="org.apache.catalina.session.PersistentManager"
   saveOnRestart="false"/>

   </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 Crowd release, not Crowd EAR-WAR) and modify the `hibernate.dialect` to the following

   ```properties
   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:
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
  - 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

**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 (for example, crowduser).
   2. Create a database for CrowdID to store data in (for example, 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 to 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](http://jdbc.postgresql.org/download.html) and save it locally for later use.

   Internet Explorer may rename the file extension from ".jar" to ".zip" when you download it. If you are using Internet Explorer, please rename the file so that it has a ".jar" extension after downloading it.

   - If you have installed JDK 6.x, get JDBC4 Postgresql Driver, Version 8.4-701.
   - If you have JDK 5.x, get JDBC3 Postgresql Driver, Version 8.4-701.

2. Add the PostgreSQL JDBC driver JAR to the following directory:
   - For Crowd 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 customize the `username`, `password`, `driverClassName` and `url` parameters for the datasource.

   ```xml
   <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="org.postgresql.Driver"
   url="jdbc:postgresql://host:port/crowdiddb" [ see also http://jdbc.postgresql.org/doc.html ]"
   [ delete the minEvictableIdleTimeMillis, timeBetweenEvictionRunsMillis and maxActive params here ]
   />
   
   <Manager className="org.apache.catalina.session.PersistentManager"
   saveOnRestart="false"/>
   
   </Context>
   
   2. Delete the `minEvictableIdleTimeMillis`, `timeBetweenEvictionRunsMillis` and `maxActive` attributes. (These are only needed for HSQL database, and degrade performance otherwise.)

4. Configure CrowdID to use PostgreSQL

1. Edit the `build.properties` file located in the root of the Crowd distribution, and modify the `hibernate.dialect` to the following

   ```properties
   hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
   ```

2. Run `./build.sh` or `build.bat`. This will configure Crowd to use the PostgreSQL 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

```java
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. 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
    - 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

## Installing Crowd and CrowdID WAR Distribution

The Crowd and CrowdID WAR distributions are intended for deployment onto an existing J2EE application server. This documentation assumes 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 distribution instead.

The standard Crowd installation guide tells you how to install the Crowd distribution, which includes Apache Tomcat. Instead, you may wish to deploy Crowd or CrowdID onto your own existing application server. For this

---

*Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
purpose, we provide WAR (Webapp ARchive) distributions of the Crowd and CrowdID server applications. Crowd supports the application servers listed on the supported platforms page.

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 WAR Distribution

Below is a summary of the steps required to install the Crowd WAR distribution.

The Crowd and CrowdID WAR distributions are intended for deployment onto an existing J2EE application server. This documentation assumes 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 distribution instead.

The standard Crowd installation guide tells you how to install the Crowd distribution, 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.

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

See Supported Platforms.

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-INSTALLATION in the rest of these instructions.

4. Specify your Crowd Home directory by editing the configuration file at CROWD-INSTALLATION/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. (Note however that the CrowdID database will be in the installation directory, not the Home directory.) To specify the Crowd Home directory:

   • Open the crowd-init.properties file. This is found at <crowd_install_directory>/crowd-webapp/WEB-INF/classes/crowd-init.properties
   • 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. Deploy the Crowd files to your Tomcat application server. Depending upon your application server, you may need to zip up the WAR file again before deploying it. Place the CROWD-INSTALLATION directory or the WAR file into your application server's deployment directory. Please consult your application server’s documentation on this point.

6. Configure a Crowd context in your Tomcat application server:

   • Create a file called crowd.xml that contains the following context:

     ```xml
     <Context path="/crowd"
     docBase="/path/to/atlassian-crowd-war-directory"
     reloadable="false"/>
     ```

   • Modify the '/path/to/atlassian-crowd-war-directory' in the above element 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 on Windows, make sure that the paths you specify for the location of the WAR file and database are full paths including drive letters.
   • Place the file in Tomcat's conf/Catalina/localhost/ directory.

7. Create a database in your chosen database server and copy the database driver to your application server, as described in Connecting Crowd to a Database.

8. Optional: Modify Tomcat's server.xml to allow for a 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 internationalized characters in usernames.

   • 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:
<Connector port="8080" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" redirectPort="8443" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true"/>

- Add a `URIEncoding="UTF-8"` property to the connector:

  <Connector port="8080" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" redirectPort="8443" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true"
URIEncoding="UTF-8"/>

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 internationalized characters.

9. Restart your application server.
10. 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 CrowdID WAR Distribution

Below is a summary of the steps required to install the CrowdID WAR distribution.

The Crowd and CrowdID WAR distributions are intended for deployment onto an existing J2EE application server. This documentation assumes 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 distribution instead.

The standard Crowd installation guide tells you how to install the Crowd distribution, 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.

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

See Supported Platforms.

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](https://docs.oracle.com/javase/1.4.2/docs/api/com/sun/dava/transaction/TransactionManager.html) 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](https://crowd.atlassian.com/download). 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](https://crowd.atlassian.com/download).** 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-INSTALLATION in the rest of these instructions.

4. **Modify file CROWDID-INSTALLATION/WEB-INF/classes/crowd.properties to point to the port of your application server.** The default is 8080, as shown in the example below:

   ```
   crowd.server.url=http://localhost:8080/crowd/services/
   application.login.url=http://localhost:8080/crowd/console/
   ```

5. **Deploy the CrowdID files to your Tomcat application server.** Depending upon your application server, you may need to zip up the WAR file again before deploying it. Place the CROWDID-INSTALLATION directory or the WAR file into your application server's deployment directory. Please consult your application server's documentation on this point.

6. **Configure a CrowdID context in your Tomcat application server:**

   - **Create a file called openidserver.xml that contains the following context:**

     ```
     <Context path="/openidserver"
     docBase="/path/to/atlassian-crowd-openid-war-directory"
     reloadable="false" />
     ```

   - **Modify the '/path/to/atlassian-crowd-openid-war-directory' in the above element 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 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.

   - **Place the file in Tomcat's conf/Catalina/localhost/ directory.**

7. **Create a database in your chosen database server, copy the database driver to your application server, add the required datasource definition and edit the jdbc.properties file, as described in Connecting CrowdID to a Database.**

8. **Optional:** Modify Tomcat's server.xml to allow for a 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 internationalized characters in usernames.
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:

```
<Connector port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"/>
```

Add a `URIEncoding="UTF-8"` property to the connector:

```
<Connector port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"
    URIEncoding="UTF-8"/>
```

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 internationalized characters.

9. Restart your application server.
10. 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

**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. (Note however that the CrowdID database will be in the installation directory, not the Home directory.) To specify the Crowd Home directory:

- Open the `crowd-init.properties` file. This is found at `<crowd_install_directory>/crowd-webapp/WEB-INF/classes/crowd-init.properties`
- 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

Important
Please, ensure that the Crowd Home directory will not match the Crowd installation directory AND it is writable by the user executing the initialization script.

- 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:

-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 Crowd distribution (not Crowd EAR-WAR).

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:

<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. (Optional) 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)

The Crowd distribution (not EAR-WAR) 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:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>Select your database server type.</td>
</tr>
<tr>
<td>Driver Class Name</td>
<td>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.</td>
</tr>
<tr>
<td>JDBC URL</td>
<td>Enter the URL at which Crowd can access the database JDBC connection. Do note that if you wish to change the default database name, you can specify it in the URL as well. E.g. for SQL Server, the default URL is <code>jdbc:jtds:sqlserver://localhost:1433/crowd</code>, and if you wish to connect to crowddb instead, modify the URL so that it looks like this: <code>jdbc:jtds:sqlserver://localhost:1433/crowddb</code></td>
</tr>
<tr>
<td>Username</td>
<td>Enter the username which Crowd will use to access the database.</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the password corresponding to the above username.</td>
</tr>
</tbody>
</table>
### 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:
  
  ```java
  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 type of database you would like to use with Crowd.

- **Embedded**: The embedded database will allow Crowd to operate without an external database. This is useful when evaluating Crowd and not recommended for production systems.
- **JDBC Connection**: Connect to an external database using a JDBC connection.
- **JNDI Datasource**: Connect to an external database through a datasource managed by the application server.

#### JNDI Datasource Configuration

- **Database**: Select a database to connect to.
- **JNDI Name**: Enter the datasource name, e.g. `jdbc/CrowdDS` or `java:comp/env/jdbc/CrowdDS`.
- **Hibernate Dialect**: Only modify the Hibernate dialect if you require a variant dialect for your database type.
- **Overwrite Existing Data**: Overwrite any existing data in the database for a clean installation of Crowd.

[Continue >>]

---

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:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>Select your database server type.</td>
</tr>
<tr>
<td>JNDI Name</td>
<td>Enter the datasource name, e.g. <code>jdbc/CrowdDS</code> or <code>java:comp/env/jdbc/CrowdDS</code>.</td>
</tr>
</tbody>
</table>
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

**Import Existing Crowd Data**

Enter the Crowd XML backup file to upgrade from.

**File Location:**

The full file path to your existing data (e.g. C:\crowd\data\xml)

[Continue »]

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

**Options**

**Deployment Title:**

The name of this Crowd instance.

**Session Timeout:**

The number of minutes a session lasts before expiring. Must be greater than 0.

**Base URL:**

The base URL for the installation of Crowd.

[Continue »]

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. (Optional) 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. You can also skip this step. If you do so, Crowd will not be able to send email notifications until the mail configuration is entered via the Crowd Administration Console.

*If in any doubt, skip this step for evaluation purposes.*

**Mail Server Option 1: SMTP**

**Mail Configuration**

- **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** — This is useful for mail client filtering rules. For example: [ACME CORP - Crowd].

**Mail Server Details**

- **Mail Server Type** — SMTP Server / JNDI Location

**SMTP Server**

- **SMTP Host** — The host address. For example: localhost or smtp.acme.com.
- **SMTP Port** — 25
- **Username** — The username to use when connecting to the mail server.
- **Password** — The password to use when connecting to the mail server.
- **Use Secure Sockets Layer (SSL)** — SMTP server requires encryption.

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.

• **Use Secure Sockets Layer (SSL)** — Select this check-box if you want to access your mail server over SSL (Secure Sockets Layer). This ensures that all email communications between Crowd and your mail server are encrypted, provided your mail server supports SSL.

### Mail Server Option 2: JNDI Location

#### Mail Configuration

- **Notification Email Address**
- **From Email Address**
- **Subject Prefix**

#### Mail Server Details

- **Mail Server Type**
- **JNDI Location**

#### JNDI Location

- **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.

- **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 log in with the default administrator account you have just created, and 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

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.
<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
<th>Screenshot: 'Crowd Checklist'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Development Kit 1.5 or higher</td>
<td><em>(The screen will show the version of JDK detected in your system, with a red exclamation mark in the 'Status' column if insufficient.)</em> Refer to the System Requirements page for information about the JDK required and where you can get it.</td>
<td></td>
</tr>
<tr>
<td>Servlet 2.3 API or higher</td>
<td><em>(The screen will show the application server and version detected in your system, with a red exclamation mark in the 'Status' column if insufficient.)</em> 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.</td>
<td></td>
</tr>
<tr>
<td>Crowd Home directory</td>
<td>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.</td>
<td>Define the directory which you want Crowd to use as its 'home'. Read all about it in the installation guide.</td>
</tr>
</tbody>
</table>
The above screenshot shows a problem with the setting of the Crowd home directory.

RELEATED TOPICS

- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd

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
- Installing Crowd as a Windows Service
- Setting Crowd to Start Automatically on Mac OS X
- Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

RELEATED 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

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 Installation Directory

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. (Note however that the CrowdID database will be in the Crowd Installation directory, not the Home 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:
<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. (Note however that the CrowdID database will be in the Crowd Installation directory, not the Crowd Home 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 database Directory in the Crowd Installation Directory

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 customized:

```properties
## 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=\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.url=http://localhost:8095/demo

# OpenID server context root
openidserver.url=http://localhost:8095/openidserver

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hibernate.dialect</td>
<td>This parameter controls the database dialect the Hibernate persistence system will use when executing commands versus your database server.</td>
</tr>
<tr>
<td>hibernate.transaction.factory_class</td>
<td>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:</td>
</tr>
<tr>
<td></td>
<td>- org.hibernate.transaction.JDBCTransactionFactory delegates to database (JDBC) transactions (default).</td>
</tr>
<tr>
<td></td>
<td>- 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).</td>
</tr>
<tr>
<td>crowd.url</td>
<td>The path and port for the root of the Crowd Administration Console web-application.</td>
</tr>
<tr>
<td>demo.url</td>
<td>The path and port for the root of the Crowd demo web-application</td>
</tr>
<tr>
<td>openidserver.url</td>
<td>The path and port for the root of the CrowdID web-application</td>
</tr>
</tbody>
</table>

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
```

**The database Directory in the Crowd Installation Directory**

If you are using the embedded HSQL database, supplied for evaluation purposes, CrowdID will store its database in this directory. (Note however that the Crowd database will be in the Crowd Home directory, not the Installation directory.)

**RELATED TOPICS**

- Finding the atlassian-crowd.log File
- Supported Platforms
- Installing Crowd and CrowdID
- Running the Setup Wizard
- Configuring Crowd
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

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.name</td>
<td>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.</td>
</tr>
<tr>
<td>application.password</td>
<td>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.</td>
</tr>
<tr>
<td>application.login.url</td>
<td>Crowd will redirect the user to this URL if their authentication token expires or is invalid due to security restrictions.</td>
</tr>
<tr>
<td>crowd.server.url</td>
<td>The URL to use when connecting with the integration libraries to communicate with the Crowd server.</td>
</tr>
<tr>
<td>crowd.base.url</td>
<td>The URL used by Crowd to create the full URL to be sent to users that reset their passwords.</td>
</tr>
<tr>
<td>session.isauthenticated</td>
<td>The session key to use when storing a Boolean value indicating whether the user is authenticated or not.</td>
</tr>
<tr>
<td>session.tokenkey</td>
<td>The session key to use when storing a String value of the user's authentication token.</td>
</tr>
<tr>
<td>session.validationinterval</td>
<td>The number of minutes to cache authentication validation in the session. If this value is set to 0, each HTTP request will be authenticated with the Crowd server.</td>
</tr>
<tr>
<td>session.lastvalidation</td>
<td>The session key to use when storing a Date value of the user's last authentication.</td>
</tr>
</tbody>
</table>

The following optional attributes in the crowd.properties file allow further customization of the client:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Default Value (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>http.proxy.host</td>
<td>The name of the proxy server used to transport SOAP traffic to the Crowd server.</td>
<td>(none)</td>
</tr>
<tr>
<td>http.proxy.port</td>
<td>The connection port of the proxy server (must be specified if a proxy host is specified).</td>
<td>(none)</td>
</tr>
<tr>
<td>http.proxy.username</td>
<td>The username used to authenticate with the proxy server (if the proxy server requires authentication).</td>
<td>(none)</td>
</tr>
<tr>
<td>http.proxy.password</td>
<td>The password used to authenticate with the proxy server (if the proxy server requires authentication).</td>
<td>(none)</td>
</tr>
<tr>
<td>http.max.connections</td>
<td>The maximum number of HTTP connections in the connection pool for communication with the Crowd server.</td>
<td>20</td>
</tr>
<tr>
<td>http.timeout</td>
<td>The HTTP connection timeout (milliseconds) used for communication with the Crowd server. A value of zero indicates that there is no connection timeout.</td>
<td>5000</td>
</tr>
</tbody>
</table>
**cookie.domain**  
A domain to use when setting cookies, overriding the SSO Domain set in Crowd (since Crowd 2.5.2).

When an SSO Domain is set in Crowd, all client applications must be in the same domain so cookies can be shared.

A Crowd deployment may have hosts with no common domain suffix, for example 'domain.example.com' and 'domain.internal'. Even though a user has already logged in to 'domain.example.com' and has a cookie set, applications running under 'domain.internal' will not receive this cookie since the domains differ, and users will be unable to log in.

Set this property in the crowd client application to override the domain. Applications within the same domain will then be able to share SSO sessions.

---

**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.

---

**socket.timeout**  
The socket timeout in milliseconds. You may wish to override the default value if the latency to the Crowd server is high.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20000</td>
<td>Passing crowd.properties as an Environment Variable</td>
</tr>
</tbody>
</table>

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
```

---

**Passing crowd.properties 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.sh` (or `build.bat`) 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
  - 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
  - Setting Crowd to Start Automatically on Mac OS X
  - Setting Crowd to Run Automatically and Use an Unprivileged System User on UNIX

**Configuring Crowd to Work with SSL**

Atlassian applications allow the use of SSL within our products, however Atlassian Support does not provide assistance for configuring it. Consequently, Atlassian **cannot guarantee providing any support for it.**

- If assistance with conversions of certificates is required, please consult with the vendor who provided the certificate.
- If assistance with configuration is required, please raise a question on Atlassian Answers.

**Why should you enable HTTPS access to Crowd?**

When web applications are accessed across the internet, there is always the possibility of usernames and passwords being intercepted by intermediaries. HTTPS is a good way to safeguard your Crowd data and user logins from being intercepted and read by outsiders.

**On this page:**

- Using Crowd over HTTPS
  - Step 1: Enable Tomcat HTTPS 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`
- Troubleshooting
• Using SSL between an LDAP Server and Crowd
• Microsoft Active Directory Connector using SSL Certificate
• Other LDAP Servers

Using Crowd over HTTPS

The process of enabling HTTPS 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 HTTPS Access

Edit <crowd installation>/apache-tomcat/conf/server.xml, and at the bottom before the </Service> tag (not to be confused with the </Server> tag!), add this section (or uncomment it if it's already there):

```xml
<Connector port="8443"
  maxThreads="150" SSLEnabled="true" scheme="https"
  secure="true"
  clientAuth="false" ssiProtocol="TLS"
  keystoreFile="${user.home}/.keystore" keystorePass="changeit"
  keyAlias="tomcat" keyPass="changeit"/>
```

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). You may need to change the values of keystoreFile, keystorePass and keyPass as appropriate for your certificates and set-up.

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:

- `%JAVA_HOME%/bin/keytool -genkey -alias tomcat -keyalg RSA` (Windows)
- `$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA` (Unix / Mac OS)

When you are asked for your "first and last name", instead supply the hostname for the Crowd server, e.g.:

```
What is your first and last name?
[Unknown]:  localhost
```

The keytool utility will prompt you for two passwords: the keystore password and the key password for Tomcat. You can use either of:

1. 'changeit' (this is the default value Tomcat expects), or
2. Any value other than 'changeit', and you must also specify it as the value of keystorePass in conf/server.xml.

You will then need to import your certificate into the truststore:

1. First, export the key you generated to a file:

```
$JAVA_HOME/bin/keytool -export -alias tomcat -file tomcat.cert
```
2. Import the key into the JRE keystore (you will need permission to write to the keystore specified, and may need elevated privileges):

```
$JAVA_HOME/bin/keytool -import -alias tomcat -file tomcat.cert -keystore $JAVA_HOME/jre/lib/security/cacerts
```

For information on adding a key pair issued by a Certificate Authority (CA), refer to the Apache Tomcat documentation.

**Importing a CA-Issued Certificate**

When using certificates issued by a Certificate Authority, you also need to import the certificate using the `keytool` command, rather than generating a self-signed key.

Here is an example of the command:

```
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 Oracle documentation for more information on the `keytool`:

- Solaris and Linux
- Windows

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:

```
#Wed Apr 09 12:36:21 EST 2008
session.lastvalidation=session.lastvalidation
session.isauthenticated=session.isauthenticated
application.password=password
application.name=crowd
session.validationinterval=0
crowd.server.url=https\://localhost\:8443/crowd/services/
session.tokenkey=session.tokenkey
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 connect to the web service.

Now start (or restart) your Crowd instance. You should be able to access Crowd at this URL:
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 .key  

testore 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

Installing Crowd as a Windows Service

If you are trying to set up Crowd as a Windows Service on a 64 bit machine, you should refer to the workaround in this document.
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 Command prompt as Administrator (search for "cmd" in the start menu/screen, right click and "Run as Administrator").
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:

   ```bash
   service.bat install Crowd
   ```

   ![Screenshot: Installing Crowd as a Windows Service](image)

   Crowd should now have been installed as a service, and will be visible in the Windows Services console.

   ![Screenshot: Windows Services Console](image)

6. Run the following command, to have the Crowd service start automatically when the server starts:
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 `regedit32.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:

  ```
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:

  ```
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:
    ```
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:
    ```
tomcat6 //US//Crowd
++JvmOptions="-Xloggc:c:\crowdinstall\logs\atlassian-gc.log"
  ```

- In order to check all the configurations for the service, you can access:

  ```
  HKEY_LOCAL_MACHINE -> SOFTWARE -> Apache Software Foundation -> Procrun 2.0 -> Crowd -> Parameters -> Java -> Options
  ```

- 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.
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.

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

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
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:
     ```
     service.bat remove Crowd
     ```
   - Or if the above does not work, use
     ```
     tomcat5 \DS\Crowd
     ```
Problem with JDK 6

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

Please refer to our Knowledge Base article if you need more details of this issue.

Notes for Windows 64-bit Operating Systems

If you are running 64-bit Windows, please note that Apache Tomcat cannot run as a Windows service if you are using a 64-bit JDK. Please ensure that you are using a 32-bit JDK. This is the recommended solution.

Alternatively, you can 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\catalina.exe with one compiled for 64-bit from these locations:

- http://svn.apache.org/viewvc/tomcat/tc5.5.x/tags/TOMCAT_5_5_24/connectors/procrun/bin/
- http://svn.apache.org/viewvc/tomcat/tc6.0.x/tags/TOMCAT_6_0_16/res/procrun/

Setting Crowd to Start Automatically on Mac OS X

For long-term use, you should configure Crowd to restart automatically when the operating system restarts. On Mac OS X, the system startup program called launchd manages long running processes – daemons or services.

Apple provides an introduction to launchd. Below we tell you how to use launchd to start Crowd automatically on Mac OS X when running Tomcat.

On this page:

- Using launchd with Tomcat
  - Step 1. Add a Wrapper Shell Script
  - Step 2. Add a launchd Property List
  - Starting and Stopping Crowd Manually
  - Troubleshooting

Using launchd with Tomcat

The Crowd distribution (not EAR-WAR) ships with Tomcat. There is a mismatch between how launchd expects a daemon to behave, and how the default startup scripts for Tomcat operate:

- OS X's launchd expects the process it starts to run forever, but 'catalina.sh start' starts the JVM to run Tomcat and then exits.
- Tomcat provides 'catalina.sh stop' to shut down Tomcat cleanly by connecting to a socket which Tomcat listens on, but launchd stops daemons by sending them a signal that kills the process immediately if no specific handling is included.
You will need a wrapper shell script and properties list to make launchd work with Tomcat.

**Step 1. Add a Wrapper Shell Script**

Add the following wrapper shell script to $CATALINA_HOME/bin:

```bash
#!/bin/bash

function shutdown()
{
    date
    echo "Shutting down Crowd"
    $CATALINA_HOME/bin/catalina.sh stop
}

date
echo "Starting Crowd"
export CATALINA_PID=/tmp/$$

# Uncomment to increase Tomcat's maximum heap allocation
# export JAVA_OPTS=-Xmx512M $JAVA_OPTS

. $CATALINA_HOME/bin/catalina.sh start

# Allow any signal that would kill a process to stop Tomcat
trap shutdown HUP INT QUIT ABRT KILL ALRM TERM TSTP

echo "Waiting for `cat $CATALINA_PID`"
wait `cat $CATALINA_PID`
```

The above shell script starts Tomcat and then waits for the process to complete, so launchd is happy that Tomcat is still running. The script also installs a signal handler, which calls the `shutdown()` function to cleanly shut down Tomcat when launchd signals the script.

You can try this script manually: Start the script, watch Crowd start, and then type `ctrl-C` and see Crowd shut down cleanly. (Note that it will not shut down cleanly if Tomcat has not started yet. It takes a few seconds for Tomcat to start listening on the shutdown socket.)

**Step 2. Add a launchd Property List**

The launchd property list (.plist) tells launchd how to start Tomcat.

Add the following plist file to /Library/LaunchDaemons, which is the location for system-wide services which are not part of base OS X:

```xml
<key>Label</key><string>Crowd</string>
<key>Disabled</key><false/>
<key>ProgramArguments</key><string>$CATALINA_HOME/bin/catalina.sh start</string>
<key>StartDelayed</key><false/>
<key>KeepAlive</key><true/>
<key>KeepAliveInterval</key><number>120</number>
<key>KeepAliveDelay</key><number>15</number>
<key>DisabledReason</key><string></string>
<key>RunAtLoad</key><false/>
<key>RunAtLoadDelay</key><number>5</number>
<key>StartLimitPolicy</key><string>Soft</string>
<key>StartLimitBurst</key><integer>10</integer>
<key>StartLimitSoft</key><integer>60</integer>
<key>MaxLaunchLimit</key><integer>256</integer>
```
crowd.plist

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <dict>
    <key>Disabled</key>
    <false/>
    <key>EnvironmentVariables</key>
    <dict>
      <key>CATALINA_HOME</key>
      <string>/Users/myname/conf/crowd-x.x.x</string>
      <key>JAVA_HOME</key>
      <string>/Library/Java/Home</string>
    </dict>
    <key>Label</key>
    <string>com.atlassian.crowd</string>
    <key>OnDemand</key>
    <false/>
    <key>ProgramArguments</key>
    <array>
      <string>/Users/myname/conf/crowd-x.x.x/bin/launchd_wrapper.sh</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
    <key>ServiceDescription</key>
    <string>Crowd</string>
    <key>StandardErrorPath</key>
    <string>/Users/myname/conf/crowd-x.x.x/logs/launchd.stderr</string>
    <key>StandardOutPath</key>
    <string>/Users/myname/conf/crowd-x.x.x/logs/launchd.stdout</string>
    <key>UserName</key>
    <string>root</string>
  </dict>
</plist>
```

Notes:

1. Replace `/Users/myname/conf/crowd-x.x.x` with the path to your Crowd installation. The string occurs four times in the above script.
2. JAVA_HOME is set to use the default JDK. On OS X version 10.4.4, the default JDK is 1.4.2. You will need to change this value if you want to use a different version of Java. For example, if you want to use JDK 1.5, you will need to change JAVA_HOME to `/System/Library/Frameworks/JavaVM.framework/Versions/1.5`.
3. In the above script, we have specified `root` as the UserName. If necessary, change the UserName to the user you want Tomcat to run as.

**Starting and Stopping Crowd Manually**

To start and stop Crowd manually, use the following commands:

- **Start:**
  ```
  cd /Library/LaunchDaemons
  ```
sudo launchctl load -w crowd.plist

- Stop:
  cd /Library/LaunchDaemons
  sudo launchctl unload -w crowd.plist

Troubleshooting

- Make sure both files launch_wrapper.sh and crowd.plist have the necessary file privileges.
- Check the console logging and log file for any abnormalities.

RELATED TOPICS

Configuring Crowd

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.

```bash
#!/bin/bash
CROWD_USER="crowd"
CROWD_GROUP="crowd"
INSTALL_BASE="/opt/crowd/atlassian-crowd-2.2.2"
CROWD_HOME="/var/crowd-home"
sudo chgrp ${CROWD_GROUP} ${INSTALL_BASE}/(*.sh,apache-tomcat/bin/*.sh)
sudo chmod g+x ${INSTALL_BASE}/(*.sh,apache-tomcat/bin/*.sh)
sudo chown -R ${CROWD_USER} ${CROWD_HOME}
${INSTALL_BASE}/apache-tomcat/{logs,work,temp}
sudo touch -a ${INSTALL_BASE}/atlassian-crowd-openid-server.log
sudo mkdir ${INSTALL_BASE}/database
sudo chown -R ${CROWD_USER} ${INSTALL_BASE}/database

Getting Crowd to Start Automatically

1. Create an init.d file (for example, 'crowd.init.d') inside your {CROWD_INSTALL} directory:
#!/bin/bash
# Crowd startup script
#chkconfig: 2345 80 05
#description: Crowd

# Based on script at http://www.bifrost.org/problems.html

RUN_AS_USER=crowd
CATALINA_HOME=/opt/crowd/atlassian-crowd-2.2.2/apache-tomcat

start() {
    echo "Starting Crowd: ">
    if [ "x$USER" != "x$RUN_AS_USER" ]; then
        su - $RUN_AS_USER -c "$CATALINA_HOME/bin/startup.sh"
    else
        $CATALINA_HOME/bin/startup.sh
    fi
    echo "done."
}

stop() {
    echo "Shutting down Crowd: ">
    if [ "x$USER" != "x$RUN_AS_USER" ]; then
        su - $RUN_AS_USER -c "$CATALINA_HOME/bin/shutdown.sh"
    else
        $CATALINA_HOME/bin/shutdown.sh
    fi
    echo "done."
}

case "$1" in
    start)
        start
    ;;
    stop)
        stop
    ;;
    restart)
        stop
        sleep 10
        #echo "Hard killing any remaining threads.."
        #kill -9 `cat $CATALINA_HOME/work/catalina.pid`
        start
    ;;
    *)
        echo "Usage: $0 {start|stop|restart}"
esac

exit 0

2. Create a symbolic link from /etc/init.d/crowd to the init.d file file.

**Hint for Red Hat systems**
On Red Hat and Red Hat-based systems such as CentOS, if you put the above script in /etc/init.d,
Upgrading Crowd

Below are instructions on upgrading an existing Crowd installation to the latest version of Crowd. There are two upgrade procedures to choose from:

- **Method 1: Automatic database upgrade.** Install the new version of Crowd and simply point it at your existing home directory. The upgrade procedure automatically updates your Crowd database.
- **Method 2: Data transfer via XML backup.** Back up your Crowd database to XML before starting the upgrade, install the new version of Crowd and then import the data into your new Crowd installation.

**Recommended Upgrade Procedure**

If you are upgrading from Crowd 2.0 or later, you can use method 1, automatic database upgrade or method 2, data transfer via XML backup.

**Upgrading from Crowd 1.6 or earlier**

If you are upgrading from a version of Crowd prior to 2.0, please make your choice based on your database server, the version of Crowd you are upgrading from and the version you are upgrading to.

- If you are using PostgreSQL, MySQL or Microsoft SQL Server and:
  - Upgrading from Crowd 1.3 or later, to Crowd 2.0.4 or later – use method 1, automatic database upgrade.
  - Upgrading from Crowd 1.2 or earlier, to Crowd 2.0.4 or later – use method 2, data transfer via XML backup.
- If you are using any other database server – use method 2, data transfer via XML backup

**Alternatives**

These are some options you may like to consider:

- If you prefer method 2, data transfer via XML backup, you can choose that option for any database server and no matter which version of Crowd you are upgrading from or to.
- If you are upgrading from Crowd 1.2 or earlier, are using PostgreSQL, MySQL or Microsoft SQL Server, and cannot perform an XML backup:
  1. Upgrade to Crowd 1.6 first, following the instructions in the Crowd 1.6 upgrade guide.
  2. Then upgrade from Crowd 1.6 to Crowd 2.0.4 or later, using the automatic database upgrade as described in the Crowd 2.0 upgrade guide.
- If for some reason you must upgrade to Crowd 2.0.0, 2.0.1, 2.0.2 or 2.0.3 (and cannot upgrade to Crowd 2.0.4), follow method 2, data transfer via XML backup.

**RELATED TOPICS**

- Crowd Release Notes
- Installing Crowd
- Upgrading Crowd
- Migrate to Another Database
- Migrating Crowd Between Servers
Migrating from OnDemand to a Crowd installed site

Upgrading Crowd via Automatic Database Upgrade

Below are instructions on upgrading an existing Crowd installation to the latest version of Crowd, using the automatic database upgrade.

Check that this is the right upgrade procedure for you
Please check that you have chosen the recommended upgrade procedure for your database server and Crowd version before you start.

On this page:
- Preparation: Read the Release Notes and Upgrade Notes
- Step 1. Shut Down Crowd and All Integrated Applications
- Step 2. Back Up your Crowd Files
- Step 3. Re-Install Crowd
- Step 4. Update your Integrated Applications
- Step 5. Start Crowd
- 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 2.11.0 Upgrade Notes
  - Crowd 2.10.1 Upgrade Notes
  - Crowd 2.9.1 Upgrade Notes
  - Crowd 2.8 Upgrade Notes
  - Crowd 2.7 Upgrade Notes
  - Crowd 2.6 Upgrade Notes
  - Crowd 2.5 Upgrade Notes
  - Crowd 2.4 Upgrade Notes
  - Crowd 2.3 Upgrade Notes
  - Crowd 2.2 Upgrade Notes
  - Crowd 2.1 Upgrade Notes
  - Crowd 2.0 Upgrade Notes
  - Crowd 1.6 Upgrade Notes
  - Crowd 1.5 Upgrade Notes
  - Crowd 1.4 Upgrade Notes
  - Crowd 1.3 Upgrade Notes
  - Crowd 1.2 Upgrade Notes
  - Crowd 1.1 Upgrade Notes
  - Crowd 1.0 Upgrade Notes

Step 1. Shut Down Crowd and All Integrated Applications

Shut down Crowd and all Crowd-connected applications.

Step 2. 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:
   - Back up your Crowd Home directory, in the location specified in the crowd-init.properties file — recommended in case something goes wrong during the upgrade process.
   - If your existing Crowd installation is version 1.3.x or 1.4.x: Back up the crowd.properties file for the Crowd Administration Console application, located at
Step 3. Re-Install Crowd

1. Download Crowd.

2. Unzip the download 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. Point the new Crowd installation at your existing 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. (Note however that the CrowdID database will be in the installation directory, not the Home directory.) To specify the Crowd Home directory:
   - Open the crowd-init.properties file. This is found at <crowd_install_directory>/crowd-webapp/WEB-INF/classes/crowd-init.properties.
   - 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
       ```
     - **Important**
       Please, ensure that the Crowd Home directory will not match the Crowd installation directory AND it is writable by the user executing the initialization script.
     - On Mac and UNIX-based systems:
       ```
crowd.home=/var/crowd-home
       ```

   - Save the crowd-init.properties file.

4. Copy the following files, saved in Step 2 above, to your new Crowd installation:
   - If your existing Crowd installation is version 1.3.x or 1.4.x: Copy the crowd.properties file for...
the Crowd Administration Console to the root of your Crowd Home directory.

As from Crowd 1.5, the crowd.properties file is located in the Home directory and not the Installation directory. This step is not required if your current Crowd installation is 1.5 or later.

- Copy the crowd.properties file for the CrowdID application to your new \{CROWD_INSTALL\}/crowd-openidservlet-webapp/WEBINF/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 customized configuration files.

Step 4. Update your Integrated Applications

1. 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.
2. If you are using CrowdID with an external database, you will will need to use the manual JNDI datasource configuration method to configure an external database connection.
3. If you are using CrowdID with the default HSQL database, copy the database/ directory from your old installation directory into your new installation directory. Please note that the HSQL database is not suitable for production environments. Connecting CrowdID to a Database describes how to migration to an enterprise database.

Step 5. Start Crowd

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. You should now be able to use the Crowd Administration Console.

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
- Migrate to Another Database
- Migrating Crowd Between Servers
- Migrating from OnDemand to a Crowd installed site

Upgrading Crowd via XML Data Transfer

Below are instructions on upgrading an existing Crowd installation to the latest version of Crowd, using the procedure that transfers your Crowd data via XML backup.

Check that this is the right upgrade procedure for you
Please check that you have chosen the recommended upgrade procedure for your database server and Crowd version before you start.

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.

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 2.11.0 Upgrade Notes
  • Crowd 2.10.1 Upgrade Notes
  • Crowd 2.9.1 Upgrade Notes
  • Crowd 2.8 Upgrade Notes
  • Crowd 2.7 Upgrade Notes
  • Crowd 2.6 Upgrade Notes
  • Crowd 2.5 Upgrade Notes
  • Crowd 2.4 Upgrade Notes
  • Crowd 2.3 Upgrade Notes
  • Crowd 2.2 Upgrade Notes
  • Crowd 2.1 Upgrade Notes
  • Crowd 2.0 Upgrade Notes
  • Crowd 1.6 Upgrade Notes
  • Crowd 1.5 Upgrade Notes
  • Crowd 1.4 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/WEB-INF/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 — You will need to copy this file to your new Crowd installation.
   • Your customized 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

Step 5. Start Crowd and Run the Setup Wizard

Step 6. Update your Integrated Applications

Troubleshooting
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. (Note however that the CrowdID database will be in the installation directory, not the Home directory.) To specify the Crowd Home directory:
   - Open the `crowd-init.properties` file. This is found at `<crowd_install_directory>/crowd-webapp/WEB-INF/classes/crowd-init.properties`
   - 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 customized configuration files.
   - **Important**
     Please, ensure that the Crowd Home directory will not match the Crowd installation directory AND it is writable by the user executing the initialization script.

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. 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.

2. 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.

3. If you are using CrowdID with the default HSQL database, copy the database/ directory from your old installation directory into your new installation directory. Please note that the HSQL database is not suitable for production environments. Connecting CrowdID to a Database describes how to migration to an enterprise database.

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
- Migrate to Another Database
- Migrating Crowd Between Servers
- Migrating from OnDemand to a Crowd installed site

Upgrade Notes

- Crowd 2.11.0 Upgrade Notes
- Crowd 2.10.1 Upgrade Notes
- Crowd 2.9.1 Upgrade Notes
- Crowd 2.8 Upgrade Notes
- Crowd 2.7 Upgrade Notes
- Crowd 2.6 Upgrade Notes
- Crowd 2.5 Upgrade Notes
- Crowd 2.4 Upgrade Notes
- Crowd 2.3 Upgrade Notes
- Crowd 2.1 Upgrade Notes
- Crowd 2.0 Upgrade Notes
- Crowd 1.6 Upgrade Notes
- Crowd 1.5 Upgrade Notes
- Crowd 1.4 Upgrade Notes
- Crowd 1.3 Upgrade Notes
- Crowd 1.2 Upgrade Notes
- Crowd 1.1 Upgrade Notes
- Crowd 1.0 Upgrade Notes
- Crowd 2.2 Upgrade Notes

Crowd 2.11.0 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 2.11. We highly recommend that you read the new features described in the Crowd 2.11 Release Notes.

On this page:
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.

Dropped support for older versions of databases

We've stopped supporting databases which were deprecated in Crowd 2.10. For the full list of supported databases you can click here.

System properties for Crowd HTTP client

Crowd's HTTP client (used for connecting to remote directories) will now use `javax.net.ssl properties` for SSL configuration. This can be used, for example, to override the default truststore or provide a keystore for SSL client-side authentication.

Refer to the Crowd 2.10.1 upgrade notes

If you have not yet upgraded to Crowd 2.10.1, please read the Crowd 2.10.1 upgrade notes before upgrading to Crowd 2.11.

We have dropped support for Trusted Applications

There are security risks related to using Trusted Applications for Application Links, and Atlassian Application Links haven't support Trusted Applications since Applinks version 5.2, so we've dropped support for Trusted Applications in this release.

Upgrade procedure

To upgrade to a Crowd 2.11.x release from any earlier version of Crowd, please follow these upgrade instructions.

Crowd 2.10.1 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 2.10.1 or later. We highly recommend that you read the new features described in the Crowd 2.10.1 Release Notes.

On this page:

- Upgrade notes
- 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.

HTTP redirects returned by Crowd now use relative Location header

The 'Location' headers returned with 302 Found responses are now relative urls, rather than absolute ones. This is consistent with RFC 7231. Most customers shouldn't notice this change, but if you are using a custom client or proxy, that expects these urls to be absolute, you might need to reconfigure them.

Event processing inside Crowd (for plugin developers)

Some semantics regarding processing of events in Crowd during a database transaction have changed. Events published during a transaction are only published after that transaction commits, and no longer share a transaction with the event publishing. See

CWD-4740 - Clarify the interaction of events and database transactions

CLOSED

for more details.

New behavior for synchronizing group memberships on user authentication

Crowd 2.10.1 allows you to synchronize group memberships on log in for newly added users only. This option
synchronizes group memberships for users who have been created in the LDAP directory, but not yet synchronized to Crowd.

This is the default behavior when creating new connector directories in Crowd 2.10.1. Existing directories are not migrated to the new behavior, but can be configured to use it, which will improve performance.

Refer to the Crowd 2.9 upgrade notes

If you have not yet upgraded to Crowd 2.9, please read the Crowd 2.9.1 upgrade notes before upgrading to Crowd 2.10.1

Upgrade procedure

To upgrade to a Crowd 2.10.x release from any earlier version of Crowd, please follow these upgrade instructions.

Crowd 2.9.1 Upgrade Notes

This document contains notes on upgrading an existing Crowd installation to Crowd 2.9.1 or later. We highly recommend that you read the new features described in the Crowd 2.9.1 Release Notes.

On this page:

- Upgrade notes
- 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.

Java 8 is required

Crowd 2.9.1 requires Java 8 to run. You can upgrade or install Oracle JDK 8 or OpenJDK 8.

Please note that JDK versions older than 8u65 may have problems connecting to LDAP servers over SSL - see CWD-4444 - Secure LDAP connections are broken when using Java 1.8u51, 1.8u60, 1.7.0_85+ and 1.6.0_101+ for more information.

Refer to the Crowd 2.8 upgrade notes

If you have not yet upgraded to Crowd 2.8, please read the Crowd 2.8 upgrade notes before upgrading to Crowd 2.9.1

Configuring compression of server output is no longer supported

As of Crowd 2.9.1 configuring Compression of Server Output is no longer supported. All responses will be gzip compressed by default

Plugin developers only - Plugins 4.0 upgrade

Crowd 2.9.1 ships with an upgraded plugins system. Please refer to the Atlassian Plugins 4.0 Upgrade Guide for a detailed list of changes introduced by the upgrade.

Upgrade procedure

To upgrade to a Crowd 2.9.x release from any earlier version of Crowd, please follow these upgrade instructions.

Migrate to Another Database

This guide applies to situations when you may need to migrate Crowd to another database.

Preparation
1. Select a new database from one of our Supported Platforms

2. 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.

3. If there is change to the Crowd IP address (i.e. when migrating the instance to an entirely new server), please refer to the additional Preparation steps in this article to prevent your admin being locked out of Crowd: Migrating Crowd Between Servers

   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 Customizations**

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 customizations, 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 and Upgrade Guide
- Running the Setup Wizard

**Migrating Crowd Between Servers**

This guide applies to situations when you may need to migrate Crowd to a new server, because:

- Your Crowd server hardware 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, if these will be different from the original server’s:
   - 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.
   - Ensure that the address list includes at least the following items: ‘localhost’, ‘127.0.0.1’ and the IP address or hostname of the machine that is going to receive the new Crowd instance. This list determines the hosts that can access the Crowd Administration Console.

3. Perform a file system backup of the Crowd Home and Crowd Installation directories.

4. Perform a full database backup.

   From this point on, we will call your existing Crowd server the 'original' server.

Migration

1. Restore the file system and database backups on to the target server.

2. Locate the crowd.properties file in the target server’s Crowd Home directory. Edit the file and modify the line `crowd.server.url` so that it points to your new Crowd server.

3. Locate the crowd-cfg.xml file in the target server’s Crowd Home directory. Modify it to point to the new database, if the name/location of it has changed, in this property: `<property name="hibernate.connection.url">`

4. Locate the crowd-init.properties file in your target server's Crowd Installation directory. Modify it to point to the new Crowd Home if the location of it has changed.

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. If your original server serves using HTTPS and the migrated server is on HTTP, you would need to turn off secure cookie with the following SQL query:

   ```sql
   UPDATE CWD_PROPERTY SET PROPERTY_VALUE = 'false' WHERE PROPERTY_NAME = 'secure.cookie' AND PROPERTY_KEY = 'crowd';
   ```

3. 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 Customizations

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, if the URL is different from the original server's URL.

2. If you have installed any Crowd plugins or added other customizations, 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
Migrating from OnDemand to a Crowd installed site

You can extract your user data from a Confluence or JIRA application installed site instance. Use these queries to export your users and memberships to CSV, and then import them into Crowd.

On this page:
- Migrate JIRA Cloud or Confluence Cloud
- Exporting user data
- Retrieving the backup data
- What data is backed up
- Importing the data into Crowd

Migrate JIRA Cloud or Confluence Cloud

First you'll need to migrate JIRA/Confluence Cloud to your server versions:

- Migrate from Confluence Cloud to Server
- Migrating from JIRA Cloud to Server applications

Exporting user data

If you are using Postgres as your database with JIRA or Confluence, you can generate the required CSV files from either of those systems, with the following queries. If you are using a different database, you may need to tweak these.

To generate users.csv

```
// Run on JIRA/Confluence
COPY (SELECT user_name AS "Username", first_name AS "First Name",
last_name AS "Last Name", email_address AS "Email Address", credential
AS "Password" FROM cwd_user) TO '/tmp/users.csv' CSV HEADER;
```

To generate group_memberships.csv

```
// Run on JIRA/Confluence
COPY (SELECT DISTINCT child_name AS "Username", parent_name AS
"Groupname" FROM cwd_membership) TO '/tmp/group_memberships.csv' CSV
HEADER;
```

Make sure that CSV files are encoded in UTF-8 (Unicode) to ensure compatibility for user data containing special or accented characters.

Retrieving the backup data

The files will be generated in /tmp, or in the folder specified in the query above.

What data is backed up
The backup includes the following data:

- User accounts
- Groups
- Memberships

**Importing the data into Crowd**

1. Follow the instructions in Importing Users from CSV Files
2. Connect Crowd to your applications: Adding an Application
3. Ensure that the Crowd Directory is placed above the Internal Directory in your User Directories in all downstream applications: Configuring User Directories. This is important because you will have the same users in both directories, and the users will authenticate against the highest directory in the hierarchy.

**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 organization'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.
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 | Answers | 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 Forgotten Passwords

Requesting Forgotten Usernames

Updating your User Profile

Viewing your Group Membership

Viewing your Applications

Crowd User's Glossary

- Alias
- Authorization to Use Crowd
- Crowd Administrator
- Crowd-Connected Application
- Directory
- Group
- Role
- Self-Service Console
- Single Sign-On

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 organization'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 authorized 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

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 authorized to access.

Authorization to Use Crowd

If you are authorized 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**

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**

A 'Crowd-connected application' is a software application which has been designed and configured to use Crowd
for user logins. 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 authorization
purposes, and allow single sign-on across the Crowd domain. The Crowd Administration Guide tells you how to
connect an application to Crowd.

**Directory**

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.

**Group**

A group is a collection of users. Administrators create groups so that the administrator can assign permissions to
a number of people at once. For example, it is quicker to give group 'X' access to JIRA, rather than giving every
team member access individually. In Crowd, each group belongs to a specific directory. It is possible to have two
groups with the same name, such as 'X', in two different directories. A user can be a member of group 'X' in one
directory, in both directories or in neither

**Role**

Support for roles, previously deprecated, was removed in Crowd 2.5.

**Self-Service Console**

Authorized 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 authorized Crowd administrators.

**Single Sign-On**

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.

**RELATED TOPICS**

Logging in to Crowd
Crowd User Guide

**Logging in to Crowd**

If you are authorized 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:

   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. Enter your username and password.

3. Click the 'Log In' button.

   ![Screenshot: Crowd login screen]

If you have forgotten your password or your username, you can click the link labeled 'Can't access your account?'. Read more about resetting your password or requesting a forgotten username.

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 authorized 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 behavior.

• 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 Forgotten Passwords
- 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 behavior.

- 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/
Changing or Resetting your Password

If you are authorized 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 ask to 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 organizations 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 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.

**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:
   - **Current Password** — Your current password.
   - **New Password** — The new password you would like to start using.
   - **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 organizations 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 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 Forgotten Passwords
Requesting Forgotten Usernames
Logging in to Crowd
Crowd User Guide

Resetting Forgotten Passwords

You can go to the Crowd 'Login' screen and request the ability to reset your password. This is useful when you have forgotten the password. Crowd will send you an email message containing a unique, randomly-generated URL. When you click the link on that URL, you will go to a screen where you can choose your own new password.

To reset your password,

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:

   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 appears. Click the link labeled 'Can't access your account?'.

3. The 'Help! I forgot my login details' screen appears. Select the option labeled 'I have forgotten my password'.

4. A panel opens where you can enter your username, as shown below. Enter your Crowd username and click the 'Continue' button.

5. You will receive an email message containing a link to a unique, randomly-generated URL. This link remains available for 24 hours. Click the link in the email message or copy the URL to your browser address bar.

6. The 'Reset Password' screen appears, as shown below. Change your password to one you can remember easily.

   **Screenshot: Forgotten password**

   ![Help! I forgot my login details...](image)
Requesting Forgotten Usernames

You can go to the Crowd 'Login' screen and ask Crowd to email you your username(s). This is useful when you have forgotten your username. Crowd will send a message to the email address you specify, containing all the usernames that are registered for that email address.

To request your username(s),

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:

   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 appears. Click the link labeled 'Can't access your account?'.

3. The 'Help! I forgot my login details' screen appears. Select the option labeled 'I have forgotten my username'.

4. A panel opens where you can enter your email address, as shown below. Enter the email address that you used when you registered with Crowd and click the 'Continue' button.

5. You will receive an email message containing the usernames registered in Crowd for that email address.

6. If you have forgotten your password too, you can now ask to reset your password.
Updating your User Profile

Provided that you are authorized 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 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 organizations 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 authorized 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

What is a Group?

A group is a collection of users. Administrators create groups so that the administrator can assign permissions to a number of people at once. For example, it is quicker to give group 'X' access to JIRA, rather than giving every team member access individually. In Crowd, each group belongs to a specific directory. It is possible to have two groups with the same name, such as 'X', in two different directories. A user can be a member of group 'X' in one directory, in both directories or in neither directory. Two groups called 'X' will be presented to an application as a single group with membership lists aggregated. Groups are particularly important in Crowd, as they are used to control access to applications.

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 organizations 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 Applications
Provided that you are authorized to use Crowd, you can see a list of the applications you are authorized 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 authorized 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 authorized 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.

### Applications

<table>
<thead>
<tr>
<th>Application</th>
<th>Description</th>
<th>Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td>crowd</td>
<td>Crowd Console</td>
<td>(none)</td>
</tr>
<tr>
<td>demo</td>
<td>Crowd Demo Application</td>
<td>(none)</td>
</tr>
<tr>
<td>crowd-openid-server</td>
<td>CrowdID OpenID Provider</td>
<td>(none)</td>
</tr>
<tr>
<td>confluence</td>
<td>Confluence</td>
<td>kbrown</td>
</tr>
</tbody>
</table>

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**
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 authorized to access.

**Authorization to Use Crowd**

If you are authorized 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**

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**

A ‘Crowd-connected application’ is a software application which has been designed and configured to use Crowd for user logins. 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 authorization purposes, and allow single sign-on across the Crowd domain. The Crowd Administration Guide tells you how to connect an application to Crowd.

**Directory**

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.

**Group**

A group is a collection of users. Administrators create groups so that the administrator can assign permissions to a number of people at once. For example, it is quicker to give group 'X' access to JIRA, rather than giving every team member access individually. In Crowd, each group belongs to a specific directory. It is possible to have two groups with the same name, such as 'X', in two different directories. A user can be a member of group 'X' in one directory, in both directories or in neither.

**Role**

Support for roles, previously deprecated, was removed in Crowd 2.5.

**Self-Service Console**

Authorized 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 authorized Crowd administrators.

**Single Sign-On**

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.

**RELATED TOPICS**

Introduction to Crowd
Crowd User Guide
Alias

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 authorized 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

Authorization to Use Crowd

If you are authorized 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

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

A 'Crowd-connected application' is a software application which has been designed and configured to use Crowd for user logins. 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 authorization 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

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

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Group

A group is a collection of users. Administrators create groups so that the administrator can assign permissions to a number of people at once. For example, it is quicker to give group ‘X’ access to JIRA, rather than giving every team member access individually. In Crowd, each group belongs to a specific directory. It is possible to have two groups with the same name, such as ‘X’, in two different directories. A user can be a member of group ‘X’ in one directory, in both directories or in neither directory. Two groups called ‘X’ will be presented to an application as a single group with membership lists aggregated. Groups are particularly important in Crowd, as they are used to control access to applications.

RELATED TOPICS

Specifying which Groups can access an Application
Specifying the Directory Order for an Application
Viewing your Group Membership
Introduction to Crowd
Crowd User Guide

Role

Support for roles, previously deprecated, has been removed in Crowd 2.5. The implementation of roles in Crowd was identical to the implementation of groups and did not provide any extra functionality.

Self-Service Console

Authorized 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 authorized Crowd administrators.

RELATED TOPICS

Introduction to Crowd
Crowd User Guide

Single Sign-On

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

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
  - 1.2 How OpenID sites interact with CrowdID
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 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

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 organizations 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.

To access CrowdID, go to http://localhost:8095/openidserver.
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’
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
- 1.2 How OpenID sites interact with CrowdID
- 1.3 Lightweight OpenID server

Crowd Documentation

1.2 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
- 1.2 How OpenID sites interact with CrowdID
- 1.3 Lightweight OpenID server

Crowd Documentation
1.3 Lightweight OpenID server

Crowd 2.8 introduces a new lightweight UI-free OpenID server, in addition to the existing OpenID server that ships with Crowd.

It uses persistent identifiers unaffected by renaming, and can be accessed at /openidserver/v2/op. It is automatically installed when you install Crowd, and no database setup is necessary.

The OpenID server is a Crowd-connected application which authenticates against the directories configured in Crowd. 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.

You can deploy multiple OpenID servers against a single Crowd instance, which may be useful in larger deployments.

Configuration

The Lightweight OpenID server has no admin UI. You control the server using its approval whitelist configuration file. By default, the whitelist is empty so no authentication will succeed until you add URLs.

For communication with Crowd, see the crowd.properties file located by default in crowd-openidserver-webapp/WEB-INF/classes. You may also set the system property crowd.openid.home to point to another directory holding this configuration file.

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 server¹.
2. View your CrowdID application² 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 server¹.
2. Map the directory to your CrowdID application² 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 server¹.
2. Map the group to your CrowdID application² 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

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 server¹.
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

<table>
<thead>
<tr>
<th>View User – jane</th>
</tr>
</thead>
<tbody>
<tr>
<td>Details</td>
</tr>
</tbody>
</table>

These are the groups the user is a member of.

<table>
<thead>
<tr>
<th>Group</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>crowd-administrators</td>
<td>Add &gt;</td>
</tr>
</tbody>
</table>

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 log in

There are three ways to specify which OpenID hosts (i.e. websites or IP addresses) your users can log in to using their CrowdID:

- **No restriction** — your CrowdID users can log in to any OpenID host
- **Blacklist** — your CrowdID users can log in to any OpenID host except the one(s) you specify
- **Whitelist** — your CrowdID users can log in to only those OpenID host(s) that you specify

In addition, you may configure an Approval Whitelist for trusted sites,

3.1 Allowing all hosts

There are three ways to specify which OpenID hosts (i.e. websites or IP addresses) your users can log in to using their CrowdID:

- **No restriction** — your CrowdID users can log in to any OpenID host
- **Blacklist** — your CrowdID users can log in to any OpenID host except the one(s) you specify
- **Whitelist** — your CrowdID users can log in 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'
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 log in to using their CrowdID:

- **No restriction** — your CrowdID users can log in to any OpenID host
- **Blacklist** — your CrowdID users can log in to any OpenID host except the one(s) that you specify
- **Whitelist** — your CrowdID users can log in 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'_
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 log in to using their CrowdID:

- **No restriction** — your CrowdID users can log in to any OpenID host
- **Blacklist** — your CrowdID users can log in to any OpenID host except the one(s) that you specify
- **Whitelist** — your CrowdID users can log in 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'*
3.4 Approval Whitelist

For trusted sites, such as internal services, you may wish to simplify the user experience by automatically approving authentication requests. Users will not be prompted to verify authentication requests from these realms.

The OpenID Verification page (2.4 Allowing or denying a login) shows the realm that a host is using. This configuration is stored in a file:

```
crowd-openidserver-webapp/WEB-INF/classes/crowdid.approval-whitelist
```

Each line is a single OpenID realm. If an authentication request is received from a site on that list it will automatically be approved as if the user had selected ‘Allow Always’.

Example

In the default configuration, using the demo OpenID client to authenticate, the OpenID server will present an OpenID verification page:

```
The following site: http://localhost:8095/openidclient
has requested that you confirm the following address as your personal identity
```

Adding:

http://localhost:8095/openidclient

to the approval whitelist would automatically approve the demo OpenID client for all users.

**RELATED TOPICS**

- 3.1 Allowing all hosts
- 3.2 Allowing all except specified hosts ('Blacklist')
- 3.3 Allowing specified hosts only ('Whitelist')
- 3.4 Approval 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
- DRAFT - 4. Configuring CrowdID system settings

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’**
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'*
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'
General Configuration

Where is the server installed?

Base URL: https://openid.siteadmin.com/
This is the base URL of this installation of the Crowd OpenID server. All links created for display, email etc will be prefixed by this URL: //openid.siteadmin.com/

What configuration options would you like?

- Allow localhost authentications: Enables authentications to be redirected back to localhost.
- Allow immediate authentication requests: Allows sites to request immediate authentication responses, preventing user interaction (such as logging in). Immediate mode will only successfully authenticate the site if the user is logged in and has always allowed authentication to the site.
- Allow stateless clients: Allows sites to request authentication without establishing a pre-shared secret (association) with the server. This will enable less secure communication to take place between external sites and this server.

Update | Cancel

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
- DRAFT - 4. Configuring CrowdID system settings

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
DRAFT - 4. Configuring CrowdID system settings

CrowdID system settings can be specified either as:

- arguments supplied to the Java VM when starting CrowdID from the command-line, or as
- properties specified in CrowdID's configuration files, in which case you must specify the location of those files in with the crowd.openid.home JVM system property (see below).

CrowdID will use the default classpath configurations if a user configuration is not supplied.

Specifying system settings on the command-line

System settings for CrowdID can be given on the command-line, when they need to be prefixed with "-D", and be placed before the "-jar" parameter.

For example:

```
java -Dcrowd.server.url=http://localhost:1234/openidserver
-Dapplication.base.url=http://localhost:1234/openidserver
-Dapplication.login.url=http://localhost:8095/openidserver
-Dapplication.login.url.template=http://localhost:1234/openidserver?returnToUrl=${RETURN_TO_URL} -jar crowdid.jar
```

Specifying system settings in a configuration file

System settings for CrowdID can be specified in configuration files, which can be in an arbitrary user-specified location. In this case, you must specify the location of the directory containing those files using the crowd.openid.home argument supplied to the JVM

For example:
The CrowdID configuration files use the standard Java properties files format.

### CrowdID configuration files

The defaults for these files can be found in `<CrowdID install directory>/WEB-INF/classes/`.

<table>
<thead>
<tr>
<th>Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crowd.properties</td>
<td>CrowdID's core settings.</td>
</tr>
<tr>
<td>crowdid.approval-whitelist</td>
<td>See 3.4 Approval Whitelist.</td>
</tr>
<tr>
<td>jdbc.properties</td>
<td>JDBC settings, which will depend on your database.</td>
</tr>
</tbody>
</table>

If the value of crowd.openid.home is set, these files will be read from that location.

For an explanation of the specific settings, see Integrating Crowd with Atlassian CrowdID.

See also:

#### 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 organization. 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
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 organization.

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 log in to any OpenID-enabled website. OpenID allows the website to communicate with your OpenID provider (e.g. your organization'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 organization.

This means that you can:

- Securely store your username and password on your organization'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 organization can use CrowdID to set up an internal OpenID provider. There are also other OpenID providers, where you can get a free OpenID.
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 organization's CrowdID server. Here are some examples of what your OpenID may look like:

```
http://my.server.name/mynname
http://mynname.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.

**Endpoint URLs**

You can also use CrowdID's **endpoint URL** to log in to OpenID-enabled websites. The OpenID URLs for specific users may look like:

```
http://my.server.name/openidserver/users/mynname
http://my.server.name/openidserver/users/anothername
```

Crowd also provides an endpoint URL that you can pass to login sites to have your identifier automatically selected:

```
http://my.server.name/openidserver/op
```

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 organization.
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 labeled '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 canceled'.

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 labeled ‘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 dialog 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 canceled.

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 canceled.

Screenshot: CrowdID Approved Sites page
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
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 organization’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
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

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 organization'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*
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
- 6.3 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*
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
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. This is useful when you have forgotten the password. Crowd will send you an email message containing a unique, randomly-generated URL. When you click the link on that URL, you will go to a screen where you can choose your own new password.

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
8. Requesting Forgotten Usernames

7.1 Changing your password

The CrowdID 'Change Your Password' page allows you to change your password across all applications in your organization, 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. The '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
- 8. Requesting Forgotten Usernames
7.2 Resetting your password

The CrowdID 'Login' page allows you to reset your password. This is useful when you have forgotten the password. Crowd will send you an email message containing a unique, randomly-generated URL. When you click the link on that URL, you will go to a screen where you can choose your own new password.

To reset your password,

1. Access CrowdID.
2. The CrowdID login page will appear. Click the link labeled 'Can't access your account?'.
3. The 'Help! I forgot my login details' screen appears. Select the option labeled 'I have forgotten my password'.
4. A panel opens where you can enter your username. Enter your Crowd username and click the 'Continue' button.
5. You will receive an email message containing a link to a unique, randomly-generated URL. This link remains available for 24 hours. Click the link in the email message or copy the URL to your browser address bar.
6. The 'Reset Password' screen appears. Change your password to one you can remember easily.

The password reset email will contain a link based on the crowd.server.url from the crowd.properties file. If the public address is different you may set crowd.base.url.

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
- 8. Requesting Forgotten Usernames

8. Requesting Forgotten Usernames

You can go to the CrowdID 'Login' screen and ask CrowdID to email you your username(s). This is useful when you have forgotten your username. CrowdID will send a message to the email address you specify, containing all the usernames that are registered for that email address.

To request your username(s),

1. Access CrowdID.
2. The CrowdID login page appears. Click the link labeled 'Can't access your account?'.
3. The 'Help! I forgot my login details' screen appears. Select the option labeled 'I have forgotten my username'.
4. A panel opens where you can enter your email address. Enter the email address that you used when you registered with CrowdID and click the 'Continue' button.
5. You will receive an email message containing the usernames registered in CrowdID for that email address.
6. If you have forgotten your password too, you can now ask to reset your password.

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
Crowd FAQ

Known issues, hints and tips and answers to commonly asked questions about Crowd:

Concepts:

- What is single sign-on (SSO)?
- What is authorization?
- What is authentication?
- What is centralized 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?

Common Evaluator Questions:

Unable to render {children}. Page not found: CONFEVAL:Frequently Asked Questions For Crowd.

Crowd Resources

Deployment FAQ

- Deploying Multiple Atlassian Applications in a Single Tomcat Container
- Finding the atlassian-crowd.log File
- Finding your Crowd Home Directory
- Recovering your Console application password
- Removing the 'crowd' Context from the Application URL
- Resetting the Domain Cookie Value
- Restarting the Setup Wizard from Scratch
- Self Signed Certificate
- Using Crowd in a Cluster is Not Supported

Guides, Hints and Tips

- How to Print Only Tomcat Logs into Crowd's catalina.out
- Principals and Users
- Using Apache Directory Studio for 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
What is single sign-on?

Single sign-on enables users to authenticate (login) once and gain access to multiple web applications within a single domain. See also centralized authentication.

What is authorization?

Authorization is the act of deciding whether a person is allowed to access a specific resource or web application. This often comes in the form of groups, roles and permissions.

What is authentication?

Authentication is the act of verifying that a user is who they say they are. This is often done through a credential such as user name and password.

What is centralized authentication?

Centralized authentication is when an end-user has the same username and password used across all web applications, even if the application cannot participate in single sign-on. This is often a major milestone before single sign-on is achieved within an organization.

Crowd provides centralization authentication and/or single sign-on depending on your application's capabilities.

What is identity management?

Identity management is the process of defining a user (a 'principal') and managing their attributes. In addition to
username and credentials (e.g. password), attributes might include phone number, address, etc.

Identity management also includes assigning users to relevant groups and roles, so that users can access appropriate applications and resources.

Another important part of identity management is managing the entire user lifecycle, for example, disabling the user account when someone leaves the organization.

**What is a directory?**

A directory is a repository of information containing user identities, their attributes and their group and role memberships.

**How does Crowd work? How is Crowd an "application security framework"?**

Crowd is made up of two parts:

- **Administration console**: a brilliantly simple and powerful web interface that manages directories, users, and their security rights.
- **Integration API**: a single security architecture where multiple web applications are integrated. With the integration API, applications can quickly access user information or perform security checks.

**What is an application connector?**

An **application connector** is the link between Crowd and one of your applications. An application connector makes it possible to connect, say, Crowd and JIRA. When you download and install Crowd, you’ll automatically get its application connectors, along with an integration API so that you can code your own application connectors too.

**What is a directory connector?**

A **directory connector** is the link between Crowd and one of your directories. It makes it possible to connect, say, Crowd and Active Directory. When you download and install Crowd, you’ll automatically get its directory connectors, along with an integration API so that you can code your own directory connectors too.

**How many users can Crowd manage?**

Crowd can support over 500 users depending upon which license you purchase. View the licensing and pricing breakdown for more information. We have customers using Crowd successfully with tens of thousands of users.

**How many applications can be used with Crowd?**

So long as they’re compatible with Crowd, you can add in as many applications as your organization needs.

**We already have an LDAP server for Confluence and/or JIRA. Do we really need Crowd?**

If one or more of the following apply, Crowd will be of benefit to you:

- your organization uses multiple applications and they have not yet been integrated into the LDAP server
- you are looking for an easy way to manage all your JIRA and Confluence users in one database with one or more directory servers
- your organization has not yet implemented single sign-on
- you are looking for a way to help save you and your organization time, frustration, and much more!

**What are Crowd’s system requirements?**

For information on compatible databases, application servers, and operating systems, read the Supported Platforms page.
What directories and applications does Crowd support out-of-the-box?

A complete list of currently supported applications and directories can be found in Crowd’s documentation. Check back often, as new connectors will be added regularly.

How can Crowd be connected to new or currently unsupported applications?

Crowd provides a simple and intuitive integration API (backed by REST or SOAP) that allows you to connect in your new or existing applications. This makes it easy to choose how much or how little to integrate based on your needs.

How does Crowd integrate with other Atlassian products?

Crowd ships with connectors for Atlassian products.

Using the out-of-the-box connectors you can consolidate all of your users into a single repository giving you the ability to manage all of your users in a single location. Users can then take advantage of single sign-on, giving them one username and password to access all of your applications.

Does Crowd include kerberos integration?

Crowd does not currently support kerberos-based authentication. For licensing and pricing — please see the Purchasing FAQ.

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 and Announcements

- Crowd Announcements
- Answers from the community

Mailing Lists

- Visit http://my.atlassian.com to sign up for mailing lists relating to Atlassian products, such as technical alerts, product announcements and developer updates.

Feature Requests

- Issue Tracker and Feature Requests for Crowd
Deployment FAQ

- Deploying Multiple Atlassian Applications in a Single Tomcat Container
- Finding the atlassian-crowd.log File
- Finding your Crowd Home Directory
- Recovering your Console application password
- Removing the 'crowd' Context from the Application URL
- Resetting the Domain Cookie Value
- Restarting the Setup Wizard from Scratch
- Self Signed Certificate
- Using Crowd in a Cluster is Not Supported

Deploying Multiple Atlassian Applications in a Single Tomcat Container

Deploying multiple Atlassian applications in a single Tomcat container is not supported. We do not test this configuration and upgrading any of the applications (even for point releases) is likely to break it. There are also a number of known issues with this configuration:

- You may not be able to start up all of the applications in the container, due to class conflicts (in 3rd party libraries bundled with our application) that result from the Atlassian applications sharing a single JVM in the Tomcat container.
- You will not be able to determine the startup order of the applications. Hence, you may experience problems such as JIRA starting before Crowd, rather than vice versa.
- Memory problems are also common as one application may allocate all of the memory in the Tomcat JVM to itself, starving the other applications.

We also do not support deploying multiple Atlassian applications to a single Tomcat container for a number of practical reasons. Firstly, you must shut down Tomcat to upgrade any application and secondly, if one application crashes, the other applications running in that Tomcat container will be inaccessible.

Finally, we recommend not deploying any other applications to the same Tomcat container that runs the Atlassian application, especially if these other applications have large memory requirements or require additional libraries in Tomcat's lib subdirectory.

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.

<table>
<thead>
<tr>
<th>Installation Type</th>
<th>Location of Log File</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowd Standalone edition</td>
<td><strong>Crowd 2.0.3 and older versions</strong>: In the root directory of your Crowd application, e.g. atlassian-crowd-2.0.0/atlassian-crowd.log</td>
</tr>
<tr>
<td></td>
<td><strong>Crowd 2.0.4 and newer versions</strong>: In the Crowd application Home Directory, e.g. Crowd-Home-Directory/logs/atlassian-crowd.log</td>
</tr>
<tr>
<td>Crowd Standalone running as a Windows service</td>
<td>C:\Windows\system32\atlassian-crowd.log</td>
</tr>
<tr>
<td>Crowd WAR edition</td>
<td>The directory from which you start the application server, e.g. apache-tomcat-6.0.16/bin/atlassian-crowd.log</td>
</tr>
</tbody>
</table>

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. (Note however that the CrowdID database will be in the installation directory, not the Home 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 authorization 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:

```java
Caused by: com.atlassian.crowd.integration.exception.InvalidAuthenticationException
  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
```
  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)
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 application_name = 'jira';
    +------------------------------------------------------------------
    | credential
    +------------------------------------------------------------------
    | sQnzu7wkTrgkQZF+0G1hi5AI3Qmzvv0bXgc5THBqi7mAsdd4Xl127ASbrT9fEyavW\n    m0QP9B81Thf+rDKy8hg== |
    +------------------------------------------------------------------
   ```

3. Now set Crowd's application credentials to the credential of your application X:

   ```
   mysql> update cwd_application set credential = 'sQnzu7wkTrgkQZF+0G1hi5AI3Qmzvv0bXgc5THBqi7mAsdd4Xl127ASbrT9fEyavWim0QP9B81Thf+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.

6. If recovering the application password didn't work, you can also change it for another one as the following:
a. In the `<Crowd-Home>/crowd.properties` file, change the application.password entry to "To7CfmPz";
b. In the database, run the following command:

```sql
UPDATE cwd_application
SET credential = '{PKCS5S2}+ebYspXujIGikRoLVenN/BpbtwEg+WfR1u0k1BjPfG4nJwbeAr0sv5U+YG7x5k'
WHERE application_name = 'crowd';
```

c. Restart 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 hashed.

**Removing the 'crowd' Context from the Application URL**

For many different reasons, when using the Crowd distribution (not EAR-WAR), you may want to access the Crowd console using `http://localhost:8095` instead of `http://localhost:8095/crowd`. In order to remove the `/crowd` part from the URL, you can take the following steps:

**IMPORTANT:** Before doing these changes in your production environment, please make sure that they will work in a test instance first.

1. Edit file `<Crowd-Install>/build.properties` and make sure that variable `crowd.url` is set to the following:

```
# Crowd context root

crowd.url=http://localhost:8095/
```

2. Run `<Crowd-Install>/build.sh` (UNIX) or `<Crowd-Install>/build.bat` (Windows).
3. In your `<Crowd-Home-Directory>/crowd.properties` file, make sure that the `crowd.server.url` and `application.login.url` URLs do not contain the `/crowd` part.

```
crowd.server.url=http://localhost:8095/services/
application.login.url=http://localhost:8095/
```

4. Change your `<Crowd-Install>/apache-tomcat/conf/server.xml` file to have the following Host section configuration:
5. Rename the `crowd.xml` file in `<Crowd-Install>/apache-tomcat/conf/Catalina/localhost` to `crowdbackup.xml` to prevent Tomcat from loading the context.

6. Run Crowd and access `http://localhost:8095`. You will be automatically redirected to the Crowd server console page.

### 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:

```sql
update cwd_property set property_value = '' where property_name = 'domain';
```

Once you have done this you will need to restart Crowd, clear your browser's cache 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.
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: [https://docs.oracle.c](https://docs.oracle.c)
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.

Guides, Hints and Tips

- How to Print Only Tomcat Logs into Crowd's catalina.out
- Principals and Users
- Using Apache Directory Studio for LDAP Configuration

How to Print Only Tomcat Logs into Crowd's catalina.out

Crowd writes its logs into atlassian-crowd.log and Tomcat logs as well. However, this might only be noticed for Crowd installed in *nix based server, as in Windows these are printed in the console. Below is the start-up process written in catalina.out, where it describe the start-up process of all applications bundled inside Crowd standalone, such as Crowd OpenID and Crowd itself:

<table>
<thead>
<tr>
<th>DateTime</th>
<th>Module</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 31, 2012</td>
<td>org.apache.catalina.startup.Catalina</td>
<td>load</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>org.apache.catalina.realm.JAASRealm</td>
<td>setContainer</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>org.apache.catalina.core.StandardRealm</td>
<td>start</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>org.apache.catalina.core.StandardEngine</td>
<td>start</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>org.apache.catalina.startup.HostConfig</td>
<td>deployDescriptor</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>System Information</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>Timezone: Malaysia Time</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>Java Version: 1.6.0_31</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>JVM Version: 20.6-b01</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>JVM Runtime: Java HotSpot(TM) 64-Bit Server VM</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>Username: sultan</td>
</tr>
<tr>
<td>May 31, 2012</td>
<td>com.atlassian.crowd.startup</td>
<td>Architecture: amd64</td>
</tr>
</tbody>
</table>
Encoding: UTF-8

2012-05-31 18:20:12,342 main INFO [com.atlassian.crowd.startup] Used Memory: 34MB
2012-05-31 18:20:12,342 main INFO [com.atlassian.crowd.startup] Runtime Information:
2012-05-31 18:20:12,342 main INFO [com.atlassian.crowd.startup] Version: 2.4.2
2012-05-31 18:20:12,342 main INFO [crowd.console.listener.StartupListener] Upgrades not performed since the application has not been setup yet.

INFO: Deploying configuration descriptor openidserver.xml
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in
[jar:file:/home/sultan/atlassian/CROWD/atlassian-crowd-2.4.2/crowd-openidserver-webapp/WEB-INF/lib/slf4j-jcl-1.0.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
### Crowd Start-up

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012-05-31 21:17:17</td>
<td>main INFO</td>
<td>[com.atlassian.crowd.startup] System Information:</td>
</tr>
<tr>
<td>2012-05-31 21:17:17</td>
<td>main INFO</td>
<td>[com.atlassian.crowd.startup] Runtime Information:</td>
</tr>
</tbody>
</table>
This might be a redundant information as they've been written into atlassian-crowd.log.

In order to get Crowd print only Tomcat logs into catalina.out, please modify these files:

- \(<\text{Crowd}_{-}\text{Install}\_\text{Directory}>/\text{crowd}_{-}\text{-webapp}/\text{WEB}_{-}\text{-INF}/\text{classes}/\text{log4j}_{-}\text{.properties}\)
- \(<\text{Crowd}_{-}\text{Install}\_\text{Directory}>/\text{crowd}_{-}\text{-openidserver}_{-}\text{-webapp}/\text{WEB}_{-}\text{-INF}/\text{classes}/\text{log4j}_{-}\text{.properties}\)
- \(<\text{Crowd}_{-}\text{Install}\_\text{Directory}>/\text{crowd}_{-}\text{-openidclient}_{-}\text{-webapp}/\text{WEB}_{-}\text{-INF}/\text{classes}/\text{log4j}_{-}\text{.properties}\)
- \(<\text{Crowd}_{-}\text{Install}\_\text{Directory}>/\text{demo}_{-}\text{-webapp}/\text{WEB}_{-}\text{-INF}/\text{classes}/\text{log4j}_{-}\text{.properties}\)

modify this line:

<table>
<thead>
<tr>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>log4j.rootLogger=INFO, console, crowdlog</td>
</tr>
</tbody>
</table>

into:

<table>
<thead>
<tr>
<th>Modify it to</th>
</tr>
</thead>
<tbody>
<tr>
<td>log4j.rootLogger=INFO, crowdlog</td>
</tr>
</tbody>
</table>

Save all of the modified log4j.properties and restart Crowd.

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 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 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*
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:

<table>
<thead>
<tr>
<th>Hostname</th>
<th>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.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>For normal LDAP connectivity, use 389. For SSL connectivity, use 636.</td>
</tr>
</tbody>
</table>

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**
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**
While you should already know the user DN (Distinguished Name) 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.
- See Creating a Connection to your LDAP Directory for details of how to connect Apache Directory Studio to your LDAP directory.
- 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.
- If you are using Nested Groups in Crowd, your group filter must include all sub-groups to pick up the sub-group members.

**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

![User information in Apache Directory Studio](image)

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.

<table>
<thead>
<tr>
<th>User DN:</th>
<th><code>cn=Users</code></th>
</tr>
</thead>
</table>
| User Object Filter: | `{(objectCategory=Person)(sAMAccountName=*)}
{(|memberOf=cn=confluence-users,ou=Groups,dc=sydney,dc=atlassian,|(memberOf=cn=confluence-administrators,ou=Groups,dc=sydney,dc=atlassian,_dc=atlassian,dc=com))}` |

Screenshot: The resulting user configuration in Crowd

![The resulting user configuration in Crowd](image)
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.

<table>
<thead>
<tr>
<th>Group DN:</th>
<th><code>ou=Groups</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Object Filter:</td>
<td>`(&amp;(objectCategory=Group) (</td>
</tr>
</tbody>
</table>

Screenshot: The resulting group/role configuration in Crowd
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:
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:

```
java.lang.VerifyError:
  (class: org/codehaus/xfire/aegis/type/basic/ObjectType, method: writeSchema signature: (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
Security Advisory Publishing Policy
Security Bugfix 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

- Our Support team will help with workarounds and bug reporting
- We'll generally fix critical bugs in the next maintenance release
- We schedule non-critical bugs according to a variety of considerations

Report a bug

Building an add-on

Are you developing an add-on for an Atlassian product or using one of our APIs? Report any related bugs here.

Bug reports

Atlassian Support is eager and happy to help verify bugs—we take pride in it! Create an issue in our support system, providing as much information as you can about how to replicate the problem you're experiencing. We'll replicate the bug to verify, then lodge the report for you. We'll also try to construct workarounds if possible.

Search existing bug reports

Use our issue tracker to search for existing bugs, and watch the ones that are important to you. When you watch an issue, we'll send you an e-mail notification when the issue's updated.

How we approach 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 (the 1 in 6.0.1, for example).

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) we'll fix it in the next maintenance release, provided that:

- The fix is technically feasible (it doesn't require a major architectural change)
- It doesn't impact the quality or integrity of a product

For non-critical bugs, the developer assigned to fixing bugs prioritises the 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

Developers responsible for fixing bugs also monitor comments on existing and new bugs, so you can comment to provide feedback if you need to. We give high priority to security issues.

When considering the priority of a non-critical bug, we try to determine a value score for a bug. The score takes into account the severity of the bug from our customers' perspective, how prevalent the bug is, and whether new features on our roadmap may render the bug obsolete. Our developers combine the value score with a complexity score (how difficult the bug is) when selecting issues to work on.

Further reading

See Atlassian Support Offerings for more support-related information.
How to Report a Security Issue

We've moved!
Go to Atlassian.com/security for the latest information.

Further reading
See Atlassian Support Offerings for more support-related information.

New Features Policy

Summary

- We encourage and display customer comments and votes openly in our issue tracking system, http://jira.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 this process.
- Atlassian provides consistent updates on the top 20 issues.

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:

- **Customer contact**: We get the chance to meet customers and hear their successes and challenges at Atlassian Summit, Atlassian Unite, developer conferences, and road shows.
- **Customer interviews**: All product managers at Atlassian do customer interviews. Our interviews are not simply to capture a list of features, but to understand our customers’ goals and plans.
- **Community forums**: There are large volumes of posts on answers, of votes and comments on jira.atlassian.com, and of conversations on community forums like groups on LinkedIn.
- **Customer Support**: Our support team provides clear insights into the issues that are challenging for customers, and which are generating the most calls to support.
- **Atlassian Experts**: Our Experts provide insights into real-world customer deployments, especially for customers at scale.
- **Evaluator Feedback**: When someone new tries our products, we want to know what they liked and disliked and often reach out to them for more detail.
- **In product feedback**: The JIRA Issue Collectors that we embed our products for evaluators and our Early Access Program give us a constant pulse on how users are experiencing our product.
- **Usage data**: Are customers using the features we have developed?
- **Product strategy**: Our long-term strategic vision for the product.
- Please read our post on Atlassian Answers for a more detailed explanation.

How to contribute to feature development

Influencing Atlassian’s release cycle

We encourage our customers to vote on issues that have been raised in our public JIRA instance, http://jira.atlassian.com. Please find out if your request already exists - if it does, vote for it. If you do not find it you may wish to create a new one.

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 Atlassian Answers.
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 Atlassian Support Offerings for more support-related information.

Security Advisory Publishing Policy

See https://www.atlassian.com/trust/security for more information.

Security Bugfix Policy

See Security @ Atlassian for more information on our security bugfix policy.

Security Patch Policy

We've moved!

Go to Atlassian.com/security for the latest information.

Further reading

See Atlassian Support Offerings for more support-related information.

Severity Levels for Security Issues

We've moved!

Go to Atlassian.com/security for the latest information

Further reading

See Atlassian Support Offerings 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 LDAP User Management
- Troubleshooting SSL certificates and Crowd
- How to Optimize Crowd Client Caching
- Troubleshooting Crowd Performance
- Troubleshooting SSO with Crowd
  - Debugging SSO in environments with Proxy Servers
- Troubleshooting CrowdID
- Provide Crowd Information to Atlassian Support
- 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

As of Crowd 2.5.1, CWD-2042 - Forward slashes not escaped correctly in DNs etc. is resolved. This
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

There is another ticket tracking problems with certain characters, eg '/':

CWD-2042 - Forward slashes not escaped correctly in DNs etc.

RESOLVED

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

<table>
<thead>
<tr>
<th>Decimal ASCII value</th>
<th>AbstractEncodingFilter Replacement Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>183</td>
<td>&quot;- &quot;</td>
<td>Middle dot, Georgian comma, Greek middle dot</td>
</tr>
<tr>
<td>8211</td>
<td>&quot;-&quot;</td>
<td>En dash</td>
</tr>
<tr>
<td>8216</td>
<td>&quot;&quot;</td>
<td>Left single quotation mark</td>
</tr>
<tr>
<td>8217</td>
<td>&quot;&quot;</td>
<td>Right single quotation mark</td>
</tr>
<tr>
<td>8220</td>
<td>&quot;&quot;</td>
<td>Left double quotation mark</td>
</tr>
<tr>
<td>8221</td>
<td>&quot;&quot;</td>
<td>Right double quotation mark</td>
</tr>
<tr>
<td>8230</td>
<td>&quot;...&quot;</td>
<td>Horizontal ellipsis, three dot leader</td>
</tr>
</tbody>
</table>

ISO-8859-1

<table>
<thead>
<tr>
<th>Decimal ASCII value</th>
<th>AbstractEncodingFilter Replacement Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>133</td>
<td>&quot;...&quot;</td>
<td>Horizontal ellipsis, three dot leader</td>
</tr>
<tr>
<td>145</td>
<td>&quot;&quot;</td>
<td>Left single quotation mark</td>
</tr>
<tr>
<td>146</td>
<td>&quot;&quot;</td>
<td>Right single quotation mark</td>
</tr>
<tr>
<td>147</td>
<td>&quot;&quot;</td>
<td>Left double quotation mark</td>
</tr>
<tr>
<td>148</td>
<td>&quot;&quot;</td>
<td>Right double quotation mark</td>
</tr>
<tr>
<td>150</td>
<td>&quot;-&quot;</td>
<td>En dash</td>
</tr>
</tbody>
</table>

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*

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>525</td>
<td>user not found</td>
</tr>
<tr>
<td>52e</td>
<td>invalid credentials</td>
</tr>
<tr>
<td>530</td>
<td>not permitted to logon at this time</td>
</tr>
<tr>
<td>531</td>
<td>not permitted to logon at this workstation</td>
</tr>
<tr>
<td>532</td>
<td>password expired</td>
</tr>
<tr>
<td>533</td>
<td>account disabled</td>
</tr>
<tr>
<td>701</td>
<td>account expired</td>
</tr>
<tr>
<td>773</td>
<td>user must reset password</td>
</tr>
<tr>
<td>775</td>
<td>user account locked</td>
</tr>
</tbody>
</table>

*This information provided by the following IBM support document.

To enable LDAP logging on your AD server, please review this Microsoft guide.

**Troubleshooting LDAP User Management**

**Scope**

This page describes troubleshooting LDAP user management configurations and setup in JIRA Software. Note other specific documents:

<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Troubleshooting User Management Upgrade Issues</td>
<td>Describes user management issues encountered during an upgrade to JIRA 4.3.x or later.</td>
</tr>
<tr>
<td>Connecting to an LDAP Directory</td>
<td>Atlassian's primary documentation on LDAP Directory configuration.</td>
</tr>
</tbody>
</table>

**About Apache Directory Studio**

Apache Directory Studio is an open source project of the Apache Software Foundation. It includes an LDAP browser/editor, a schema browser, an LDIF editor, a DSML editor and more. It is a highly useful tool for troubleshooting integration problems with JIRA Software and Confluence. It is an Eclipse RCP...
application that is cross-platform or run within Eclipse itself as a plugin. For more information on Eclipse, please refer to the Wikipedia article on Eclipse Software.

We recommend using it for testing as a method of isolating where the problem with an integration exists. It allows you to test if an application other than JIRA Software or Confluence can connect to the LDAP/AD server. If a successful connection cannot be established it can indicate problems on the side of the directory, if it can connect it can indicate problems with the configuration of JIRA Software or Confluence.

### Installation and Set Up

It is recommended to install Directory Studio on a computer other than the server that hosts the LDAP/AD server (a local workstation is usually best). Directory Studio is able to connect to the directory using the LDAP protocol and display the contents of the directory using its built-in browser. To establish the connection, please refer to the connection section in the Directory Studio user guide.

If Directory Studio is unable to initiate a connection it can indicate there is a problem with the LDAP/AD server connection. Refer to the troubleshooting section below.

### Browsing the Directory

Using the Browser functionality in Directory Studio you can navigate through the Directory tree to see where users and groups are stored. It will also allow you to identify how data is stored within the directory, including the names of attributes and specific values.

This is a comparison between the Active Directory and Apache Directory Studio. Below is Active Directory:
And this is how it would look in Directory Studio:

As can be seen from the above example, Directory Studio is very useful for identifying which attributes to use when setting the Schema Settings and User Schema Settings as described in Connecting to an LDAP Directory as it shows the names and values of the attributes in a clear, readable format.

**Directory Attributes**

LDAP uses a schema (DIT) to define how it stores data. These are much like a database - an attribute holds a value just like a field within a table.

- Expand to see a list of common directory attributes

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
Common attributes:

<table>
<thead>
<tr>
<th>Attribute Description</th>
<th>Attribute Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIT</td>
<td>Directory Information Tree</td>
</tr>
<tr>
<td>DC</td>
<td>Domain Controller</td>
</tr>
<tr>
<td>DN</td>
<td>Distinguished Name</td>
</tr>
<tr>
<td>CN</td>
<td>Common Name</td>
</tr>
<tr>
<td>SN</td>
<td>Surname</td>
</tr>
<tr>
<td>OU</td>
<td>Organizational Unit</td>
</tr>
<tr>
<td>UID</td>
<td>Unique ID</td>
</tr>
</tbody>
</table>

Example of an LDAP/AD tree for the Administrator user:

- **Domain Controller**: ad01,crowd,atlassian,com
- **Common Name**: Users
- **Common Name**: Administrator

**Distinguished Name**: CN=Administrator,CN=Users,DC=ad01,DC=crowd,DC=atlassian,DC=com

**Go to DN**

The Go to DN is essentially a search option that allows you to find an LDAP element easily. It can be accessed through the context menu in the LDAP Browser (right click). It is useful for quickly finding Distinguished Names that would relate to configuration settings, such as Base DN, Additional User DN or Additional Group DN.

Expand to see an example
Troubleshooting

Prior to troubleshooting, please ensure that you have verified the following credentials of the LDAP/AD server with your System Administrator. If they are not correct, you will not be able to successfully connect or bind to the LDAP/AD server.

- Server Hostname (or IP).
- Server Port.
- Bind Username (DN).
- Bind Password.

A successful connection requires the following two steps:

1. **Server Connectivity**: the application is able to communicate with the LDAP/AD server.
   - Uses Hostname and Port.
2. **LDAP/AD Bind**: the application is able to log into the directory.
   - Uses Bind Username & Password.

**Identify Active Directory DNs**

If using Active Directory, the `dsquery` command can be executed on the command-line to identify DNs to particular objects, as per this [Dsquery Windows Server](#) article. Please refer to the examples below for further information.

Expand to see an example of how to locate the full DNs
Access the command line and execute the following to identify the DN for a user:

```
C:\>dsquery user -name Administrator
"CN=Administrator,CN=Users,DC=sydney,DC=atlassian,DC=com"
```

In the above example, this breaks down to:

- **Domain Controller**: sydney.atlassian.com
- **Common Name**: Users
- **Common Name**: Administrator

The following can then be used to identify the DN for any groups:

```
C:\>dsquery group -name jira-administrators
"CN=jira-administrators,OU=Groups,DC=sydney,DC=atlassian,DC=com"
```

In the above example, this breaks down to:

- **Domain Controller**: sydney.atlassian.com
- **Organizational Unit**: Groups
- **Common Name**: jira-administrators

Using the the above results, when entering the configuration for an Active Directory Connector in JIRA Software, the following DNs would be used:

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>DC=sydney,DC=atlassian,DC=com</td>
</tr>
<tr>
<td>Additional User DN</td>
<td>CN=Users</td>
</tr>
<tr>
<td>Additional Group DN</td>
<td>OU=Groups</td>
</tr>
</tbody>
</table>

The above settings would synchronize all users in `CN=Users,DC=sydney,DC=atlassian,DC=com` and all groups in `OU=Groups,DC=sydney,DC=atlassian,DC=com`.

**Testing the LDAP/AD Server Connectivity**

If any of the below tests fail it indicates there is most likely a problem with the LDAP/AD Server or the port/IP is not correct. Troubleshooting those problems is outside of the scope of this document.

Testing the LDAP/AD Bind is done through Directory Studio.

**Windows**

Expand to see Windows instructions

Test that the server is reachable by pinging it.

```
C:\>ping -n 1 ldap-host
Pinging ldap-host.example.com [192.168.1.100] with 32 bytes of data:
Reply from 192.168.1.100: bytes=32 time=4ms TTL=127
Ping statistics for 172.20.4.167:
   Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
   Approximately round trip times in milli-seconds:
   Minimum = 4ms, Maximum = 4ms, Average = 4ms
```

Check that the port is open (typically 389). A successful connection to the port indicates it is open. If a
A successful connection is made, the screen will go blank. If not, the below message will be generated. To exit telnet, use CTRL+C.

C:\>telnet ldap-host 389
Connecting To ldap-host...Could not open connection to the host, on port 389: Connect failed

**Linux**

**Expand to see Linux instructions**

Test that the server is reachable by pinging it.

```
$ ping -c1 ldap-host
PING ldap-host (192.168.1.100): 56 data bytes
64 bytes from 192.168.1.100: icmp_seq=0 ttl=127 time=0.521 ms
--- crowd-ad1 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.521/0.521/0.521/0.000 ms
```

Check that the port is open (typically 389). A successful connection to the port indicates it is open. To exit telnet, use CTRL+C.

```
$ telnet ldap-host 389
Trying 192.168.1.100...
Connected to crowd-ad1.
Escape character is '^]'.
```

**Common LDAP Errors**

**Expand to see an example error from Directory Studio**

These are a list of errors returned from Directory Studio and Embedded Crowd that can help identify configuration issues with the LDAP/AD server.

<table>
<thead>
<tr>
<th>Error</th>
<th>Possible Cause(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>simple bind failed: ldap-host: 389</td>
<td>This is a very general error, and it means something went wrong when trying to bind to LDAP/AD. Check to see if the LDAP/AD server name and/or port number you have specified is incorrect or an incorrect DN was specified as the administrator username. It could also indicate the Encryption method was specified incorrectly in the Network Parameter tab of the connector properties in Directory Studio.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Connection refused (ldap-host:389) No connection</td>
<td>The port number specified for the LDAP/AD server is incorrect or a firewall/network configuration is blocking the connection.</td>
</tr>
<tr>
<td>Unknown Host: ldap-host No connection</td>
<td>The LDAP/AD server name/IP specified is incorrect.</td>
</tr>
<tr>
<td>LDAP: error code 2 InvalidSearchFilterException</td>
<td>Invalid search filter passed to the LDAP/AD server or the filter is malformed (e.g.: it’s missing a open and closing of parenthesis). Please refer to this LDAP Query Basics Microsoft TechNet article for further information on LDAP/AD queries.</td>
</tr>
<tr>
<td>LDAP: error code 3</td>
<td>The application is timing out when waiting on a response from LDAP. Check the LDAP filters are correct, the LDAP engine is working correctly. The default timeout can be increased in the configuration.</td>
</tr>
<tr>
<td>LDAP: error code 4</td>
<td>The Active Directory/LDAP Server that JIRA Software is synchronizing with has a limitation on the number of objects it can return with one query. Enable 'Use Paged Results'.</td>
</tr>
<tr>
<td>LDAP: error code 34</td>
<td>The DN is most likely incorrect - confirm this by checking it against Directory Studio or using dsquery if using Active Directory.</td>
</tr>
<tr>
<td>LDAP: error code 48</td>
<td>The Authentication Method in the Authentication tab of the connector properties is incorrect.</td>
</tr>
<tr>
<td>LDAP: error code 32</td>
<td>LDAP_NO_SUCH_OBJECT - Indicates the target object cannot be found. This code is not returned on following operations: Search operations that find the search base but cannot find any entries that match the search filter. Bind operations.</td>
</tr>
</tbody>
</table>
| LDAP: error code 49 | The Bind Username (DN) or Password is incorrect. This could also be caused by the following:  
  - Pointed to non-user DN.  
  - Pointed to a non-existent DN.  
  - Pointed to an incorrect admin DN (might be using UID instead of CN).  
  - Pointed to a non administrator user.  
  - Pointed to a valid administrator user, however the password is incorrect.  
A DN describes a path to the object in the database, so the CN may be correct however the OUs and DCs in the DN specified may be incorrect or in the wrong order. |
| NoSuchAttributeException | This can be caused by providing a name for an attribute that is not correct or does not exist. |
| InvalidNameException | This can be caused by a bad prefix specified in the Settings tab on most LDAP/AD systems.  
This could mean you did not specify a prefix at all, which means the LDAP/AD server did not receive a full DN from CPS or that you did not specify a correct prefix, such as CN instead of UID, which results in the LDAP/AD server not receiving a correct DN from CPS.  
Can also be caused by a missing comma at the beginning of the suffix or an extra comma at the end of the suffix. This error could also mean the authentication type is incorrect. |
AuthenticationException

Could not authenticate the user trying to login. This can be the result of an incorrect username or password or an incorrect prefix and/or suffix specified in the Settings tab, depending on the type of LDAP/AD system. Could also mean the authentication type is incorrect.

Common Configuration Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Possible Cause(s)</th>
<th>Possible Fix</th>
</tr>
</thead>
<tbody>
<tr>
<td>• JIRA Software runs out of memory, experiences slow performance or high CPU load during User Directory sync. • Users are imported, however groups are not. • Groups are imported, however users are not.</td>
<td>The Schema Settings may be incorrect causing JIRA Software to attempt to import unnecessary or incorrect data from the LDAP/AD server.</td>
<td>Verify by browsing with Directory Studio (using 'Go to DN' is recommended) and ensure the following settings are correct within the Schema Settings in JIRA Software: Base DN Additional User DN Additional Group DN</td>
</tr>
<tr>
<td>User details (e.g.: Username, Email, Full Name, Password) are not correct after a successful sync.</td>
<td>The User Schema Settings may be incorrect or the value within the attributes is incorrect. • If this applies to only one user, it is likely the user details are incorrect. • If it applies to multiple, it is likely the JIRA Software configuration is incorrect.</td>
<td>Verify by browsing with Directory Studio (using 'Go to DN' is recommended) and ensure the following attributes are correct within the User Schema Settings in JIRA Software: User Name User First Name User Last Name User Display Name User Email</td>
</tr>
<tr>
<td>Users and Groups are imported, however Users aren't automatically added to the Groups they belong to.</td>
<td>The Membership Schema Settings may be incorrect. • If this applies to only one user, it is likely the user details are incorrect. • If it applies to multiple, it is likely the JIRA Software configuration is incorrect.</td>
<td>Verify by browsing with Directory Studio (check with a sample user) and ensure the following attributes are correct within the Membership Schema Settings in JIRA Software: Group Members Attribute User Membership Attribute</td>
</tr>
<tr>
<td>Unable to add/delete Users or make changes to their Groups, Project Roles or Edit them.</td>
<td>The LDAP/AD User Directory is set to Read Only.</td>
<td>Verify the Permissions Settings are correct for the external User Directory. If you want to make changes, Read/Write will need to be activated. If using Active Directory, please refer to this page on Configuring an SSL Certificate.</td>
</tr>
</tbody>
</table>

Providing Additional Information

Often when the above troubleshooting steps don't resolve the problem and in particularly complicated scenarios, further information will need to be provided so Atlassian can investigate further.

Generating 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 LDAP Configuration

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 Crowd distribution's (not EAR-WAR) 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.

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 Optimize 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!

Created in 2017 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
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.
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
- Authorization 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. 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.
2. Set each application to use centralized 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:
<authenticator
class="com.atlassian.jira.security.login.JiraSeraphAuthenticator"/>

you should have this:

<authenticator
class="com.atlassian.jira.security.login.SSOSeraphAuthenticator"/>

Please, see our Adding an Application Tutorial page to check the SSO authenticator classes for other applications.

3. If you are using a reverse proxy in front of any of the applications, you'll need to make sure that the host header is preserved in the forward. For example, in an Apache reverse proxy, you need to enable the "ProxyPreserveHost" option, and in IIS you need to use "Application Request Routing" to achieve the same.

4. Once each application is using centralized authentication, confirm you can log in to each application with the same username and password.

5. Inside of Crowd, ensure that each application is configured to use the same user directory. SSO will not work if you log in to Confluence through one user directory, but JIRA through a different user directory, even if the usernames are identical.

6. 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.

7. Check if the "Require Consistent Client IP Address" is enabled in the Session Configuration. If it is, try disabling it and test SSO.

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 packag e to DEBUG.
2. Replicate the SSO problem you are having.
3. Please raise a support issue on our Support System, attaching your {CROWD HOME}/logs/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 consisting of an IP address, User-Agent, Random Number.

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 192.168.3.125 which is really the proxy's. This results in a token that doesn't match my existing one: as compared to onk7YDa9kfyPp0ipf26gwA00 and thus, I'm prompted to login again. A clue that I was going through the proxy is the 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.
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.

   1. Change the openid package from INFO to DEBUG in CROWD/crowd-openidserver-webapp/WEB-INF/classes/log4j.properties

   # Set the following lines to DEBUG to enable logging on incoming, outgoing and fault SOAP messages
   
   log4j.logger.com.atlassian.crowd.integration.service.soap.xfire.XfireInLoggingMethodHandler=DEBUG
   log4j.logger.com.atlassian.crowd.integration.service.soap.xfire.XfireOutLoggingMethodHandler=DEBUG
   log4j.logger.com.atlassian.crowd.integration.service.soap.xfire.XfireFaultLoggingMethodHandler=DEBUG

   log4j.logger.com.atlassian.crowd.openid=DEBUG
   log4j.logger.com.atlassian.crowd=INFO

   2. Change the openid package from INFO to DEBUG in CROWD/crowd-openidclient-webapp/WEB-INF/classes/log4j.properties

   log4j.logger.com.atlassian.crowd.openid=DEBUG
   log4j.logger.com.atlassian.crowd=INFO

Step 2: Test CrowdID with the bundled OpenID client:
Documentation for Crowd 2.11

• 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.

Provide Crowd Information to Atlassian Support

Purpose

Sending us your log files and configuration details will speed up resolution of your issue.

Solution

Please provide these artifacts when you create a ticket:

• A zip archive of <crowd_home>/logs/
• A zip archive of <crowd_install>/apache-tomcat/logs/
• All the text from Crowd Admin > System Information > Support Information.

The <crowd_home> directory is the path defined in <crowd_install>/crowd-webapp/WEB-INF/classes/crowd-init.properties.

The text can be attached in a file or pasted into the description or comment of your ticket at https://support.atlassian.com.

If the zip files are less than 125MB you can attach them to your ticket. For larger files please request a transfer account and your support agent create one for you.

Contributing to the Crowd Documentation

Would you like to share your Crowd hints, tips and techniques with us and with other Crowd users? We welcome your contributions.

On this page:

• Blogging your Technical Tips and Guides – Tips of the Trade
• Contributing Documentation in Other Languages
• Updating the Documentation Itself
  • Getting Permission to Update the Documentation
  • Our Style Guide
  • How we Manage Community Updates

Blogging your Technical Tips and Guides – Tips of the Trade

Have you written a blog post describing a specific configuration of Crowd or a neat trick that you have discovered? Let us know, and we will link to your blog from our documentation. More....

Contributing Documentation in Other Languages

Have you written a guide to Crowd in a language other than English, or translated one of our guides? Let us know, and we will link to your guide from our documentation. More....

Updating the Documentation Itself

Have you found a mistake in the documentation, or do you have a small addition that would be so easy to add
yourself rather than asking us to do it? You can update the documentation page directly.

Getting Permission to Update the Documentation

Please submit the Atlassian Contributor License Agreement.

Our Style Guide

Please read our short guidelines for authors.

How we Manage Community Updates

Here is a quick guide to how we manage community contributions to our documentation and the copyright that applies to the documentation:

- **Monitoring by technical writers.** The Atlassian technical writers monitor the updates to the documentation spaces, using RSS feeds and watching the spaces. If someone makes an update that needs some attention from us, we will make the necessary changes.

- **Wiki permissions.** We use wiki permissions to determine who can edit the documentation spaces. We ask people to sign the Atlassian Contributor License Agreement (ACLA) and submit it to us. That allows us to verify that the applicant is a real person. Then we give them permission to update the documentation.

- **Copyright.** The Atlassian documentation is published under a Creative Commons CC BY license. Specifically, we use a Creative Commons Attribution 2.5 Australia License. This means that anyone can copy, distribute and adapt our documentation provided they acknowledge the source of the documentation. The CC BY license is shown in the footer of every page, so that anyone who contributes to our documentation knows that their contribution falls under the same copyright.

RELATED TOPICS

Tips of the Trade
Author Guidelines
Atlassian Contributor License Agreement

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 them on your production site.

On this page:
- `omniauth_crowd`
- `Integrating Enterprise Tester with Crowd – A .Net Integration`
- `Three's a Crowd - securing a Grails application with Acegi and Crowd`
- `SSO for RoundCube Webmail with Atlassian Crowd`
- `Nexus Crowd Plugin Introduction`
- `Integrating Crowd with Subversion`
- `Install Crowd Apache2 Module`
- `Video – Configuring a Delegated Authentication Directory`
### Application Connectors

#### omniauth_crowd

- **By:** Rob Di Marco, on GitHub
- **About:** Integrating Crowd with OmniAuth to allow Rack-based applications (such as Rails, Sinatra) to use Crowd as an authentication source
- **Date and Crowd version:** March 2011, Crowd 2.2

#### Integrating Enterprise Tester with Crowd – A .Net Integration

- **By:** Catch Limited, on the 'Atlassian Blog'
- **About:** Integrating Enterprise Tester, a .Net application, with Crowd. Enterprise Tester is a test management solution from Catch Limited.
- **Date and Crowd version:** 8 July 2010
- **Related documentation:** Microsoft .NET Client

#### 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

#### Integrating Crowd with Subversion

- **By:** Trisummit Technologies
- **About:** Integrating Crowd with Subversion and Apache
- **Date and Crowd version:** 4 April 2010
- **Related documentation:** Integrating Crowd with Subversion

### Directory Connectors

#### Video – Configuring a Delegated Authentication Directory

- **By:** Brendan Patterson, on Screencast.com
- **About:** Configuring a Delegated Authentication directory and an LDAP directory connector
- **Date and Crowd version:** August 2010; Crowd 2.0
- **Related documentation:** Configuring a Delegated Authentication Directory

#### Video – Configuring a Custom Directory Connector

- **By:** Brendan Patterson, on Screencast.com
- **About:** Creating a custom directory connector
- **Date and Crowd version:** August 2010; Crowd 2.0
- **Related documentation:** Configuring a Custom Directory Connector
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
Answers from the community
Atlassian Blog
Crowd plugins

Crowd Documentation in Other Languages

Below are some links to Crowd documentation written in other languages. In some cases, the documentation may be a translation of the English documentation. In other cases, the documentation is an alternative guide written from scratch in another language. This page presents an opportunity for customers and community authors to share documentation that they have written in other languages.

Please be aware that these are external guides.
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. The information in the linked guides has not been tested or reviewed by Atlassian.

On this page:
• No guides yet

<table>
<thead>
<tr>
<th>None</th>
</tr>
</thead>
</table>

No guides yet

We do not yet have any guides to link here. Be the first to suggest one!

Adding Your Own Guide to this Page

Have you written a guide for Crowd in another language? Add a comment to this page, linking to your guide. We will include it if the content fits the requirements of this page.

Giving Feedback about One of the Guides

If you have feedback on one of the guides listed above, please give the feedback to the author of the linked guide.

If you want to let us know how useful (or otherwise) one of these guides is, please add a comment to this page.

Other Sources of Information

Crowd documentation
Atlassian website
Atlassian blog
Crowd plugins and connectors