Documentation for Stash 1.2
# Contents

Getting started .................................................. 4
  Supported platforms ........................................ 4
  Installing Stash on Windows ............................... 5
    Running Stash as a Windows service .................. 9
  Installing Stash on Linux and Mac ...................... 11
  Getting started with Git and Stash .................... 15
  Configuring JIRA integration in the Setup Wizard ... 21

Using Stash .................................................... 27
  Creating projects and managing permissions for a project 28
  Creating repositories .................................... 30
  Basic Git commands ...................................... 31
  Permanently authenticating with Git repositories ...... 33
  Setting up SSH ............................................. 35
    Setting up SSH on Windows ........................... 36
    Setting up SSH on Linux and Mac .................. 39

Administering Stash .......................................... 42
  Users and groups ......................................... 42
  Global permissions ...................................... 49
  JIRA integration ......................................... 50
    JIRA compatibility .................................... 52
  External user directories ............................... 54
    Configuring an LDAP directory ....................... 56
    Configuring delegated LDAP authentication ........ 63
    Connecting to Crowd .................................. 69
    Connecting to JIRA for user management ........... 70
  Setting up your mail server ............................ 73
  Specifying the base URL for Stash .................... 74
  Connecting Stash to an external database ............ 74
    Connecting Stash to MySQL .......................... 76
    Connecting Stash to Oracle ......................... 79
    Connecting Stash to PostgreSQL ..................... 81
    Connecting Stash to SQL Server .................... 84

Advanced actions .............................................. 87
  Data recovery and backups ............................. 87
  Enabling Stash debug logging ......................... 88
  Integrating Stash with Apache HTTP Server ........... 88
  Scaling Stash ............................................ 96
  System settings ......................................... 100
  Stash config properties ................................ 100
  Securing Stash with HTTPS ............................ 101
  Enabling SSH access to Git repositories in Stash .... 107
    Setting up SSH port forwarding ..................... 110

Releases ...................................................... 114
  Stash upgrade guide ................................... 114
  Stash 1.2 release notes ................................ 116
    Stash 1.2 change log ................................ 121
  Stash 1.1 release notes ................................ 121
    Stash 1.1 change log ................................ 125
  Stash 1.0 release notes ................................ 126
    Stash 1.0 change log ................................ 130
    Stash 1.0.1 release notes ......................... 132
  Stash 1.0 upgrade guide ................................ 134
  Stash security advisories .............................. 135
  Git resources ............................................ 135
  Stash FAQ ................................................ 136
  How do I change the external database password ..... 138
Installation troubleshooting guide ........................................ 139
Stash home directory ......................................................... 139
Support policies ................................................................. 140
  Bug fixing policy ............................................................ 140
  How to report a security issue ............................................ 141
  New features policy ......................................................... 142
  Patch policy ................................................................. 143
  Security advisory publishing policy ................................... 143
  Security patch policy ....................................................... 144
  Severity levels for security issues ...................................... 145
Building Stash from source .................................................. 146
Getting started
This section describes how to install, set up and get started with Atlassian Stash.

In this section:
- Supported platforms
- Installing Stash on Windows
  - Running Stash as a Windows service
- Installing Stash on Linux and Mac
- Getting started with Git and Stash
- Configuring JIRA integration in the Setup Wizard

Related pages:
- Administering Stash
- Stash FAQ

Supported platforms
This page lists the supported platforms for Stash 1.2.x.

Key: ✔️ = Supported; ❌ = Not Supported

<table>
<thead>
<tr>
<th>Java Version</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle JDK (1)</td>
<td>✔️ 1.7</td>
</tr>
<tr>
<td></td>
<td>✔️ 1.6</td>
</tr>
<tr>
<td></td>
<td>❌ 1.5</td>
</tr>
</tbody>
</table>

Operating Systems

<table>
<thead>
<tr>
<th>Operating Systems</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Windows (2)</td>
<td>✔️</td>
</tr>
<tr>
<td>Linux (2)</td>
<td>✔️</td>
</tr>
<tr>
<td>Apple Mac OS X (2)</td>
<td>✔️</td>
</tr>
</tbody>
</table>

Databases (4)

<table>
<thead>
<tr>
<th>Databases</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PostgreSQL</td>
<td>✔️ 8.2, 8.3, 8.4, 9.1</td>
</tr>
<tr>
<td>Oracle</td>
<td>✔️ 11g</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>✔️ 2008 R2</td>
</tr>
<tr>
<td>MySQL</td>
<td>✔️ MySQL Enterprise Server 5.1.x or later ✔️ MySQL Community Server 5.1.x or later</td>
</tr>
<tr>
<td>HSQLDB</td>
<td>✔️ (bundled; for evaluation use only)</td>
</tr>
</tbody>
</table>

Web Browsers

Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
### Notes:

1. **Oracle JDK:**
   - For the server, please ensure that you have the Java JDK. It is not enough to have just the JRE. JDKs other than the Oracle JDK are not supported.
   - You can download the Java SE Development Kit (JDK) from the [Oracle website](http://oracle.com).
   - Once the JDK is installed, you will need to set the environment variable, pointing to the root JAVA_HOME directory of the JDK. Some JDK installers set this automatically (check by typing `echo %JAVA_HOME%` in a command prompt, or `echo $JAVA_HOME` in a shell). You should do this before installing Stash.

2. Please note:
   - Stash is a pure Java application and should run on any platform, provided all the JDK requirements are satisfied.
   - If you are using **Linux/UNIX**: A dedicated user should be created to run Stash, as Stash runs as the user it is invoked under and therefore can potentially be abused. Here is an example of how to create a dedicated user to run Stash in Linux:
     ```
     $ sudo /usr/sbin/useradd --create-home --home-dir /usr/local/Stash --shell /bin/bash Stash
     ```

3. 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.

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

4. Please see our documentation on [connecting Stash to an external database](http://www.atlassian.com).  

5. The version of Git installed on machines that interact with Stash must be compatible with the version of Git installed for use by the Stash server.

### Installing Stash on Windows

**Hey! We're going to install Stash on Windows. There are a few steps, but we think you'll really like Stash once it's up and running.**

#### 1. Check supported platforms

Better check the [Supported platforms](http://www.atlassian.com) page first; it lists the application servers, databases, operating systems, web browsers and JDKs that we have tested Stash with and recommend.
Atlassian only officially supports Stash running on x86 hardware and 64-bit derivatives of x86 hardware.

### Related pages:
- Running Stash as a Windows service
- Installing Stash on Linux and Mac
- Getting started with Git and Stash
- Supported platforms

2. Check your version of Java

In a command prompt, run this:

```
java -version
```

The version of Java should be 1.6.0 or higher.

**If you don’t see Java 1.6.0 or higher, then get Java...**

Download and install the Java Platform JDK (not the JRE) from [Oracle’s website](https://www.oracle.com/).  

*The Java install path must not contain spaces, so don’t install into `C:\Program Files\Java`.*  

Instead, use a path like `C:\Java`.  

Now try running `java -version` again to check the installation. The version of Java should be 1.6.0 or higher.

3. Check that Windows can find Java

Windows uses the `JAVA_HOME` environment variable to find Java. To check that, in a command prompt, run:

```
echo %JAVA_HOME%
```

You should see a path to the Java install location that does not contain spaces.

**If you don’t see a path without spaces...**
• If you see a path with spaces, like this `C:\Program Files\Java\`, then sorry, but go back to 2. and reinstall Java to a location that doesn't have spaces.

• If you don’t see a path at all, or if you just see `%JAVA_HOME%`, then set JAVA_HOME as follows:

**For Windows 7:**

Go to **Start**, search for “sys env” and choose **Edit the system environment variables**.

Click **Environment Variables**, and then **New** under ‘System variables’.

Enter "JAVA_HOME" as the **Variable name**, and the absolute path to where you installed Java as the **Variable value**. Don’t use a trailing backslash.

Now, in a new command prompt, try running `%JAVA_HOME%\bin\java -version`. You should see the same version of Java as you saw in 2. above.

4. **Check your version of Git**

In a command prompt, run:

```
  git --version
```

The version of Git should be **1.7.6** or higher.

*If you don’t see version 1.7.6 or higher, then get Git...*
Download the Full installer for official Git for Windows.

Run the Git installer., ensure that git.exe is available in the path.

- You must select the option which puts git.exe and the other UNIX tools on the path, as shown below.
- Do not select the option to only add Git to the path -- this will not work with Stash.

Now, in a new command prompt, try running 'git --version' again. The version of Git should be 1.7.6 or higher.

5. Now it's time to get Stash

Download Stash from the Atlassian download site.

Extract the downloaded file to an install location. The path to the extracted directory is referred to as the <Stash installation directory> in these instructions.

6. Tell Stash where to store your data

The Stash home directory is where your Stash data is stored.

You should not locate your Stash home directory inside the <Stash installation directory> — they should be entirely separate locations. If you do put the home directory in the <Stash installation directory> it will be overwritten, and lost, when Stash gets upgraded. And, by the way, you can't use the same Stash home directory for multiple instances of Stash.

Create your Stash home directory, and then tell Stash where you created it by setting a STASH_HOME environment variable, as follows.

For Windows 7:
1. Go to Start, search for "sys env" and choose Edit the system environment variables.
2. Click **Environment Variables**, and then **New** under ‘System variables’.
3. Enter "STASH_HOME" as the **Variable name**, and the absolute path to your Stash home directory as the **Variable value**. Don't use a trailing backslash.

### 7. Start Stash!

In a command prompt, change directory to the `<Stash installation directory>` and run:

```
bin\start-stash.bat
```

In your browser, go to [http://localhost:7990](http://localhost:7990) and run through the Setup Wizard. In the Setup Wizard:

- Select **Internal** at the 'Database' step, if you are evaluating Stash. Stash will happily use its internal database, and you can easily migrate to external database later. See [Connecting Stash to an external database](#).
- You can set up JIRA integration, but you can also do this later if you wish. See [Configuring JIRA integration in the Setup Wizard](#).

### 8. Set up your mail server

Configure your email server so users can receive a link from Stash that lets them generate their own passwords. See [Setting up your mail server](#).

### 9. Additional steps for production environments

For production environments we recommend that you configure the additional aspects below. These are not necessary when installing for evaluation purposes.

**Use an external database**

For production environments Stash should use an external database, rather than the embedded database. See [Connecting Stash to an external database](#).

**Secure the Stash home directory**

For production environments the Stash home directory (created in step 7 above) should be secured against unauthorised access. See [Stash home directory](#).

**Secure Stash with HTTPS**

For production environments access to Stash should be secured using HTTP over SSL, especially if your data is sensitive and Stash is exposed to the internet. See [Securing Stash with HTTPS](#).

**Run Stash as a Windows service**

See [Running Stash as a Windows service](#).

### 10. Stop Stash (optional)

In a command prompt, change directory to the `<Stash installation directory>` and run:

```
bin\stop-stash.bat
```
Running Stash as a Windows service

For long-term use on a Windows server, Stash should be configured to run as a Windows service. This has the following advantages:

- Stash will be automatically restarted when the operating system restarts.
- Stash is less likely to be accidentally shut down, as can happen if the console window Stash was manually started in is closed.
- Stash logs are properly managed by the Windows service.

Before you start

- If you are running a 64-bit version of Windows, note that Apache Tomcat cannot run as a Windows service with a 64-bit JDK. Please ensure that you are using a 32-bit JDK.
- On any Windows operating system with User Account Control (UAC) such as Windows Vista or Windows 7, simply logging in to Windows with an Administrator account will not be sufficient to execute the script in the procedure below. You must either disable UAC or run ‘cmd.exe’ as an administrator (e.g. by right-clicking on ‘cmd.exe’ and choosing Run as administrator).
- Ensure the JAVA_HOME variable is set to the root of your Java platform’s installation directory.
  
  Note: Your JAVA_HOME cannot contain spaces, so the default Java installation directory of C:\Program Files\Java won’t work.

- When you run Stash as a Windows service, all settings in setenv.bat are ignored. Ensure that you have set STASH_HOME as a system environment variable.

Setting up Stash as a Windows service

To run Stash as a Windows service:

1. Stop Stash.
2. Open a Command Prompt (as an Administrator, see the Before you start section above).
3. Change directory to the Stash installation directory and then into the bin subdirectory. If a directory in the path has spaces (e.g. C:\Program Files\..), use its eight-character equivalent (e.g. C:\Progra-1\..).
4. Run the following commands:

   > service.bat install
   > tomcat6 //US///STASH --Startup auto

   This will create a service with the name “STASH” and a display name of “Atlassian Stash”. If you would
like to customize the name you can instead run:

```
> service.bat install MyName
> tomcat6 //US//MyName --Startup auto
```

This will create the service as "MyName" with a display name of "Atlassian Stash MyName".

5. Run the following command to increase the amount of memory that Stash can use (the default is 256 Mb):

```
> tomcat6 //US//service_name --JvmMx 1024
```

6. Verify that the Stash service comes back up after restarting the machine.

Here is an example:

```
C:\Program Files (x86)\atlassian-stash-1.0.0\bin>service.bat install
Installing the service 'STASH' ...
Using CATALINA_HOME:    "C:\Program Files (x86)\atlassian-stash-1.0.0"
Using CATALINA_BASE:    "C:\Program Files (x86)\atlassian-stash-1.0.0"
Using JAVA_HOME:        "C:\Java\jre6"
Using JVM:              "auto"
The service 'STASH' has been installed.

C:\Program Files (x86)\atlassian-stash-1.0.0\bin>tomcat6.exe //US//STASH
---Startup auto

C:\Program Files (x86)\atlassian-stash-1.0.0\bin>tomcat6.exe //US//STASH --JvmMx 1024

C:\Program Files (x86)\atlassian-stash-1.0.0\bin>net start STASH
The Atlassian Stash service is starting.
The Atlassian Stash service was started successfully.
```

Troubleshooting

- Problems may occur when trying to setup Stash to run as a Windows service with JDK 1.6. The problem is due to failure to locate MSVCR71.DLL, which can be found in %JAVA_HOME%/bin. There are two options to resolve this problem:
  - Add %JAVA_HOME%/bin to PATH, then restart the Stash server.
  - Copy MSVCR71.DLL to system path, C:\WINDOWS\SYSTEM32 or C:\WINNT\SYSTEM32.
- Take note of the username that the service is running as, and be sure to modify the /temp and /work directories in your install directory so that this user has read and write permissions.
- You cannot run Stash as a service on a 64-bit operating system if you require allocating more than 1.5GB of memory, due to 32-bit JDK memory limitations and 64-bit JDK/Tomcat service issues.

Installing Stash on Linux and Mac

Hey! We're going to install Stash on a Linux box, or a Mac. There are a few steps, but we think you'll really like Stash once it's up and running.
1. Check supported platforms

Better check the Supported platforms page first; it lists the application servers, databases, operating systems, web browsers and JDKs that we have tested Stash with and recommend.

Atlassian only officially supports Stash running on x86 hardware and 64-bit derivatives of x86 hardware.

Related pages:
- Installing Stash on Windows
- Getting started with Git and Stash
- Supported platforms

2. Check your version of Java

In a terminal, run this:

```
java -version
```

The version of Java should be 1.6.0 or higher.

**If you don't see Java 1.6.0 or higher, then get Java...**

Download and install the Java Platform JDK (not the JRE) from Oracle's website.

Now try running 'java -version' again to check the installation. The version of Java should be 1.6.0 or higher.

3. Check that the system can find Java

In a terminal, run this:

```
echo $JAVA_HOME
```

You should see a path like /System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/.

**If you don't see a path then set JAVA_HOME...**

If you don't know what your Java home directory is, try running the command `which java` in your terminal.
### Linux

Do either of the following:

- If `JAVA_HOME` is not set, log in with ‘root’ level permissions and run:

  ```
  echo
  JAVA_HOME="path/to/JAVA_HOME"
  >> /etc/environment
  ```

  where `path/to/JAVA_HOME` may be like: `/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/`

- If `JAVA_HOME` needs to be changed, open the `/etc/environment` file in a text editor and modify the value for `JAVA_HOME` to:

  ```
  JAVA_HOME="path/to/JAVA_HOME"
  ```

  It should look like:

  ```
  JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/
  ```

### Mac

Open your `~/.profile` file in a text editor and insert:

- `JAVA_HOME="path/to/JAVA_HOME"
- export JAVA_HOME`

  where `path/to/JAVA_HOME` may be like: `/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/`

- Refresh your `~/.profile` in the terminal and confirm that `JAVA_HOME` is set:

  ```
  source ~/.profile
  $JAVA_HOME/bin/java -version
  ```

You should see a version of Java that is 1.6.0 or higher, like this:

```
java version "1.6.0_24"
```

---

### 4. Check your version of Git

In a terminal, run this:

```
  git --version
```

The version of Git should be 1.7.6 or higher.

- **If you don't see version 1.7.6 or higher, then get Git...**
Download and install the latest stable Git release from the [Git website](https://git-scm.com/download).  

Now try running 'git --version' again. The version of Git should be **1.7.6** or higher.

Please note the following:

- See the [Git resources](https://git-scm.com/resources) page for links to more Git download sites.
- At the time of writing, the default Git version on Ubuntu Linux is 1.7.5.x, which is too old for Stash: see [https://launchpad.net/~git-core/+archive/ppa](https://launchpad.net/~git-core/+archive/ppa).
- At the time of writing, on Mac OS X, the Git tar archive can fail on special characters when using **SSH** to secure connections between your computer and Stash. (The Git archive command allows you to download as a single file the files in a checkout of the Git repository.) We recommend that you use the zip format; you can set that using the following command:

  ```shell
  git archive --format zip
  ```

5. **Now it’s time to get Stash**

Download Stash from the Atlassian download site.  

Extract the downloaded file to an install location. The path to the extracted directory is referred to as the `<Stash installation directory>` in these instructions.

6. **Tell Stash where to store your data**

The Stash **home directory** is where your Stash data is stored.  

⚠️ **You should not** locate your Stash home directory inside the `<Stash installation directory>` — they should be entirely separate locations. If you do put the home directory in the `<Stash installation directory>` it will be overwritten, and lost, when Stash gets upgraded. And by the way, you'll need separate Stash home directories if you want to run multiple instances of Stash.

Create your Stash home directory, and then tell Stash where you created it by editing the `<Stash installation directory>/bin/setenv.sh` file — uncomment the `STASH_HOME` line and add the absolute path to your home directory. Here's an example of what that could look like when you're done:

```bash
# One way to set the STASH_HOME path is here via this variable. Simply uncomment it and set a valid path
# /stash/home. You can of course set it outside in the command terminal; that will also work.
STASH_HOME="/Users/spittet/stash-home"

# Occasionally Atlassian Support may recommend that you set some specific JVM arguments. You can use this below to do that.
```

7. **Start Stash!**

In a terminal, change directory to `<Stash installation directory>` and run this:

```
bin/start-stash.sh
```
In your browser, go to http://localhost:7990 and run through the Setup Wizard. In the Setup Wizard:

- Select Internal at the 'Database' step, if you are evaluating Stash. Stash will happily use its internal database, and you can easily migrate to external database later. See Connecting Stash to an external database.
- You can set up JIRA integration, but you can do this later if you wish. See Configuring JIRA integration in the Setup Wizard.

8. Set up your mail server

Configure your email server so users can receive a link from Stash that lets them generate their own passwords. See Setting up your mail server.

9. Additional steps for production environments

For production environments we recommend that you configure the additional aspects below. These are not necessary when installing for evaluation purposes.

Use an external database

For production environments Stash should use an external database, rather than the embedded database. See Connecting Stash to an external database.

Secure the Stash home directory

For production environments the Stash home directory (created in step 7 above) should be secured against unauthorised access. See Stash home directory.

Secure Stash with HTTPS

For production environments access to Stash should be secured using HTTP over SSL, especially if your data is sensitive and Stash is exposed to the internet. See Securing Stash with HTTPS.

10. Stop Stash (optional)

In a terminal, change directory to <Stash installation directory> and run this:

```
bin/stop-stash.sh
```

Getting started with Git and Stash

Atlassian Stash is the repository management solution that allows everyone in your organisation to easily collaborate on all your Git repositories.

This page will guide you through the basics of Stash. By the end you should know how to:

- Create accounts for your collaborators, and organize these into groups with permissions.
- Create a project and set up permissions.
- Create repositories, and know the basic commands to interact with them.

Assumptions

This guide assumes that you don't have prior experience with Git. But we do assume that:

- You have Git version 1.7.6 or higher installed on your local computer.
- You are using a supported browser.
Install Stash

Get Stash running on your computer. See the details here:

- [Installing Stash on Linux and Mac](#)
- [Installing Stash on Windows](#)

On this page:

- Assumptions
- Install Stash
- Add users to Stash and grant permissions
- Create your first project and share it with collaborators
  - Creating your project
  - Opening up project access to others
- Create a repository and get your code into Stash
  - Create a repository
  - A simple clone and push

Related pages:

- [Installing Stash on Windows](#)
- [Installing Stash on Linux and Mac](#)
- [Supported platforms](#)

Watch the movie »

Add users to Stash and grant permissions

The first thing you can do in Stash is to add collaborators.

Go to the Users listing in the Administration area.

Click Create User to go directly to the user creation form.
Create user

There are 24 users remaining on your 500 user license

Username* jdoe

Full name* John Doe

Email address* john@doe.com

√ Email a link to the user to set their password

Create User Cancel

Once you've created a user, click Change permissions to set up their access permissions.

There are 4 levels of user permissions:

- **System Administrator** — can access all the configuration settings of the Stash instance.
• **Administrator** — same as System Admins, but they can’t modify file paths or the Stash server settings.
• **Project Creator** — can create, modify and delete projects.
• **Stash User** — active users who can access Stash.

### Create your first project and share it with collaborators

#### Creating your project

The next thing you do in Stash is to create a project. You’ll add repositories to this project later.

Simply click **Create Project** in the Projects page.

![Create Project](image)

Complete the form and submit it to create your new project.

**Create a Project**

- **Name**: Angry Nerds Mobile
- **Project Key**: ANM
- **Description**: Project for the mobile version of Angry Nerds

[Create Project] [Cancel]
Opening up project access to others

If you are a project administrator, you can grant project permissions to other collaborators.

Click the Permissions tab for the project.

On that page you can add users and groups to a project you've already created.

There are 3 levels of project access:

- **Project Administrator** — can create, edit and delete repositories and projects, and configure permissions for projects.
- **Contributor** — can push to and pull from all the repositories in the project.
- **Observer** — can only browse code and comments in, and pull from, the repositories in the project.

Create a repository and get your code into Stash

Create a repository
If you are a project administrator, you can create repositories in the project.

Once a repository is created, the project permissions are applied to the repository. That means all repositories created in a project share the same access and permission settings.

Click **Create Repository** to open the repository creation form.

Once submitted you will be taken directly to your repository homepage. As there is no content in your repository yet, you'll see some instructions to help you push code to your repository.

**A simple clone and push**

Here you will simply clone the repository you just created and then push a commit back to it. You can see the clone URL to use at the top right of the screen, **SSH access** may be available.

In a terminal, run the following command (replace `<stashURL>` with the URL for your instance of Stash):

```bash
    git clone <stashURL>/git/<projectname>/<reponame>.git
```

Use your Stash username and password.

The result in your terminal should be similar to what you can see in the screenshot below.

You should now have a new empty directory tracked by Git, in the user space of your local machine. Let's add some content and push it back to Stash.

In your `<reponame>` directory, create a text file named `helloworld.txt` and write "Hello World" in it.

Now run the following command in your terminal

```bash
    cd <reponame>
    git add .
    git commit -a -m 'My first commit'
    git push origin master
```
If everything went fine, when you refresh the Stash screen, you will see that the homepage of your repository has been replaced with a file browser showing you a link to helloworld.txt.

There you go, you’re ready to get coding with your collaborators.

Check out our Basic Git commands for more information regarding the basic Git commands that you will probably use often.

**Configuring JIRA integration in the Setup Wizard**

This page describes the ‘JIRA integration’ screen of the Stash setup wizard.

You can connect your application to a JIRA server, to manage your users via JIRA and share information with JIRA. When you are installing the application, the setup wizard gives you the opportunity to configure the JIRA connection automatically. This is a quick way of setting up your JIRA integration with the most common options.

You can also configure the JIRA connections via the application administration screens. In that case, you will need to set up connections individually. There are two parts to the integration process:

- A peer-to-peer link between JIRA and the application for sharing information and facilitating integration features. This link is set up via Application Links.
- A client-server link between the application and JIRA for delegating user and group management to your JIRA server.

**Requirements:** You need JIRA 4.3 or later.

---

**On this page:**

- Connecting to JIRA in the Setup Wizard
- Troubleshooting
- Notes

**Related pages:**

- Getting started
- JIRA integration
- Connecting to JIRA for user management

---

**Connecting to JIRA in the Setup Wizard**

To configure JIRA integration while running the Stash setup wizard:

1. Configure the following setting in JIRA: [Allow remote API access](#).
2. Click Integrate with JIRA and enter the following information when you get to the ‘Connect to JIRA’ step of the setup wizard:

<table>
<thead>
<tr>
<th><strong>JIRA base URL</strong></th>
<th>The web address of your JIRA server. Examples are: <a href="http://www.example.com:8080/jira/">http://www.example.com:8080/jira/</a> <a href="http://jira.example.com">http://jira.example.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>JIRA admin username</strong></td>
<td>The credentials for a user with the ‘JIRA System Administrators’ global permission in JIRA.</td>
</tr>
<tr>
<td><strong>JIRA password</strong></td>
<td>JIRA will use this URL to access your Stash server. The URL you give here will override the base URL specified in your Stash administration console, for the purposes of the JIRA connection.</td>
</tr>
<tr>
<td><strong>Stash base URL</strong></td>
<td>JIRA will use this URL to access your Stash server. The URL you give here will override the base URL specified in your Stash administration console, for the purposes of the JIRA connection.</td>
</tr>
</tbody>
</table>
3. Click **Connect**.
4. Finish the setup process.

## Troubleshooting

**Click to see troubleshooting information...**

This section describes the possible problems that may occur when integrating your application with JIRA via the setup wizard, and the solutions for each problem.

<table>
<thead>
<tr>
<th>Symptom</th>
<th>Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>The setup wizard displays one of the following error messages:</td>
<td>The setup wizard failed to complete registration of the peer-to-peer application link with JIRA. JIRA integration is only partially configured.</td>
<td>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</td>
</tr>
<tr>
<td>• Failed to create application link from JIRA server at <code>&lt;URL&gt;</code> to this <code>&lt;application&gt;</code> server at <code>&lt;URL&gt;</code>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Failed to create application link from this <code>&lt;application&gt;</code> server at <code>&lt;URL&gt;</code> to JIRA server at <code>&lt;URL&gt;</code>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Failed to authenticate application link from JIRA server at <code>&lt;URL&gt;</code> to this <code>&lt;application&gt;</code> server at <code>&lt;URL&gt;</code>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Failed to authenticate application link from <code>&lt;application&gt;</code> server at <code>&lt;URL&gt;</code> to this JIRA server at <code>&lt;URL&gt;</code>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The setup wizard displays one of the following error messages:</td>
<td>The setup wizard failed to complete registration of the client-server link with JIRA for user management. The peer-to-peer link was successfully created, but integration is only partially configured.</td>
<td>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</td>
</tr>
<tr>
<td>• Failed to register <code>&lt;application&gt;</code> configuration in JIRA for shared user management. Received invalid response from JIRA: <code>&lt;response&gt;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Failed to register <code>&lt;application&gt;</code> configuration in JIRA for shared user management. Received: <code>&lt;response&gt;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The setup wizard displays the following error message:</td>
<td>The setup wizard successfully established the peer-to-peer link with JIRA, but could not persist the client-server link for user management in your config.xml file. This may be caused by a problem in your environment, such as a full disk.</td>
<td>Please investigate and fix the problem that prevented the application from saving the configuration file to disk. Then remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</td>
</tr>
<tr>
<td>• Error setting Crowd authentication</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The setup wizard displays the following error message:

- Error reloading Crowd authentication

The setup wizard has completed the integration of your application with JIRA, but is unable to start synchronizing the JIRA users with your application.

Restart your application. You should then be able to continue with the setup wizard. If this solution does not work, please contact Atlassian Support.

The setup wizard displays the following error message:

- An error occurred: java.lang.IllegalStateException: Could not create the application in JIRA/Crowd (code: 500). Please refer to the logs for details.

The setup wizard has not completed the integration of your application with JIRA. The links are only partially configured. The problem occurred because there is already a user management configuration in JIRA for this <application> URL.

Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.

No users can log in after you have set up the application with JIRA integration.

Possible causes:
- There are no users in the group that you specified on the 'Connect to JIRA' screen.
- For FishEye: There are no groups specified in the 'groups to synchronize' section of your administration console.
- For Stash: You may not have granted any JIRA groups or users permissions to log in to Stash.

Go to JIRA and add some usernames to the group.
- For FishEye: Go to the FishEye administration screens and specify at least one group to synchronize. The default is 'jira-users'.
- For Stash: Grant the Stash User permission to the relevant JIRA groups on the Stash Global permissions page.

If this solution does not work, please contact Atlassian Support.

---

Solution 1: Removing a Partial Configuration – The Easiest Way

If the application's setup wizard fails part-way through setting up the JIRA integration, you may need to remove the partial configuration from JIRA before continuing with your application setup. Please follow the steps below.

Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup wizard:

1. Log in to JIRA as a user with the 'JIRA System Administrators' global permission.
2. Click the 'Administration' link on the JIRA top navigation bar.
3. Remove the application link from JIRA, if it exists:
   a. Click 'Application Links' in the JIRA administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
   b. Look for a link to your application. It will have a base URL of the application linked to JIRA. For example:
      - If you want to remove a link between JIRA and FishEye, look for the one where the 'Application URL' matches the base URL of your FishEye server.
      - If you want to remove a link between JIRA and Confluence, look for the one where the 'Application URL' matches the base URL of your Confluence server.
      - If you want to remove a link between JIRA and Stash, look for the one where the 'Application URL' matches the base URL of your Stash server.
   c. Click the 'Delete' link next to the application link that you want to delete.
   d. A confirmation screen will appear. Click the 'Confirm' button to delete the application link.
4. Remove the user management configuration from JIRA, if it exists:
a. Go to the JIRA administration screen for configuring the applications that have been set up to use JIRA for user management:
   • In JIRA 4.3: Click ‘Other Applications’ in the ‘Users, Groups & Roles’ section of the JIRA administration screen.
   • In JIRA 4.4: Select 'Administration' > 'Users' > ‘JIRA User Server’.

b. Look for a link to your application. It will have a name matching this format:

\[
\text{<Type> - <HostName> - <Application ID>}
\]

For example:

\[
\text{FishEye / Crucible - localhost - 92004b08-5657-3048-b5dc-f886e662ba15}
\]

Or:

\[
\text{Confluence - localhost - 92004b08-5657-3048-b5dc-f886e662ba15}
\]

If you have multiple servers of the same type running on the same host, you will need to match the application ID of your application with the one shown in JIRA. To find the application ID:
   • Go to the following URL in your browser:

\[
\text{<baseUrl>/rest/applinks/1.0/manifest}
\]

Replace \text{<baseUrl>} with the base URL of your application.
For example:

\[
\text{http://localhost:8060/rest/applinks/1.0/manifest}
\]

• The application links manifest will appear. Check the application ID in the \text{id} element.

   c. In JIRA, click ‘Delete’ next to the application that you want to remove.

5. Go back to the setup wizard and try the ‘Connect to JIRA’ step again.

**Solution 2: Removing a Partial Configuration – The Longer Way**

If solution 1 above does not work, you may need to remove the partial configuration and then add the full integration manually. Please follow these steps:

1. Skip the 'Connect to JIRA' step and continue with the setup wizard, to complete the initial configuration of the application.
2. Log in to JIRA as a user with the ‘JIRA System Administrators’ global permission.
3. Click the ‘Administration’ link on the JIRA top navigation bar.
4. Remove the application link from JIRA, if it exists:
   a. Click ‘Application Links’ in the JIRA administration menu. The ‘Configure Application Links’ page will appear, showing the application links that have been set up.
   b. Look for a link to your application. It will have a base URL of the application linked to JIRA. For example:
      • If you want to remove a link between JIRA and FishEye, look for the one where the ‘Application
If the ‘Application URL’ matches the base URL of your FishEye server.

- If you want to remove a link between JIRA and Confluence, look for the one where the ‘Application URL’ matches the base URL of your Confluence server.

- If you want to remove a link between JIRA and Stash, look for the one where the ‘Application URL’ matches the base URL of your Stash server.

c. Click the ‘Delete’ link next to the application link that you want to delete.

d. A confirmation screen will appear. Click the ‘Confirm’ button to delete the application link.

5. Remove the user management configuration from JIRA, if it exists:

a. Go to the JIRA administration screen for configuring the applications that have been set up to use JIRA for user management:

   - In JIRA 4.3: Click ‘Other Applications’ in the ‘Users, Groups & Roles’ section of the JIRA administration screen.
   - In JIRA 4.4: Select ‘Administration’ > ‘Users’ > ‘JIRA User Server’.

b. Look for a link to your application. It will have a name matching this format:

   `<Type> - <HostName> - <Application ID>`

For example:

   FishEye / Crucible - localhost - 92004b08-5657-3048-b5dc-f886e662ba15

Or:

   Confluence - localhost - 92004b08-5657-3048-b5dc-f886e662ba15

If you have multiple servers of the same type running on the same host, you will need to match the application ID of your application with the one shown in JIRA. To find the application ID:

- Go to the following URL in your browser:

  `<baseUrl>/rest/applinks/1.0/manifest`

Replace `<baseUrl>` with the base URL of your application.

For example:

   http://localhost:8060/rest/applinks/1.0/manifest

- The application links manifest will appear. Check the application ID in the `<id>` element.

c. In JIRA, click ‘Delete’ next to the application that you want to remove.

6. Add the application link in JIRA again, so that you now have a two-way trusted link between JIRA and your application:

a. Click ‘Add Application Link’. Step 1 of the link wizard will appear.

b. Enter the server URL of the application that you want to link to (the ‘remote application’).

c. Click the ‘Next’ button.

d. Enter the following information:

   - ‘Create a link back to this server’ – Tick this check box to add a two-way link between the two applications.
   - ‘Username’ and ‘Password’ – Enter the credentials for a username that has administrator
access to the remote application.

Note: These credentials are only used to authenticate you to the remote application, so that Application Links can make the changes required for the new link. The credentials are not saved.

- **Reciprocal Link URL** – The URL you give here will override the base URL specified in your remote application’s administration console, for the purposes of the application links connection. Application Links will use this URL to access the remote application.

e. Click the **Next** button.

f. Enter the information required to configure authentication for your application link:

- **The servers have the same set of users** – Tick this check box, because the users are the same in both applications.
- **These servers fully trust each other** – Tick this check box, because you trust the code in both applications and are sure both applications will maintain the security of their private keys.

For more information about configuring authentication, see Configuring Authentication for an Application Link.

g. Click the **Create** button to create the application link.

7. Configure a new connection for user management in JIRA:

a. Go to the JIRA administration screen for configuring the applications that have been set up to use JIRA for user management:

   - In JIRA 4.3: Click **Other Applications** in the ‘Users, Groups & Roles’ section of the JIRA administration screen.
   - In JIRA 4.4: Select ‘Administration’ > ‘Users’ > ‘JIRA User Server’.

b. Add an application.

c. Enter the **application name** and **password** that your application will use when accessing JIRA.

d. Enter the **IP address** or addresses of your application. Valid 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.

e. Save the new application.

8. Set up the JIRA user directory in the application.

   - For Confluence:
     a. Go to the Confluence Administration Console.
     b. Click ‘User Directories’ in the left-hand panel.
     c. Add a directory and select type ‘Atlassian JIRA’.
     d. Enter the following information:
        - **Name** – Enter the name of your JIRA server.
        - **Server URL** – Enter web address of your JIRA server. Examples:
          
          http://www.example.com:8080/jira/
          http://jira.example.com

        - **Application name** and **Application password** – Enter the values that you defined for Confluence in the settings on JIRA.
     e. Save the directory settings.
     f. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the ‘User Directories’ screen.

   For details see Connecting to Crowd or JIRA for User Management.

   - For FishEye/Crucible:
     a. Click ‘Authentication’ on the FishEye/Crucible ‘Admin Menu’.
     b. Click ‘Edit’ in the section titled ‘JIRA or Crowd Authentication’.
     c. Enter the following information:
        - **Application name** and **Application password** – Enter the values that you defined for your application in the settings on JIRA.
• **JIRA/Crowd URL** – The web address of your JIRA server. Examples:

```
http://www.example.com:8080/jira/
http://jira.example.com
```

• **Auto-add** – Select ‘Create a FishEye/Crucible user on successful login’ (default) to ensure that your JIRA users will be automatically enrolled into FishEye/Crucible when they first log in via JIRA.

• **Synchronize users with JIRA/Crowd** – Select ‘Yes’ (default) to ensure that JIRA will synchronize all changes in the user information on a regular basis. The synchronization interval is set to 60 minutes (1 hour) by default.

• **Single sign on (SSO)** – Select ‘Disabled’. This option is not available when using JIRA for user management and if enabled will make the integration fail.

• **Groups of users to synchronize** – Select at least one group to synchronize. The default is ‘jira-users’.

d. Click ‘Apply changes’.

• For Stash:
  a. Go to the Stash Administration Console.
  b. Click ‘User Directories’ in the left-hand panel.
  c. Add a directory and select type ‘Atlassian JIRA’.
  d. Enter the following information:
     • **Name** – Enter the name of your JIRA server.
     • **Server URL** – Enter web address of your JIRA server. Examples:

```
http://www.example.com:8080/jira/
http://jira.example.com
```

• **Application name** and **Application password** – Enter the values that you defined for Stash in the settings on JIRA.

e. Save the directory settings.

f. Define the **directory order** by clicking the blue up- and down-arrows next to each directory on the ‘User Directories’ screen.

For details see [Connecting to JIRA for user management](#).

**Notes**

When you connect to JIRA in the setup wizard, the setup procedure will configure OAuth authentication between Stash and JIRA. See [Configuring OAuth Authentication for an Application Link](#) for more information.

**Using Stash**

This section describes the basics of using Stash.

---

**In this section:**

- Creating projects and managing permissions for a project
- Creating repositories
- Basic Git commands
- Permanently authenticating with Git repositories
- Setting up SSH
  - Setting up SSH on Windows
  - Setting up SSH on Linux and Mac
Creating projects and managing permissions for a project

Projects allow you to group repositories and to manage permissions for them in an aggregated way.

On this page:

1. Create a project
2. Manage a project's permissions

1. Create a project

Click on Create Project:

![Create Project button](image)

Fill out the form and click Create Project when you're done.
We recommend that you use a short project key. It will be used as an identifier for your project and will appear in the URLs.

2. Manage a project's permissions

Simply click on the Permissions tab for the project.

There are 3 types of permissions that you can assign to a user or group for a project: Observer, Contributor or Project Administrator.

<table>
<thead>
<tr>
<th>Role</th>
<th>Browse</th>
<th>Clone / Pull</th>
<th>Push</th>
<th>Create Repositories</th>
<th>Edit settings / permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observer</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Contributor</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Project Administrator</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

Click Add Users or Add Groups in the relevant section to grant permissions to particular users or groups.

Use the check boxes to modify specific permissions for a user or group.
Creating repositories

Repositories allow you to collaborate on code with your co-workers.

⚠️ In order to create repositories you need to be a Project Admin of the project you want to add a repository to.

Related pages:
- Getting started with Git and Stash
- Global permissions
- Creating projects and managing permissions for a project

Once a repository is created, the project permissions are applied to the repository. That means all repositories created in a project share the same access and permission settings.

Click Create Repository to open the repository creation form.
Once submitted you will be taken directly to your repository homepage. As there is no content in your repository yet, you'll see some instructions to help you push code to your repository.

You will find your clone URL in the upper right corner of the repository homepage. You can use this URL and share it with other people.

Let other people collaborate with you

In order to grant users access to this repository you have to set up the permission at the parent project level. More information is available on Creating projects and managing permissions for a project.

Basic Git commands

Here is a list of some basic Git commands to get you going with Git.

See Git resources for Git cheat sheets and other resources.

<table>
<thead>
<tr>
<th>Git task</th>
<th>Notes</th>
<th>Git commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a new local repository</td>
<td></td>
<td>git init</td>
</tr>
<tr>
<td>Check out a repository</td>
<td>Create a working copy of a local repository:</td>
<td>git clone /path/to/repository</td>
</tr>
<tr>
<td><strong>For a remote server, use:</strong></td>
<td>git clone username@host:/path/to/repository</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>
| **Add files** Add one or more files to staging (index): | git add <filename>  
git add * |
| **Commit** Commit changes to head (but not yet to the remote repository): | git commit -m "Commit message"  
Commit any files you've added with `git add`, and also commit any files you've changed since then: | git commit -a |
| **Push** Send changes to the master branch of your remote repository: | git push origin master |
| **Status** List the files you've changed and those you still need to add or commit: | git status |
| **Connect to a remote repository** If you haven't connected your local repository to a remote server, add the server to be able to push to it: | git remote add origin <server> |
| **Branches** Create a new branch and switch to it: | git checkout -b <branchname>  
Switch from one branch to another: | git checkout <branchname>  
List all the branches in your repo, and also tell you what branch you're currently in: | git branch  
Delete the feature branch: | git branch -d <branchname>  
Push the branch to your remote repository, so others can use it: | git push origin <branchname>  
Push all branches to your remote repository: | git push --all origin  
Delete a branch on your remote repository: | git push origin :<branchname> |
| **Update from the remote repository** Fetch and merge changes on the remote server to your working directory: | git pull  
To merge a different branch into your active branch: | git merge <branchname> |
View all the merge conflicts:  
git diff

View the conflicts against the base file:  
git diff --base <filename>

Preview changes, before merging:  
git diff <sourcebranch> <targetbranch>

After you have manually resolved any conflicts, you mark the changed file:  
git add <filename>

**Tags**

You can use tagging to mark a significant changeset, such as a release:  
git tag 1.0.0 <commitID>

CommitId is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:  
git log

Push all tags to remote repository:  
git push --tags origin

**Undo local changes**

If you mess up, you can replace the changes in your working tree with the last content in head:  
git checkout -- <filename>

Changes already added to the index, as well as new files, will be kept.

Instead, to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it, do this:  
git fetch origin

git reset --hard origin/master

**Search**

Search the working directory for `foo()`:  
git grep "foo()"

---

**Permanently authenticating with Git repositories**

Currently, Atlassian Stash only supports HTTP or HTTPS for pushing and pulling from managed Git repositories. Git has no method of caching the user’s credentials, so you need to re-enter them each time you perform a clone, push or pull.

This page describes two methods for permanently authenticating with Git repositories so that you can avoid typing your username and password each time you are pushing to or pulling from Stash.

---

**On this page:**

- Using credential caching
- Using the `.netrc` file
Using credential caching

⚠️ You need Git 1.7.9 or above to use the HTTPS Credentials Caching feature.

Windows

On Windows you can use the application `git-credential-winstore`.

1. Download the software.
2. Run it.
3. You will be prompted for credentials the first time you access a repository, and Windows will store your credentials for use in the future.

Linux

On Linux you can use the 'cache' authentication helper that is bundled with git 1.7.9 and higher. From the git documentation:

This command caches credentials in memory for use by future git programs. The stored credentials never touch the disk, and are forgotten after a configurable timeout. The cache is accessible over a Unix domain socket, restricted to the current user by filesystem permissions.

Run the command below to enable credential caching. After enabling credential caching any time you enter your password it will be cached for 1 hour (3600 seconds):

```
    git config --global credential.helper 'cache --timeout 3600'
```

Run the command below for an overview of all configuration options for the 'cache' authentication helper:

```
    git help credential-cache
```

OSX

Follow these steps if you want to use Git with credential caching on OSX:

1. Download the binary `git-credential-osxkeychain`.
2. Run the command below to ensure the binary is executable:

```
    chmod a+x git-credential-osxkeychain
```

3. Put it in the directory `/usr/local/bin`. 
4. Run the command below:

```
$ git config --global credential.helper osxkeychain
```

**Using the `.netrc` file**

The `.netrc` file is a mechanism that allows you to specify which credentials to use for which server. This method allows you to avoid entering a username and password every time you push to or pull from Git, but your Git password is stored in plain text.

⚠️ **Warning!**
- Git uses a utility called `cURL` under the covers, which respects the use of the `.netrc` file. Be aware that other applications that use cURL to make requests to servers defined in your `.netrc` file will also now be authenticated using these credentials. Also, this method of authentication is potentially unsuitable if you are accessing your Stash server via a proxy, as all cURL requests that target a path on that proxy server will be authenticated using your `.netrc` credentials.
- cURL will not match the machine name in your `.netrc` if it has a username in it, so make sure you edit your `.git/config` file in the root of your clone of the repository and remove the user and '@' part from any clone URL’s (URL fields) that look like `https://user@machine.domain.com/...` to make them look like `http://machine.domain.com/...`

**Windows**

1. Create a text file called `_netrc` in your home directory (e.g. `c:\users\kannonboy\.netrc`). cURL has problems resolving your home directory if it contains spaces in its path (e.g. `c:\Documents and Settings\kannonboy`). However, you can update your `%HOME%` environment variable to point to any directory, so create your `_netrc` in a directory with no spaces in it (for example `c:\curl-auth\`) then set your `%HOME%` environment variable to point to the newly created directory.

2. Add credentials to the file for the server or servers you want to store credentials for, using the format described below:

```
machine stash1.mycompany.com
login myusername
password mypassword

machine stash2.mycompany.com
login myotherusername
password myotherpassword
```

**Linux or OSX**

1. Create a file called `.netrc` in your home directory (`~/.netrc`). Unfortunately, the syntax requires you to store your passwords in plain text - so make sure you modify the file permissions to make it readable only to you.

2. Add credentials to the file for the server or servers you want to store credentials for, using the format described in the 'Windows' section above. You may use either IP addresses or hostnames, and you do not need to specify a port number, even if you’re running Stash on a non-standard port.

3. And that’s it! Subsequent `git clone`, `git pull` and `git push` requests will now be authenticated using the credentials specified in this file.
Setting up SSH

You can use SSH keys to establish a secure connection between your computer and Stash for when you are performing Git operations. Stash supports DSA and RSA2 key types. RSA1 is not supported. A Stash user can add any number of keys to their account.

Before you can do this, however, the following must have already been done:

- you need to have added your SSH key to Stash, either on Windows, or on Linux and Mac OS X.
- your Stash administrator must have already enabled SSH access, on Stash.

Once you have SSH access set up, using it is easy!

**Related pages:**

- Setting up SSH on Windows
- Setting up SSH on Linux and Mac
- Enabling SSH access to Git repositories in Stash

Using SSH to access your repositories:

1. SSH access needs to be set up, as described above. Once this is done, you can use SSH access as follows:
2. Go to Projects, choose a project, click Repositories, and then choose a repository.
3. At the top right, you'll see the clone URLs for the repository.
4. Choose the clone URL you want to use. SSH is available if you have already added an SSH key to your profile. If you haven't done that yet, click Add an SSH key. See Setting up SSH on Windows or Setting up SSH on Linux and Mac for details.

Setting up SSH on Windows

You can use SSH keys to establish a secure connection between your computer and Stash for when you are performing Git operations.

There are a few steps to set up SSH keys for Stash, but it's quite straightforward, so follow along.

*If you already have a key you want to use, go to step 4.*

- Your Stash administrator must have already enabled SSH access to Git repositories.
- Supported key types are DSA and RSA2. RSA1 is not supported.

**On this page:**

1. Check for existing SSH keys
2. Back up old SSH keys
3. Generate a new SSH key
4. Add an SSH key to Stash
1. Check for existing SSH keys

You should check for existing SSH keys on your local computer. If you already have a key pair that you want to use, you can **go to step 4**.

Open a command prompt, and run the following:

```
cd ~/.ssh
```

- If you see "No such file or directory, then there aren't any existing keys: **go to step 3**.
- Check to see if you have a key already:

```
dir id_`
```

- If there are existing keys, you may want to use them: **go to step 4**.

2. Back up old SSH keys

If you have existing SSH keys, but you don't want to use them when connecting to Stash, you should back those up.

Do this in a command prompt on your local computer, by running:

```
mkdir key_backup
cp id_rsa* key_backup
```

Now generate a new SSH key.

3. Generate a new SSH key

If you don't have an existing SSH key to use, you need to generate one.

1. Log in to your local computer as an administrator.
2. Open a command prompt, and run the following:

```
ssh-keygen
```

Note that the `ssh-keygen` command is only available if you have already **installed Git** (with Git Bash). You'll see a response similar to this:
3. Just press `<Enter>` to accept the default location and file name. If the `.ssh` directory doesn’t exist, the system creates one for you.

4. Enter, and re-enter, a passphrase when prompted. The whole interaction will look similar to this:

```
C:\Users\ASUS>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/ASUS/.ssh/id_rsa):
```

```
C:\Users\ASUS>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/ASUS/.ssh/id_rsa):
Created directory '/c/Users/ASUS/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/ASUS/.ssh/id_rsa.
Your public key has been saved in /c/Users/ASUS/.ssh/id_rsa.pub.
The key fingerprint is:
C:\Users\ASUS>
```

5. You’re done! Now add the new key to Stash.

4. **Add an SSH key to Stash**

1. In your command prompt, change directory to the `.ssh` directory, and copy the public key file to your clipboard, by running:

   ```
   cd .ssh
   clip < id_rsa.pub
   ```

2. In Stash, go to your profile:

3. Click on **SSH Keys** and then **Add Key**:

4. Paste the key into the text box:
Setting up SSH on Linux and Mac

You can use SSH keys to establish a secure connection between your computer and Stash for when you are performing Git operations.

There are a few steps to set up SSH keys for Stash, but it’s quite straightforward, so follow along.

If you already have a key you want to use, go to step 4.

- Your Stash administrator must have already enabled SSH access to Git repositories.
- Supported key types are DSA and RSA2. RSA1 is not supported.

At the time of writing, on Mac OS X, the Git tar archive can fail on special characters when using SSH to secure connections between your computer and Stash. (The Git archive command allows you to download as a single file the files in a checkout of the Git repository.)

We recommend that you use the zip format; you can set that using the following command:

```
git archive --format zip
```

On this page:

1. Check for existing SSH keys
2. Back up old SSH keys
3. Generate a new key
4. Add a key to Stash

Related pages:

- Setting up SSH on Windows
- Setting up SSH port forwarding
- Permanently authenticating with Git repositories

1. Check for existing SSH keys

You should check for existing SSH keys on your local computer. If you already have a key pair that you want to use, you can go to step 4.

Open a terminal and run the following:
cd ~/.ssh

- If you see "No such file or directory, then there aren't any existing keys: go to step 3.
- Check to see if you have a key already:

  ls id_*

- If there are existing keys, you may want to use them: go to step 4.

2. Back up old SSH keys

If you have existing SSH keys, but you don't want to use them when connecting to Stash, you should back those up.

Do this in a terminal on your local computer, by running:

  mkdir key_backup
  cp id_rsa* key_backup

Now generate a new SSH key.

3. Generate a new key

If you don't have an existing SSH key to use, you need to generate one.

   1. Open a terminal on your local computer and enter the following:

      ssh-keygen

You'll see a response similar to this:

```
pwats:Mac-Pro:~ pwatson$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/pwatson/.ssh/id_rsa):
```

   2. Just press <Enter> to accept the default location and file name. If the .ssh directory doesn't exist, the system creates one for you.
   3. Enter, and re-enter, a passphrase when prompted.
      The whole interaction will look similar to this:
4. You're done! Now add the new key to Stash.

4. Add a key to Stash

1. In your terminal, copy the public key file to your clipboard by entering:

   **Mac OS X**
   ```
pbcopy < ~/.ssh/id_rsa.pub
   ```

   **Linux**
   ```
sudo apt-get install xclip
   xclip -sel clip < ~/.ssh/id_rsa.pub
   ```

   Note that on Linux, you may need to download and install xclip, as shown in the code snippet above.

2. In Stash, go to your profile:

3. Click on **SSH Keys** and then **Add Key**:

4. Paste the key into the text box:
Administering Stash

This section describes some of the administrative actions that can be performed from the Stash Administration user interface.

In this section:
- Users and groups
- Global permissions
- JIRA integration
  - JIRA compatibility
- External user directories
  - Configuring an LDAP directory
  - Configuring delegated LDAP authentication
  - Connecting to Crowd
  - Connecting to JIRA for user management
- Setting up your mail server
- Specifying the base URL for Stash
- Connecting Stash to an external database
- Enabling SSH access to Git repositories in Stash

Related pages:
- Supported platforms
- Using Stash
- Stash FAQ

System administration actions that can be performed outside of the Stash user interface include:

- Enabling Stash debug logging
- Integrating Stash with Apache HTTP Server
- Scaling Stash
- Securing Stash with HTTPS
- System settings
- Setting up SSH port forwarding
- Data recovery and backups

Users and groups
Stash comes with an internal user directory already built-in that is enabled by default at installation. When you create the first administrator during the setup procedure, that administrator’s username and other details are stored in the internal directory.

Stash Admins and Sys Admins can manage users and groups in Stash as described on this page. You can also set up Stash to use external user directories.

Note that access permissions can also be applied to projects.

On this page:

- Creating a user
- Creating a group
- Adding users to groups
  - From the user profile
  - From the group page

Related pages:

- Getting started with Stash
- External user directories

Creating a user

In the Administration section, click Create a User to go directly to the user creation form:
Create user

There are 24 users remaining on your 500 user license

Username*  jdoe
Full name*   John Doe
Email address*  john@doe.com

Email a link to the user to set their password

[Create User] [Cancel]

Once you've created a user, click **Change permissions** to set up their access permissions.
Creating a group

In the Administration section, click Create a Group, and then enter the name for the new group:

Create group

Group name: Red team

Now you can add users to your new group:
Adding users to groups

You can add users to groups in two ways:

- add a particular user to multiple groups, from the user's profile
- add multiple users to a particular group, from the group's page.

From the user profile

To add a user to a group from the user's profile, go to Users in the Administration section, and use the filter to find the user:

<table>
<thead>
<tr>
<th>Name</th>
<th>Username</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Doe</td>
<td>jdoe</td>
<td><a href="mailto:john@doe.com">john@doe.com</a></td>
</tr>
</tbody>
</table>

On the page for the user, click Add Group to go to the list of available groups:
You can use the filter to find the group you want to add the user to. Click **Add Group** to make the user a member of the group.

**Available Groups**

<table>
<thead>
<tr>
<th>Filter groups</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ams-sales</td>
<td></td>
</tr>
<tr>
<td>ams-staff</td>
<td><strong>Add Group</strong></td>
</tr>
<tr>
<td>another test group</td>
<td></td>
</tr>
<tr>
<td>allaseye-users</td>
<td></td>
</tr>
<tr>
<td>atlassian-administration</td>
<td></td>
</tr>
</tbody>
</table>

Click **Done** when you have finished.

**From the group page**

To add a user to a group from the group's page, go to **Groups** in the **Administration** section, and use the filter to find the group:
On the page for the group, click **Add Users** to go to the list of available users:

In the user picker, click **Add user** to make a user a member of the group:
Global permissions

Stash uses four levels of permissions to control user and group access to Stash projects and to the Stash server configuration.

<table>
<thead>
<tr>
<th></th>
<th>Login / Browse</th>
<th>Create projects</th>
<th>Manage users / groups</th>
<th>Manage global permissions</th>
<th>Edit application settings</th>
<th>Edit server config</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stash User</td>
<td>✔</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Project Creator</td>
<td>✔</td>
<td>✔</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Administrator</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>❌</td>
</tr>
<tr>
<td>System Administrator</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

Note that you can also apply access permissions to projects.

**Related pages:**
- Getting started with Stash
- Users and groups
- Manage project permissions

To edit the permissions for a Stash user or group:

1. Click Administration in the top menu.
2. Choose Accounts > Permissions.
3. Click Add Users or Add Groups if you want to set the permissions for additional users or groups.

You can remove a user or group from an access level by clicking the X at the right-hand end of the row.
JIRA integration

When Stash is integrated with Atlassian JIRA, you can:

- See the JIRA issues related to particular commits in Stash, as shown in this screenshot of the Commits tab for a Stash project, and click through to the issues in JIRA.
- Use JIRA for delegated user management. See External user directories.

Stash integrates with JIRA by means of an 'application link'. You can set up application links either during the Stash install process, or as described below.

⚠️ Stash integration with JIRA requires an upgraded version of the FishEye plugin in JIRA. See JIRA compatibility for details about upgrading the JIRA FishEye plugin, and for download links to the upgraded plugin versions.

**On this page:**

- Linking Stash with JIRA
- Known issues with the JIRA integration
Linking Stash with JIRA

To link Stash to a JIRA server:

Go to the Configure Application Links screen in Stash:

![Stash Configure Application Links](image)

Click Add Application Link:

![Stash Add Application Link](image)

Complete the application link wizard to connect Stash to your JIRA server. It is recommended that you make use of the automatic link-back from JIRA to Stash.

More detailed information about application links can be found on Configuring Application Links.

Once finished, you will not have to do any extra configuration. Your JIRA issues will appear in the changesets and commit lists in Stash. On the JIRA side, the commits associated with a specific issue will appear in the issue's Source tab.
Known issues with the JIRA integration

We have tried to make the integration of JIRA with Stash as straightforward as possible. However, we are aware of the following issues:

- Stash only supports one JIRA server; we only pick the primary one.
- There is no checking for project or issue-key validity; Stash may link to issues that do not actually exist.

We apologise for the inconvenience. You can watch the 2 issues below to keep track of our progress:

- [STASH-2471](https://example.com/stash-2471) - Authenticate to see issue details
- [STASH-2470](https://example.com/stash-2470) - Authenticate to see issue details

JIRA compatibility

Atlassian JIRA's support for Stash is built into the FishEye plugin that is bundled with JIRA. This allows you to see all of your code changes in one place, even if you’re running multiple Atlassian FishEye and Stash servers.

Supported JIRA Versions

If you're using a version of JIRA earlier than 5.0.2 you may need to upgrade the FishEye plugin in JIRA to get support for Atlassian Stash.

<table>
<thead>
<tr>
<th>JIRA Version</th>
<th>Compatibility</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0.4+</td>
<td>Works straight out of the box!</td>
<td>NA</td>
</tr>
<tr>
<td>5.0-5.0.3</td>
<td>Requires <a href="https://example.com/jira-fisheye-plugin-5.0.4.1">JIRA FishEye Plugin</a></td>
<td><a href="https://example.com">https://example.com/jira-fisheye-plugin-5.0.4.1</a></td>
</tr>
<tr>
<td>4.4.x</td>
<td>Requires <a href="https://example.com/jira-fisheye-plugin-3.4.12">JIRA FishEye Plugin</a></td>
<td><a href="https://example.com">https://example.com/jira-fisheye-plugin-3.4.12</a></td>
</tr>
<tr>
<td>4.3.x</td>
<td>Requires <a href="https://example.com/jira-fisheye-plugin-3.1.8">JIRA FishEye Plugin</a></td>
<td><a href="https://example.com">https://example.com/jira-fisheye-plugin-3.1.8</a></td>
</tr>
<tr>
<td>Earlier versions</td>
<td>JIRA-to-Stash integration is unsupported</td>
<td>NA</td>
</tr>
</tbody>
</table>

**JIRA 4.3.x Upgrade Note**

When upgrading the plugin in JIRA 4.3.x you may see a "zip file closed" error message in the logs. This can be ignored. See "[IllegalStateException: zip file closed] when upgrading JIRA FishEye Plugin in JIRA 4.3" for more details.

Upgrade Guide

To upgrade the plugin, copy the link from the table above that matches your JIRA version. Then navigate in JIRA...
Next, select the **Install Plugins** tab and click **Upload Plugin**.

Now paste the URL copied from the table above into the **From this URL** field, and click **Upload**.
You should see that the plugin is installed. Now you can continue integrating Atlassian Stash with your JIRA server. See [JIRA integration](#) for details.

**External user directories**

You can connect Stash to external user directories. This allows you to make use of existing users and groups stored in an enterprise directory.

Note that Stash comes with an internal user directory, already built-in, that is enabled by default at installation. When you create the first administrator during the setup procedure, that administrator’s username and other details are stored in the internal directory.

---

**Connecting Atlassian Stash to your external directory will not be sufficient to allow your users to log in to Stash. You will need to grant them explicitly the right to use Atlassian Stash in the global permission screen.**

We recommend you to use groups instead of individual accounts when granting permissions.

---

**LDAP**

You can connect Stash to an LDAP directory for two purposes:

- For full user and group management, and for authentication — see [Configuring an LDAP directory](#) for instructions.
- For delegated authentication only, while still using the internal directory for user and group management — see [Configuring delegated LDAP authentication](#) for instructions.

You should consider connecting to an LDAP directory server if your users and groups are stored in an enterprise directory.
Stash is able to connect to the following LDAP directory servers:

- Microsoft Active Directory
- Apache Directory Server (ApacheDS) 1.0.x and 1.5.x
- Apple Open Directory (Read-Only)
- Fedora Directory Server (Read-Only Posix Schema)
- Novell eDirectory Server
- OpenDS
- OpenLDAP
- OpenLDAP (Read-Only Posix Schema)
- Generic Posix/RFC2307 Directory (Read-Only)
- Sun Directory Server Enterprise Edition (DSEE)
- Any generic LDAP directory server

**Crowd**

You can connect Stash to Atlassian Crowd for user and group management, and for authentication.

Crowd is an application security framework that handles authentication and authorisation for your web-based applications. With Crowd you can integrate multiple web applications and user directories, with support for single sign-on (SSO) and centralised identity management. See the [Crowd Administration Guide](#).

You should consider connecting to Crowd if you want to use Crowd to manage existing users and groups in multiple directory types, or if you have users of other web-based applications.

See [Connecting to Crowd](#) for configuration instructions.

**JIRA**

You can delegate Stash user and group management, and authentication, to an Atlassian JIRA instance.

You should consider connecting to JIRA as an alternative to using Atlassian Crowd for simple configurations with a limited number of users. Stash can only connect to a JIRA server running JIRA 4.3 or later.

See [Connecting to JIRA for user management](#) for configuration instructions.
Configuring an LDAP directory

You can connect Stash to an external LDAP user directory. This allows you to make use of existing users and groups stored in an enterprise directory.

Stash is able to connect to the following LDAP directory servers:

- Microsoft Active Directory
- Apache Directory Server (ApacheDS) 1.0.x and 1.5.x
- Apple Open Directory (Read-Only)
- Fedora Directory Server (Read-Only Posix Schema)
- Novell eDirectory Server
- OpenDS
- OpenLDAP
- OpenLDAP (Read-Only Posix Schema)
- Generic Posix/RFC2307 Directory (Read-Only)
- Sun Directory Server Enterprise Edition (DSEE)
- Any generic LDAP directory server

To connect Stash to an LDAP directory:

1. Log in as a user with 'Admin' permission.
2. Click Administration in the top menu.
3. Choose Accounts > User Directories.
4. Click Add Directory and select either Microsoft Active Directory or LDAP as the directory type.
5. Configure the directory settings, as described in the tables 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 'User
Directories’ screen. The directory order has the following effects:

- 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.

**Server settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>Enter a meaningful name to help you identify the LDAP directory server. Examples:</td>
</tr>
<tr>
<td></td>
<td>• Example Company Staff Directory</td>
</tr>
<tr>
<td></td>
<td>• Example Company Corporate LDAP</td>
</tr>
<tr>
<td><strong>Directory Type</strong></td>
<td>Select the type of LDAP directory that you will connect to. If you are adding a new LDAP connection, the value you select here will determine the default values for many of the options on the rest of screen. Examples:</td>
</tr>
<tr>
<td></td>
<td>• Microsoft Active Directory</td>
</tr>
<tr>
<td></td>
<td>• OpenDS</td>
</tr>
<tr>
<td></td>
<td>• And more.</td>
</tr>
<tr>
<td><strong>Hostname</strong></td>
<td>The host name of your directory server. Examples:</td>
</tr>
<tr>
<td></td>
<td>• ad.example.com</td>
</tr>
<tr>
<td></td>
<td>• ldap.example.com</td>
</tr>
<tr>
<td></td>
<td>• opens.example.com</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>The port on which your directory server is listening. Examples:</td>
</tr>
<tr>
<td></td>
<td>• 389</td>
</tr>
<tr>
<td></td>
<td>• 10389</td>
</tr>
<tr>
<td></td>
<td>• 636 (for example, for SSL)</td>
</tr>
<tr>
<td><strong>Use SSL</strong></td>
<td>Tick this check box if the connection to the directory server is an SSL (Secure Sockets Layer) connection. Note that you will need to configure an SSL certificate in order to use this setting.</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>The distinguished name of the user that the application will use when connecting to the directory server. Examples:</td>
</tr>
<tr>
<td></td>
<td>• cn=administrator,cn=users,dc=ad,dc=example,dc=com</td>
</tr>
<tr>
<td></td>
<td>• cn=user,dc=domain,dc=name</td>
</tr>
<tr>
<td></td>
<td>• <a href="mailto:user@domain.name">user@domain.name</a></td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password of the user specified above.</td>
</tr>
</tbody>
</table>

**LDAP schema**
## Setting | Description
--- | ---
**Base DN** | The root distinguished name (DN) to use when running queries against the directory server. Examples:
- o=example,c=com
- cn=users,dc=ad,dc=example,dc=com
- For Microsoft Active Directory, specify the base DN in the following format: `dc=domain1,dc=local`. You will need to replace `domain1` and `local` or 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.

**Additional User DN** | This value is used in addition to the base DN when searching and loading users. If no value is supplied, the subtree search will start from the base DN. Example:
- `ou=Users`

**Additional Group DN** | This value is used in addition to the base DN when searching and loading groups. If no value is supplied, the subtree search will start from the base DN. Example:
- `ou=Groups`

## LDAP permission

| Setting | Description |
--- | --- |
**Read Only** | LDAP users, groups and memberships are retrieved from your directory server and can only be modified via your directory server. You cannot modify LDAP users, groups or memberships via the application administration screens. |

**Read Only, with Local Groups** | LDAP users, groups and memberships are retrieved from your directory server and can only be modified via your directory server. You cannot modify LDAP users, groups or memberships via the application administration screens. However, you can add groups to the internal directory and add LDAP users to those groups. |

## Advanced settings

<p>| Setting | Description |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>Enable or disable support for nested groups. Some directory servers allow you to define a group as a member of another group. Groups in such a structure are called 'nested groups'. 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.</td>
</tr>
<tr>
<td>Use Paged Results</td>
<td>Enable or disable the use of the LDAP control extension for simple paging of search results. If paging is enabled, the search will retrieve sets of data rather than all of the search results at once. Enter the desired page size – that is, the maximum number of search results to be returned per page when paged results are enabled. The default is 1000 results.</td>
</tr>
<tr>
<td>Follow Referrals</td>
<td>Choose whether to allow the directory server to redirect requests to other servers. This option uses the node referral (JNDI lookup java.naming.referral) configuration setting. It is generally needed for Active Directory servers configured without proper DNS, to prevent a 'javax.naming.PartialResultException: Unprocessed Continuation Reference(s)' error.</td>
</tr>
<tr>
<td>Naive DN Matching</td>
<td>If your directory server will always return a consistent string representation of a DN, you can enable naive DN matching. Using naive DN matching will result in a significant performance improvement, so we recommend enabling it where possible.</td>
</tr>
</tbody>
</table>

This setting determines how your application will compare DNs to determine if they are equal.

- If this check box is ticked, the application will do a direct, case-insensitive, string comparison. This is the default and recommended setting for Active Directory, because Active Directory guarantees the format of DNs.
- If this check box is not ticked, the application will parse the DN and then check the parsed version.
Enable Incremental Synchronisation

Enable incremental synchronisation if you only want changes since the last synchronisation to be queried when synchronising a directory.

⚠️ Please be aware that when using this option, the user account configured for synchronisation must have read access to:

- The uSNChanged attribute of all users and groups in the directory that need to be synchronised.
- The objects and attributes in the Active Directory deleted objects container (see Microsoft's Knowledge Base Article No. 892806 for details).

If at least one of these conditions is not met, you may end up with users who are added to (or deleted from) the Active Directory not being respectively added (or deleted) in JIRA.

Synchronisation Interval (minutes)

Synchronisation 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.

Read Timeout (seconds)

The time, in seconds, to wait for a response to be received. If there is no response within the specified time period, the read attempt will be aborted. A value of 0 (zero) means there is no limit. The default value is 120 seconds.

Search Timeout (seconds)

The time, in seconds, to wait for a response from a search operation. A value of 0 (zero) means there is no limit. The default value is 60 seconds.

Connection Timeout (seconds)

This setting affects two actions. The default value is 0.

- The time to wait when getting a connection from the connection pool. A value of 0 (zero) means there is no limit, so wait indefinitely.
- The time, in seconds, to wait when opening new server connections. A value of 0 (zero) means that the TCP network timeout will be used, which may be several minutes.

### User schema settings

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

Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
| User Object Class | This is the name of the class used for the LDAP user object. Example:  
| | • user  
| User Object Filter | The filter to use when searching user objects. Example:  
| | • (&(objectCategory=Person)(sAMAccountName=*))  
| User Name Attribute | The attribute field to use when loading the username. Examples:  
| | • cn  
| | • sAMAccountName  
| | NB: In Active Directory, the 'sAMAccountName' is the 'User Logon Name (pre-Windows 2000)' field. The User Logon Name field is referenced by 'cn'.  
| User Name RDN Attribute | The RDN (relative distinguished name) to use when loading the username. The DN for each LDAP entry is composed of two parts: the RDN and the location within the LDAP directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure. Example:  
| | • cn  
| User First Name Attribute | The attribute field to use when loading the user’s first name. Example:  
| | • givenName  
| User Last Name Attribute | The attribute field to use when loading the user’s last name. Example:  
| | • sn  
| User Display Name Attribute | The attribute field to use when loading the user’s full name. Example:  
| | • displayName  
| User Email Attribute | The attribute field to use when loading the user’s email address. Example:  
| | • mail  
| User Password Attribute | The attribute field to use when loading a user's password. Example:  
| | • unicodePwd  

**Group schema settings**
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Object Class</td>
<td>This is the name of the class used for the LDAP group object. Examples:</td>
</tr>
<tr>
<td></td>
<td>• groupOfUniqueNames</td>
</tr>
<tr>
<td></td>
<td>• group</td>
</tr>
<tr>
<td>Group Object Filter</td>
<td>The filter to use when searching group objects. Example:</td>
</tr>
<tr>
<td></td>
<td>• (objectCategory=Group)</td>
</tr>
<tr>
<td>Group Name Attribute</td>
<td>The attribute field to use when loading the group's name. Example:</td>
</tr>
<tr>
<td></td>
<td>• cn</td>
</tr>
<tr>
<td>Group Description Attribute</td>
<td>The attribute field to use when loading the group's description. Example:</td>
</tr>
<tr>
<td></td>
<td>• description</td>
</tr>
</tbody>
</table>

**Membership schema settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Members Attribute</td>
<td>The attribute field to use when loading the group's members. Example:</td>
</tr>
<tr>
<td></td>
<td>• member</td>
</tr>
<tr>
<td>User Membership Attribute</td>
<td>The attribute field to use when loading the user's groups. Example:</td>
</tr>
<tr>
<td></td>
<td>•memberOf</td>
</tr>
</tbody>
</table>

**Use the User Membership Attribute, when finding the user's group membership**

Put a tick in the checkbox if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.)

- If this checkbox is ticked, your application will use the group membership attribute on the user when retrieving the list of groups to which a given user belongs. This will result in a more efficient retrieval.
- If this checkbox is not ticked, your application will use the members attribute on the group ('member' by default) for the search.
- If the 'Enable Nested Groups' checkbox is ticked, your application will ignore the 'Use the User Membership Attribute' option and will use the members attribute on the group for the search.
Use the User Membership Attribute, when finding the members of a group

Put a tick in the checkbox if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.)

- If this checkbox is ticked, your application will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient search.
- If this checkbox is not ticked, your application will use the members attribute on the group ('member' by default) for the search.

Configuring delegated LDAP authentication

You can configure Stash to use an LDAP directory for delegated authentication, while still using the internal directory for user and group management. There is an option to create users in the internal directory automatically when they attempt to log in, as described in the settings section below.

To connect Stash to an LDAP directory for delegated authentication:

1. Log in as a user with 'Admin' permission.
2. Click Administration in the top menu.
3. Choose Accounts > User Directories.
4. Click Add Directory and select Internal with LDAP Authentication as the directory type.
5. Configure the directory settings, as described in the tables 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 'User Directories' screen. The directory order has the following effects:
   - 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.

On this page:

- Server settings
  - Copying users on login
- LDAP schema
- Advanced settings
- User schema settings
- Group schema settings
- Membership schema settings

Server settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A descriptive name that will help you to identify the directory. Examples:</td>
</tr>
<tr>
<td></td>
<td>- Internal directory with LDAP Authentication</td>
</tr>
<tr>
<td></td>
<td>- Corporate LDAP for Authentication Only</td>
</tr>
</tbody>
</table>
### Directory Type
Select the type of LDAP directory that you will connect to. If you are adding a new LDAP connection, the value you select here will determine the default values for some of the options on the rest of screen. Examples:
- Microsoft Active Directory
- OpenDS
- And more.

### Hostname
The host name of your directory server. Examples:
- ad.example.com
- ldap.example.com
- opens.example.com

### Port
The port on which your directory server is listening. Examples:
- 389
- 10389
- 636 (for example, for SSL)

### Use SSL
Select this check box if the connection to the directory server is an SSL (Secure Sockets Layer) connection. Note that you will need to configure an SSL certificate in order to use this setting.

### Username
The distinguished name of the user that the application will use when connecting to the directory server. Examples:
- cn=administrator,cn=users,dc=ad,dc=example,dc=com
- cn=user,dc=domain,dc=name
- user@domain.name

### Password
The password of the user specified above.

### Copying users on login

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

Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
| **Copy User on Login** | This option affects what will happen when a user attempts to log in. If this check box is selected, the user will be created automatically in the internal directory that is using LDAP for authentication when the user first logs in and their details will be synchronised on each subsequent log in. If this check box is not selected, the user’s login will fail. If you select this check box the following additional fields will appear on the screen, which are described in more detail below:  
- Default Group Memberships  
- Synchronise Group Memberships  
- User Schema Settings (described in a separate section below) |
| **Default Group Memberships** | This field appears if you select the **Copy User on Login** check box. If you would like users to be automatically added to a group or groups, enter the group name(s) here. To specify more than one group, separate the group names with commas. Each time a user logs in, their group memberships will be checked. If the user does not belong to the specified group(s), their username will be added to the group(s). If a group does not yet exist, it will be added to the internal directory that is using LDAP for authentication. Please note that there is no validation of the group names. If you mis-type the group name, authorisation failures will result – users will not be able to access the applications or functionality based on the intended group name. Examples:  
- confluence-users  
- confluence-users,jira-users,jira-developers |
| **Synchronise Group Memberships** | This field appears if you select the **Copy User on Login** check box. If this check box is selected, group memberships specified on your LDAP server will be synchronised with Confluence each time the user logs in. If you select this check box the following additional fields will appear on the screen, both described in more detail below:  
- Group Schema Settings (described in a separate section below)  
- Membership Schema Settings (described in a separate section below) |
### LDAP schema

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base DN</strong></td>
<td>The root distinguished name (DN) to use when running queries against the directory server. Examples:</td>
</tr>
<tr>
<td></td>
<td>- o=example,c=com</td>
</tr>
<tr>
<td></td>
<td>- cn=users,dc=ad,dc=example,dc=com</td>
</tr>
<tr>
<td></td>
<td>- For Microsoft Active Directory, specify the base DN in the following format: <code>dc=domain1,dc=local</code>. You will need to replace the <code>domain1</code> and <code>local</code> or your specific configuration. Microsoft Server provides a tool called ldp.exe which is useful for finding out and configuring the the LDAP structure of your server.</td>
</tr>
<tr>
<td><strong>User Name Attribute</strong></td>
<td>The attribute field to use when loading the username. Examples:</td>
</tr>
<tr>
<td></td>
<td>- cn</td>
</tr>
<tr>
<td></td>
<td>- sAMAccountName</td>
</tr>
</tbody>
</table>

### Advanced settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use Paged Results</strong></td>
<td>Enable or disable the use of the LDAP control extension for simple paging of search results. If paging is enabled, the search will retrieve sets of data rather than all of the search results at once. Enter the desired page size – that is, the maximum number of search results to be returned per page when paged results are enabled. The default is 1000 results.</td>
</tr>
<tr>
<td><strong>Follow Referrals</strong></td>
<td>Choose whether to allow the directory server to redirect requests to other servers. This option uses the node referral (JNDI lookup <code>java.naming.referral</code>) configuration setting. It is generally needed for Active Directory servers configured without proper DNS, to prevent a <code>javax.naming.PartialResultException: Unprocessed Continuation Reference(s)</code> error.</td>
</tr>
</tbody>
</table>

### User schema settings

Note: this section is only visible when **Copy User on Login** is enabled.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
### Additional User DN
This value is used in addition to the base DN when searching and loading users. If no value is supplied, the subtree search will start from the base DN.
Example:
- `ou=Users`

### User Object Class
This is the name of the class used for the LDAP user object. Example:
- `user`

### User Object Filter
The filter to use when searching user objects. Example:
- `(&(objectCategory=Person)(sAMAccountName=*))`

### User Name RDN Attribute
The RDN (relative distinguished name) to use when loading the username. The DN for each LDAP entry is composed of two parts: the RDN and the location within the LDAP directory where the record resides. The RDN is the portion of your DN that is not related to the directory tree structure. Example:
- `cn`

### User First Name Attribute
The attribute field to use when loading the user’s first name. Example:
- `givenName`

### User Last Name Attribute
The attribute field to use when loading the user’s last name. Example:
- `sn`

### User Display Name Attribute
The attribute field to use when loading the user’s full name. Example:
- `displayName`

### User Email Attribute
The attribute field to use when loading the user’s email address. Example:
- `mail`

### Group schema settings
Note: this section is only visible when both Copy User on Login and Synchronise Group Memberships are enabled.

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

---

*Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.*
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Additional Group DN**         | This value is used in addition to the base DN when searching and loading groups. If no value is supplied, the subtree search will start from the base DN. Example:  
• ou=Groups                                                                 |
| **Group Object Class**          | This is the name of the class used for the LDAP group object. Examples:  
• groupOfUniqueNames  
• group                                                                 |
| **Group Object Filter**         | The filter to use when searching group objects. Example:  
• (objectCategory=Group)                                                                 |
| **Group Name Attribute**        | The attribute field to use when loading the group's name. Example:  
• cn                                                                 |
| **Group Description Attribute** | The attribute field to use when loading the group's description. Example:  
• description                                                                 |

**Membership schema settings**

Note: this section is only visible when both Copy User on Login and Synchronise Group Memberships are enabled.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Group Members Attribute**                      | The attribute field to use when loading the group's members. Example:  
• member                                                                 |
| **User Membership Attribute**                    | The attribute field to use when loading the user’s groups. Example:  
• memberOf                                                                 |
| Use the User Membership Attribute, when finding the user’s group membership | Select the check box if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.)  
• If this check box is selected, your application will use the group membership attribute on the user when retrieving the members of a given group. This will result in a more efficient retrieval.  
• If this check box is not selected, your application will use the members attribute on the group ('member' by default) for the search. |
Connecting to Crowd

You can configure Stash to use Atlassian Crowd for user and group management, and for authentication.

Atlassian Crowd is an application security framework that handles authentication and authorisation for your web-based applications. With Crowd you can integrate multiple web applications and user directories, with support for single sign-on (SSO) and centralised identity management. See the Crowd Administration Guide.

Connect to Crowd if you want to use Crowd to manage existing users and groups in multiple directory types, or if you have users of other web-based applications.

**On this page:**
- Server settings
- Crowd permissions
- Advanced settings

To connect Stash to Crowd:

1. Log in as a user with 'Admin' permission.
2. Click Administration in the top menu.
3. Choose Accounts > User Directories.
4. Click Add Directory and select Atlassian Crowd.
5. Enter settings, as described below.
6. Test and save the directory settings.
7. Define the directory order, on the 'User Directories' screen, by clicking the blue up- and down-arrows next to each directory. The directory order has the following effects:
   - 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.

### Server settings

<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>
<tr>
<td>Server URL</td>
<td>The web address of your Crowd console server. Examples:</td>
</tr>
<tr>
<td></td>
<td>• <a href="http://www.example.com:8095/crowd/">http://www.example.com:8095/crowd/</a></td>
</tr>
<tr>
<td></td>
<td>• <a href="http://crowd.example.com">http://crowd.example.com</a></td>
</tr>
<tr>
<td>Application Name</td>
<td>The name of your application, as recognised 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.</td>
</tr>
</tbody>
</table>
### 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](#).

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>Enable or disable support for nested groups. Before enabling nested groups, please check to see if the user directory or directories in Crowd support nested groups. When nested groups are enabled, you can define a group as 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.</td>
</tr>
<tr>
<td>Synchronisation Interval (minutes)</td>
<td>Synchronisation 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>

### Crowd permissions

Stash offers **Read Only** permissions for Crowd directories. The users, groups and memberships in Crowd directories are retrieved from Crowd and can only be modified from Crowd. You cannot modify Crowd users, groups or memberships using the Stash administration screens.

For local Stash directories, **Read Only** and **Read/Write** permissions are available.

### Advanced settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Nested Groups</td>
<td>Enable or disable support for nested groups. Before enabling nested groups, please check to see if the user directory or directories in Crowd support nested groups. When nested groups are enabled, you can define a group as 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.</td>
</tr>
<tr>
<td>Synchronisation Interval (minutes)</td>
<td>Synchronisation 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>

### Connecting to JIRA for user management

You can connect Stash to an Atlassian JIRA instance to delegate Stash user and group management, and authentication.

Choose this option as an alternative to Atlassian Crowd, for simple configurations with a limited number of users. Note that Stash can only connect to a JIRA server running JIRA 4.3 or later.

Connecting Stash and JIRA is a 3-step process:

1. Set up JIRA to allow connections from Stash.
2. Set up Stash to connect to JIRA.
3. Set up Stash users and groups in JIRA.

You can connect to JIRA either when you first run Stash, using the Setup Wizard, or at any time after setup is complete.

If using the Stash Setup Wizard to configure JIRA integration, we recommend that you make use of the automatic back-linking from JIRA to Stash.

⚠️ You need to be an administrator in both JIRA and Stash to do this.
Connecting to JIRA

1. Set up JIRA to allow connections from Stash

1. Log in as a user with the 'JIRA Administrators' global permission.
2. For JIRA 4.3.x, select Other Application from the 'Users, Groups & Roles' section of the 'Administration' menu.
   For JIRA 4.4 or later, choose Administration > Users > JIRA User Server.
3. Click Add Application.
4. Enter the application name and password that Stash will use when accessing JIRA.
5. Enter the IP address of your Stash server. Valid 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.
6. Click Save.
7. Define the directory order, on the 'User Directories' screen, by clicking the blue up- and down-arrows next to each directory. The directory order has the following effects:
   - 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.

2. Set up Stash to connect to JIRA

1. Log in as a user with 'Admin' permission.
2. Click Administration in the top menu and then User Directories.
3. Click Add Directory and select Atlassian JIRA.
4. Enter settings, as described below.
5. Test and save the directory settings.
6. Define the directory order, on the 'User Directories' screen, by clicking the blue up- and down-arrows next to each directory. The directory order has the following effects:
   - 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.

3. Set up Stash users and groups in JIRA

In order to use Stash, users must be a member of the Stash-users group or have Stash 'can use' permission. Follow these steps to configure your Stash groups in JIRA:

1. Add the stash-users and stash-administrators groups in JIRA.
2. Add your own username as a member of both of the above groups.
3. Choose one of the following methods to give your existing JIRA users access to Stash:
   - Option 1: In JIRA, find the groups that the relevant users belong to. Add the groups as members of one or both of the above Stash groups.
- Option 2: Log in to Stash using your JIRA account, click **Administration** and then **Global Permissions**. Assign the appropriate permissions to the relevant JIRA groups.

## Server settings

<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 amongst your list of directory servers. Examples:</td>
</tr>
<tr>
<td></td>
<td>• JIRA 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 name used by your application when accessing the JIRA server that acts as user manager. Note that you will also need to define your application to that JIRA server, via the 'Other Applications' option in the 'Users, Groups &amp; Roles' section of the 'Administration' menu.</td>
</tr>
<tr>
<td>Application Password</td>
<td>The password used by your application when accessing the JIRA server that acts as user manager.</td>
</tr>
</tbody>
</table>

## JIRA server permissions

<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>
<tr>
<td>Read/Write</td>
<td>The users, groups and memberships in this directory are retrieved from the JIRA server that is acting as user manager. When you modify a user, group or membership, the changes will be applied directly to your application and to the JIRA server that is acting as user manager.</td>
</tr>
</tbody>
</table>

## Advanced settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
Enable Nested Groups

Enable or disable support for nested groups. Before enabling nested groups, please check to see if nested groups are enabled on the JIRA server that is acting as user manager. When nested groups are enabled, you can define a group as 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.

Enable Incremental Synchronisation

Enable or disable incremental synchronisation. Only changes since the last synchronisation will be retrieved when synchronising a directory.

Synchronisation Interval (minutes)

Synchronisation 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.

Setting up your mail server

Setting up your mail server will allow you to:

- Send emails when creating a user so that they can generate their own password.
- Allow your users to reset their password if they forget it.

Go to the 'Mail server configuration' page in the Administration.

Fill in the form and click Save.
Specifying the base URL for Stash

This is the base URL for this installation of Stash. All links created, for example in Stash email notifications, will be prefixed by this URL.

To specify Stash’s base URL:

1. Click Administration in the top menu bar.
2. Click Server settings (under ‘Settings’).
3. In the Base URL field, type the URL address of your Stash server (for example, "https://stash.your_org.com").
4. Click Save.

Related pages:

- Administering Stash

Connecting Stash to an external database

This page provides information about using Stash with an external database.
Stash ships with an embedded database that it uses straight out-of-the-box, with no configuration required. This is great for evaluation purposes, but for production installations we recommend that you use one of the supported external databases.

Stash supports the following external databases:

- MySQL
- Oracle
- PostgreSQL
- SQL Server

Please refer to Supported platforms for the versions of external databases supported by Stash.

You can configure Stash to use an external database using:

- The Setup Wizard (when you install Stash); or
- The Database Migration Wizard in a running instance of Stash.

On this page:

- Why would I want to use an external database?
- Using the Setup Wizard to configure Stash to use an external database
- Using the Database Migration Wizard
- Notes about database migration

Related pages:

- Connecting Stash to MySQL
- Connecting Stash to PostgreSQL
- Connecting Stash to Oracle
- Connecting Stash to SQL Server
- How do I change the external database password

Why would I want to use an external database?

Stash ships with an embedded database that is great for evaluation purposes, but for production installations we recommend that you make use of one of the supported external databases, for the following reasons:

- **Improved protection against data loss**: The Stash built-in database, which runs HSQLDB, is susceptible to data loss during system crashes. External databases are generally more resistant to data loss during a system crash. HSQLDB is not supported in production environments and should only be used for evaluation purposes.
- **Performance and scalability**: If you have a large number of users on your Stash instance, running the database on the same server as Stash may slow it down. When using the embedded database, the database will always be hosted and run on the same server as Stash, which will limit performance.
- **Unified back-up**: Use your existing DBMS tools to back up your Stash database alongside your organisation's other databases.

Using the Setup Wizard to configure Stash to use an external database

You can connect Stash to an external database when you run the Setup Wizard, immediately after starting Stash for the first time. See Installing Stash on Windows or Installing Stash on Linux and Mac for more information.

If you have been using Stash for a while while using the embedded database, you can migrate to an external database, as described below.
Using the Database Migration Wizard

You can use the Database Migration Wizard to migrate Stash from the embedded database to an external database, or from an external database to another external database.

1. Log in to Stash
2. Visit Administration > Database (under 'Settings').
3. Click Migrate database and follow the instructions for running the migration.

Notes about database migration

- **Back up the database and Stash home directory:**
  Before starting the database migration process you should back up your Stash home directory. If you intend to migrate from one external database to another, you should also backup the existing database before proceeding. See Data recovery and backups for more information.

- **Stash unavailable during the migration:**
  Stash will not be available to users during the database migration operation. In addition, running the migration when people are using Stash can sometimes cause the migration to time out waiting for all activity in Stash that uses the database to complete. For these reasons we recommend that you run the database migration outside of normal usage periods.

- **Migration will usually take less that 30 minutes:**
  The duration of the migration process depends on the amount of data in the Stash database being migrated. For new installations of Stash, containing very little data, the migration process typically takes just a few seconds. If you have been using Stash for some time, its database will contain more data, and the migration process will therefore take longer. If Stash has been linked to a JIRA instance, and there are hundreds of thousands of commits in Stash with JIRA keys in the commit messages, the migration may take tens of minutes.

- **We strongly recommend using a new clean database for the new Stash database:**
  In case of a migration failure, Stash may have partially populated the target database. If the target database is new (therefore empty) and set aside for Stash's exclusive use, it's very easy to clean up after a failed migration; just drop the target database and use a clean target database instance for the next attempt.

- **Ensure your Stash home directory is secured against unauthorised access:**
  - After the migration, the connection details (including the username and password) for the database are stored in the stash-config.properties file.
  - Migration will create a dump file of the contents of your database in the Stash home export directory. This is used during the migration and is kept for diagnostic purposes in the case of an error. You may remove this after migration but it may reduce Atlassian Support's ability to help you in the case of migration issues.

  You can edit the database password if needed after migration.

Connecting Stash to MySQL

This page describes how to connect Stash to a MySQL database.

The overall process for using a MySQL database with Stash is:

- Install MySQL where it is accessible to Stash.
- Create a database and user on the MySQL server for Stash to use.
- Install Stash on Windows, or on Linux or Mac.
- Either:
  - at Stash install time, run the Setup Wizard to connect Stash to the MySQL database, or
  - at a later time, migrate Stash to the MySQL database.
It is assumed here that you already have MySQL installed and running. MySQL documentation is available at http://dev.mysql.com/doc/.

See Supported platforms for the versions of MySQL supported by Stash.

On this page:

1. Prerequisites
   - Download and install the JDBC driver
   - Backup your data
   - Create the Stash database

2. Connect Stash to the MySQL database

Related pages:
- Connecting Stash to an external database
- Connecting Stash to Oracle
- Connecting Stash to PostgreSQL
- Connecting Stash to SQL Server

1. Prerequisites

Download and install the JDBC driver

The JDBC drivers for MySQL Enterprise Server are not bundled with Stash (due to licensing restrictions). You need to download and install the driver yourself.

1. Download the MySQL Connector/J JDBC driver from the download site.
2. Expand the downloaded zip/tar.gz file.
3. Copy the mysql-connector-java-5.1.XX-bin.jar file from the extracted directory to the <Stash installation directory>/lib directory (create the lib/ directory if it doesn't already exist).
4. Stop Stash, on Windows, or on Linux and Mac.
5. Restart Stash, on Windows, or on Linux and Mac.

Backup your data

If you are migrating your data from the internal Stash database, back up the Stash home directory.

If you are migrating your Stash data from another external database, back up that database by following the instructions provided by the database vendor before proceeding with these instructions.

See Data recovery and backups.

Create the Stash database

Before you can use Stash with MySQL, you must set up MySQL as follows:

- create a database on MySQL for Stash to use
- create a Stash user on the database
- configure the database to use utf8 character set encoding
- configure the database to use utf8_bin collation (ensures case sensitivity)
- configure the database to use the InnoDB storage engine.

Here is an example of how to do that. When Stash and MySQL run on the same physical computer (accessible through localhost), run the following commands (replacing stashuser and password with your own

Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
This creates an empty MySQL database with the name `stash`, and a user that can log in from the host that Stash is running on who has full access to the newly created database. In particular, the user should be allowed to create and drop tables, indexes and other constraints.

If the MySQL database and Stash servers are on the same physical computer, you can use `localhost` and *not set a password* by omitting `IDENTIFIED BY 'password'` from the 2nd MySQL statement above (if you trust the security *within* this computer).

If the MySQL database and Stash servers are on different computers, just replace the `localhost` part of the `GRANT ALL` statement above with the hostname of the machine that Stash is running on. See the documentation at [http://dev.mysql.com/doc/refman/5.1/en/account-names.html](http://dev.mysql.com/doc/refman/5.1/en/account-names.html).

Note that Stash will generally require about 25–30 connections to the database.

### 2. Connect Stash to the MySQL database

You can now connect Stash to the MySQL database, either:

- when you run the Setup Wizard, at install time,
- when you wish to migrate to MySQL, either from the embedded database or from another external database.

#### When running the Setup Wizard at install time:

1. Select `External` at the ‘Database’ step.
2. Select `MySQL` for `Database Type`.
3. Complete the form. See the table below for details.
4. Click `Next`, and follow the instructions in the Stash Setup Wizard.

#### When migrating to MySQL:

1. Click `Administration` and then `Database` (under ‘Settings’).
2. Click `Migrate database`.
3. Select `MySQL` for `Database Type`.
4. Complete the form. See the table below for details.
5. Click `Start Migration`.

<table>
<thead>
<tr>
<th><strong>Hostname</strong></th>
<th>The host name or IP address of the computer running the database server.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port</strong></td>
<td>The TCP port with which Stash can connect to the database server. The default value is the default port that MySQL runs against. You can change that if you know the port that your MySQL instance is using.</td>
</tr>
<tr>
<td><strong>Database</strong></td>
<td>The name of the database that Stash should connect to.</td>
</tr>
</tbody>
</table>
Connecting Stash to Oracle

This page describes how to connect Stash to a Oracle database.

The overall process for using a Oracle database with Stash is:

- Install Oracle where it is accessible to Stash.
- Create a database and user on the Oracle server for Stash to use.
- Install Stash on Windows, or on Linux or Mac.
- Either:
  - at Stash install time, run the Setup Wizard to connect Stash to the Oracle database, or
  - at a later time, migrate Stash to the Oracle database.

It is assumed here that you already have Oracle installed and running. For information about installing Oracle and creating Oracle databases, see the Oracle documentation pages. For the versions of Oracle supported by Stash see Supported platforms.

On this page:

1. Prerequisites
2. Connect Stash to the Oracle database

Related pages:

- Connecting Stash to an external database
- Connecting Stash to MySQL
- Connecting Stash to PostgreSQL
- Connecting Stash to SQL Server

1. Prerequisites

Backup

If you are migrating your data from the internal Stash database, back up the Stash home directory.
If you are migrating your Stash data from a different external database, back up that database by following the instructions provided by the database vendor before proceeding with these instructions.

See Data recovery and backups.

Create the Oracle schema for Stash

Before you can use Stash with Oracle, you must set up Oracle as follows:

- Ensure that you have a database instance available for Stash (either create a new one or use an existing one)
  The character set of the database must be set to either AL32UTF8 or UTF8, to support storage of Unicode data as per the Oracle documentation.
  Note that it is important to the proper operation of Stash that the database store its data in a case-sensitive manner. By changing the values of the NLS_COMP and/or NLS_SORT variables, it is possible to cause Oracle to perform its searches in a case-insensitive manner. We therefore strongly recommend that those variables be left at their default values.

- Create a user that Stash will connect as (e.g. stash).
  Remember the database user name; it will be used to configure Stash's connection to the database in subsequent steps.
  When you create a user in Oracle, a schema is automatically created.

  It is strongly recommended that you create a new database user for use by Stash rather than sharing one that is used by other applications or people.

- Grant the Stash user privileges to create sessions and tables on the schema that is automatically created for you.

The format of the command to create a user in Oracle is:

```
CREATE USER <user>
  IDENTIFIED BY <password>
  DEFAULT TABLESPACE <tablespace>
  QUOTE UNLIMITED ON <tablespace>;
```

Here is a simple example, using SQL*Plus, of how one might create a user called stash with password jdHyd6Sn21 in tablespace users, and grant the user a minimal set of privileges. When you run the command on your machine, remember to replace the user name, password and tablespace names with your own values.

```
CREATE USER stash
  IDENTIFIED BY jdHyd6Sn21
  DEFAULT TABLESPACE users
  QUOTA UNLIMITED ON users;

GRANT CREATE SESSION, CREATE TABLE, CREATE SEQUENCE, CREATE TRIGGER to stash;
```

This creates an empty Oracle schema with the name stash, and a user that can log in from the host that Stash is running on and who has full access to the newly created schema. In particular, the user is allowed to create sessions and tables.

Note that Stash will generally require about 25–30 connections to the database.

2. Connect Stash to the Oracle database
You can now connect Stash to the Oracle database, either:

- when you run the Setup Wizard, at install time,
- when you wish to migrate to Oracle, either from the embedded Stash database or from another external database.

When running the Setup Wizard at install time:

1. Select **External** at the 'Database' step.
2. Select **Oracle** for **Database Type**.
3. Complete the form. See the table below for details.
4. Click **Next**, and follow the instructions in the Stash Setup Wizard.

When migrating to Oracle:

1. Click **Administration** and then **Database** (under 'Settings').
2. Click **Migrate database**.
3. Select **Oracle** for **Database Type**.
4. Complete the form. See the table below for details.
5. Click **Start Migration**.

<table>
<thead>
<tr>
<th>Hostname</th>
<th>The host name or IP address of the computer running the Oracle server.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port</strong></td>
<td>The TCP port with which Stash can connect to the database server. The default value is the default port that Oracle runs against. You can change that if you know the port that your Oracle instance is using.</td>
</tr>
<tr>
<td><strong>SID</strong></td>
<td>The system identifier of the Oracle instance that Stash should connect to.</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>The username that Stash should use to access the database.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password that Stash should use to access the database.</td>
</tr>
</tbody>
</table>

### Connecting Stash to PostgreSQL

This page describes how to connect Stash to a PostgreSQL database.

The overall process for using a PostgreSQL database with Stash is:
1. Prerequisites

Backup

If you are migrating your Stash data from the HSQL internal database, back up the Stash home directory.

If you are migrating your Stash data from another external database, back up that database by following the instructions provided by the database vendor before proceeding with these instructions.

See Data recovery and backups.

Create the Stash database

Before you can use Stash with PostgreSQL, you must:

- Create a role for Stash to use when it connects to the database. We strongly recommend that this role be established for Stash's use exclusively; it should not be shared by other applications or people.

- Create a database in which Stash can store its data. The database must be configured to use the UTF-8 character set. During normal operation, Stash will acquire 25–30 connections to the database.

Here is an example of how to create a user called stashuser with password jellyfish, and a database called stash, which is configured for use by stashuser. Using a PostgreSQL client application like psql or pgAdmin, run the following commands, replacing the user name, password, and database name with your own values.

Related pages:
- Connecting Stash to an external database
- Connecting Stash to MySQL
- Connecting Stash to Oracle
- Connecting Stash to SQL Server
2. Connect Stash to the PostgreSQL database

You can now connect Stash to the PostgreSQL database, either:

- when you run the Setup Wizard, at install time,
- when you wish to migrate Stash to PostgreSQL, either from the embedded HSQL database or from another external database.

When running the Setup Wizard at install time:

1. Select External at the 'Database' step.
2. Select PostgreSQL for Database Type.
3. Complete the form. See the table below for details.
4. Click Next, and follow the instructions in the Stash Setup Wizard.

When migrating to PostgreSQL:

1. Click Administration and then Database (under 'Settings').
2. Click Migrate database.
3. Select PostgreSQL for Database Type.
4. Complete the form. See the table below for details.
5. Click Start Migration.

<table>
<thead>
<tr>
<th>Hostname</th>
<th>The host name or IP address of the computer running the database server.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>The TCP port with which Stash can connect to the database server. The default value is the default port that PostgreSQL runs against. You can change that if you know the port that your PostgreSQL instance is using.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that Stash should connect to.</td>
</tr>
<tr>
<td>Username</td>
<td>The username that Stash should use to access the database.</td>
</tr>
<tr>
<td>Password</td>
<td>The password that Stash should use to access the database.</td>
</tr>
</tbody>
</table>
Connecting Stash to SQL Server

This page describes how to connect Stash to a Microsoft SQL Server database.

The overall process for using a SQL Server database with Stash is:

- Install SQL Server where it is accessible to Stash.
- Create a database and user on the SQL Server server for Stash to use.
- Install Stash on Windows, or on Linux or Mac.
- Either:
  - at Stash install time, run the Setup Wizard to connect Stash to the SQL Server database, or
  - at a later time, migrate Stash to the SQL Server database.


See [Supported platforms](#) for the versions of SQL Server supported by Stash.

### On this page:

1. Prerequisites
2. Connect Stash to the SQL Server database

### Related pages:

- [Connecting Stash to an external database](#)
- [Connecting Stash to MySQL](#)
- [Connecting Stash to Oracle](#)
- [Connecting Stash to PostgreSQL](#)

### 1. Prerequisites

**Back up your current database**

If you are migrating your data from the internal Stash database, back up the [Stash home directory](#).

If you are migrating your Stash data from a different external database, back up that database by following the instructions provided by the database vendor before proceeding with these instructions.

See [Data recovery and backups](#).
Create the SQL Server database

Before you can use Stash with SQL Server, you must set up SQL Server as follows:

<table>
<thead>
<tr>
<th>Step</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a database</td>
<td>e.g. stash. Remember this database name for the connection step below.</td>
</tr>
<tr>
<td>Set the collation type</td>
<td>This should be case-sensitive, for example, 'SQL_Latin1_General_CP1_CS_AS' (CS = Case Sensitive).</td>
</tr>
<tr>
<td>Create a database user</td>
<td>e.g. stashuser. This database user should not be the database owner, but should be in the db_owner role. See SQL Server Startup Errors. Remember this database user name for the connection step below.</td>
</tr>
<tr>
<td>Set database user permissions</td>
<td>The Stash database user has permission to connect to the database, and to create and drop tables, indexes and other constraints, and insert and delete data, in the newly-created database.</td>
</tr>
<tr>
<td>Enable TCP/IP</td>
<td>Ensure that TCP/IP is enabled on SQL Server and that SQL Server is listening on the correct port (which is 1433 for a default SQL Server installation). Remember this port number for the connection step below.</td>
</tr>
<tr>
<td>Check the authentication mode</td>
<td>Ensure that SQL Server is operating in the appropriate authentication mode. By default, SQL Server operates in 'Windows Authentication Mode'. However, if your user is not associated with a trusted SQL connection, 'Microsoft SQL Server, Error: 18452' is received during Stash startup, and you will need to change the authentication mode to 'Mixed Authentication Mode'.</td>
</tr>
<tr>
<td>Check that SET NOCOUNT is off</td>
<td>Ensure that the SET NOCOUNT option is turned off. You can do that in SQL Server Management Studio as follows:</td>
</tr>
<tr>
<td></td>
<td>1. Navigate to Tools &gt; Options &gt; Query Execution &gt; SQL Server &gt; Advanced. Ensure that the SET NOCOUNT option is cleared.</td>
</tr>
<tr>
<td></td>
<td>2. Now, go to the Server &gt; Properties &gt; Connections &gt; Default Connections properties box and clear the no count option.</td>
</tr>
</tbody>
</table>

Note that Stash will generally require about 25–30 connections to the database.

Here is an example of how to create and configure the SQL Server database from the command line. When Stash and SQL Server run on the same physical computer (accessible through localhost), run the following commands (replacing stashuser and password with your own values):
1. Create a new SQL Server database with the name `stash` and a user that can log in from the host that Stash is running on who has full access to the newly created database. In particular, the user should be allowed to create and drop tables, indexes and other constraints.

2. Connect Stash to the SQL Server database

You can now connect Stash to the SQL Server database, either:

- when you run the Setup Wizard, at install time,
- when you wish to migrate to SQL Server, either from the embedded database or from another external database.

When running the Setup Wizard at install time:

1. Select External at the 'Database' step.
2. Select SQL Server for Database Type.
3. Complete the form. See the table below for details.
4. Click Next, and follow the instructions in the Stash Setup Wizard.

When migrating to SQL Server:

1. Click Administration and then Database (under 'Settings').
2. Click Migrate database.
3. Select SQL Server for Database Type.
4. Complete the form. See the table below for details.
5. Click Start Migration.

<table>
<thead>
<tr>
<th>Hostname</th>
<th>The host name or IP address of the computer running the database server.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>The TCP port with which Stash can connect to the database server. The default value of 1433 is the default port that SQL Server runs against. You can change that if you know the port that your SQL Server instance is using.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that Stash should connect to.</td>
</tr>
</tbody>
</table>
### Username
The username that Stash should use to access the database.

### Password
The password that Stash should use to access the database.

---

**Advanced actions**

This section describes the administrative actions that can be performed from outside of the Stash Administration user interface.

**In this section:**
- Enabling Stash debug logging
- Integrating Stash with Apache HTTP Server
- Scaling Stash
- Securing Stash with HTTPS
- System settings
- Setting up SSH port forwarding
- Data recovery and backups

**Related pages:**
- Administering Stash
- Supported platforms
- Stash FAQ

---

**Data recovery and backups**

Stash does not currently have any built-in data backup or recovery solutions. We **highly recommend** that you establish a data recovery plan that is aligned with your company's policies.

⚠️ **Important**: Stash should be shut down before making any backups, to ensure data consistency.

The simplest and most effective backup solution is to make a backup of the entire Stash home directory.

If you are using Stash with an external database, you will also need to backup your database as well. Please speak to your DBA or refer to your database software backup guidelines.
Enabling Stash debug logging

This page describes how to enable debug level logging in Stash. Stash logs can be found in `<STASH_HOME>/logs`.

When using the standard Stash distribution, logs for the Tomcat webserver that hosts Stash can be found in `<Stash installation directory>/log`.

Enabling debug logging on startup

To enable debug logging whenever Stash is started, edit the `<STASH_HOME>/stash-config.properties` file (if this file doesn't exist then you should create it) and add the following line:

```
logging.logger.ROOT=DEBUG
```

Enabling debug logging at runtime

To enable debug logging for the root logger once Stash has been started, run the following command in your terminal:

```
curl -u <ADMIN_USERNAME> -v -X PUT -d "" -H "Content-Type: application/json" <BASE_URL>/rest/api/latest/logs/rootLogger/debug
```

# e.g.
```
curl -u admin -v -X PUT -d "" -H "Content-Type: application/json"
http://localhost:7990/rest/api/latest/logs/rootLogger/debug
```

To enable debug logging for a specific logger, run the following command in your terminal:

```
curl -u <ADMIN_USERNAME> -v -X PUT -d "" -H "Content-Type: application/json" <BASE_URL>/rest/api/latest/logs/logger/<LOGGER_NAME>/debug
```

# e.g.
```
curl -u admin -v -X PUT -d "" -H "Content-Type: application/json"
```

Integrating Stash with Apache HTTP Server

This page explains how to establish a network topology in which Apache HTTP Server acts as a reverse proxy for Stash. Typically, such a configuration would be used when Stash is installed in a protected zone "behind the firewall", and Apache HTTP Server provides a gateway through which users outside the firewall can access Stash.

Be aware that Stash does not need to run behind a web server, since it is capable of serving web requests.
directly; to secure Stash when run in this way see Securing Stash with HTTPS. Otherwise, if you want to install Stash in an environment that incorporates Apache HTTP Server, this document is for you.

About using Apache software

This section has general information pertaining to the use of Apache HTTP Server and Apache Tomcat. It is important that you read this section before proceeding to the steps that follow.

Configuring Tomcat 6

The Stash distribution includes an instance of Tomcat 6, the configuration of which is determined by the contents of the server.xml file, which can be found in the conf directory immediately under the Stash installation directory. Note that any changes that you make to the server.xml file will be effective upon starting or re-starting Stash.

You may find it helpful to refer to the Apache Tomcat 6.0 Proxy Support HowTo page.

On this page:

- About using Apache software
- Step 1: Configure the Tomcat Connector
- Step 2: Change Stash's base URL
- Step 3 (optional): Set a context path for Stash
- Step 4: Enable mod_proxy and mod_proxy_http in Apache HTTP Server
- Step 5: Configure mod_proxy to map requests to Stash
- Step 6: Configure mod_proxy to disable forward proxying
- Step 7: Allow proxying to Stash from everywhere
- Step 8 (optional): Configure Apache HTTP Server for SSL
- A note about application links
- Troubleshooting

Related pages:

- Securing Stash with HTTPS

Configuring Apache HTTP Server

Since Apache HTTP Server is not an Atlassian product, Atlassian does not guarantee to provide support for its configuration. You should consider the material on this page to be for your information only; use it at your own risk. If you encounter problems with configuring Apache HTTP Server, we recommend that you refer to the Apache HTTP Server Support page.

You may find it helpful to refer to the Apache HTTP Server Documentation, which describes how you can control Apache HTTP Server by changing the contents of the httpd.conf file. The section on Apache Module mod_proxy is particularly relevant. Note that any changes you make to the httpd.conf file will be effective upon starting or re-starting Apache HTTP Server.

This document relates to Apache HTTP Server version 2.4.2; the configuration of other versions may differ.

Step 1: Configure the Tomcat Connector

Find the normal (non-SSL) Connector directive in Tomcat's server.xml file, and add the scheme, proxyN
The **name**, and **proxyPort** attributes as shown below. Instead of mycompany.com, set the **proxyName** attribute to the domain name that Apache HTTP Server will be configured to serve. This informs Stash of the domain name and port of the requests that reach it via Apache HTTP Server, and is important to the correct operation of the Stash functions that construct URL's.

```xml
<Connector port="7990"
    protocol="HTTP/1.1"
    connectionTimeout="20000"
    useBodyEncodingForURI="true"
    redirectPort="8443"
    compression="on"
    compressableMimeType="text/html,text/xml,text/plain,text/css,application/json,application/javascript,application/x-javascript"
    scheme="http"
    proxyName="mycompany.com"
    proxyPort="80" />
```

**Note:** Apache HTTP Server’s **ProxyPreserveHost** directive is another way to have the hostname of the incoming request recognised by Stash instead of the hostname at which Stash is actually running. However, the **ProxyPreserveHost** directive does not cause the scheme to be properly set. Since we have to mess with **ProxyPreserveHost** anyway, we recommend that you stick with the above-described approach, and don't bother to set the **ProxyPreserveHost** in Apache HTTP Server.

For more information about configuring the Tomcat Connector, refer to the [Apache Tomcat 6.0 HTTP Connector Reference](#).

---

**Step 2: Change Stash’s base URL**

After re-starting Stash, open a browser window and log into Stash using an administrator account. Visit *Administration > Settings > General Settings*, and change the base URL to match the proxy URL (the URL that Apache HTTP Server will be serving).

**Step 3 (optional): Set a context path for Stash**

By default, Stash is configured to run with an empty context path; in other words, from the 'root' of the server's name space. In that default configuration, Stash is accessed at:

```
http://localhost:7990/
```

It's perfectly fine to run Stash with the empty context path as above. Alternatively, you can set a context path by changing the **Context** directive in Tomcat's server.xml file:

```xml
    <Context path="/stash" docBase="${catalina.home}/atlassian-stash.war"
    reloadable="false" useHttpOnly="true">
    ....
    </Context>
```

If you do set a context path, it is important that the same path be used in **Step 5**, when setting up the **ProxyPass** and **ProxyPassReverse** directives. You should also append the context path to Stash's base URL (see **Step 2**).
Step 4: Enable mod_proxy and mod_proxy_http in Apache HTTP Server

In the mod_proxy documentation, you will read that mod_proxy can be used as a forward proxy, or as a reverse proxy (gateway); you want the latter. Where the mod_proxy documentation mentions 'origin server', it refers to your Stash server. Unless you have a good reason for doing otherwise, load mod_proxy and mod_proxy_http dynamically, using the LoadModule directive; that means un-commenting the following lines in the httpd.conf file:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

Experienced administrators may be aware of the Apache Connector module, mod_jk. Atlassian does not recommend use of the mod_jk module with Stash, since it has proven itself to be less reliable than mod_proxy.

Step 5: Configure mod_proxy to map requests to Stash

To configure mod_proxy for use with Stash, you need to use the ProxyPass and ProxyPassReverse directives in Apache HTTP Server's httpd.conf file as follows:

```
ProxyPass / http://localhost:7990/ connectiontimeout=5 timeout=300
ProxyPassReverse / http://localhost:7990/
```

Suppose Apache HTTP Server is configured to serve the mycompany.com domain; then the above directives tell Apache HTTP Server to forward web requests of the form http://mycompany.com/* to the Tomcat connector (Stash) running on port 7990 on the same machine.

The connectiontimeout attribute specifies the number of seconds Apache HTTP Server waits for the creation of a connection to Stash.

The timeout attribute specifies the number of seconds Apache HTTP Server waits for data to be sent to Stash.

If you set up a context path for Stash in Step 3, you'll need to use that context path in your ProxyPass and ProxyPassReverse directives. Suppose your context path is set to "/stash", the directives would be as follows:

```
ProxyPass /stash http://localhost:7990/stash connectiontimeout=5 timeout=300
ProxyPassReverse /stash http://localhost:7990/stash
```

If Stash is to run on a different domain and/or different port, you should use that domain and/or port number in the ProxyPass and ProxyPassReverse directives; for example, suppose that Stash will run on port 9900 on private.mycompany.com under the context path /stash, then you would use the following directives:

```
ProxyPass /stash http://private.mycompany.com:9900/stash connectiontimeout=5 timeout=300
```
Step 6: Configure mod_proxy to disable forward proxying

If you are using Apache HTTP Server as a reverse proxy only, and not as a forward proxy server, you should turn forward proxying off by including a ProxyRequests directive in the httpd.conf file, as follows:

```
ProxyRequests Off
```

Step 7: Allow proxying to Stash from everywhere

Strictly speaking, this step is unnecessary because access to proxied resources is unrestricted by default. Nevertheless, we explicitly allow access to Stash from any host so that this policy will be applied regardless of any subsequent changes to access controls at the global level. Use the Proxy directive in the httpd.conf file as follows:

```
<Proxy *>
    Order Deny,Allow
    Allow from all
</Proxy>
```

The Proxy directive provides a context for the directives that are contained within its delimiting tags. In this case, we specify a wild-card url (the asterisk), which applies the two contained directives to all proxied requests.

The Order directive controls the order in which any Allow and Deny directives are applied. In the above configuration, we specify "Deny, Allow", which tells Apache HTTP Server to apply any Deny directives first, and if any match, the request is denied unless it also matches an Allow directive. In fact, "Deny, Allow" is the default; we include it merely for the sake of clarity. Note that we specify one Allow directive, which is described below, and don't specify any Deny directives.

The Allow directive, in this context, controls which hosts can access Stash via Apache HTTP Server. Here, we specify that all hosts are allowed access to Stash.

Step 8 (optional): Configure Apache HTTP Server for SSL

If you want to set up SSL access to Stash, take steps 7(a) to 7(d) below. When you are finished, users will be able to make secure connections to Apache HTTP Server; connections between Apache HTTP Server and Stash will remain unsecured (not using SSL). If you don't want to set up SSL access, you can skip this section entirely.

Note: It would be possible to set up an SSL connection between Apache HTTP Server and Tomcat (Stash), but that configuration is very unusual, and not recommended in most circumstances.

Step 8(a): Configure the Tomcat Connector for SSL

Find the normal (non-SSL) Connector directive in Tomcat's server.xml file, and change the redirectPort, scheme and proxyPort attributes as follows:
The `redirectPort` directive causes Tomcat-initiated redirections to secured resources to use the specified port. Right now, the Stash configuration of Tomcat does not involve Tomcat-initiated redirections, so the change to `redirectPort` is redundant. Nevertheless, we suggest that you change it as directed above for the sake of completeness.

**Step 8(b): Set up a virtual host in Apache HTTP Server**

Un-comment the following LoadModule directive in Apache HTTP Server's `httpd.conf` file:

```
LoadModule ssl_module modules/mod_ssl.so
```

Add the following directives to the `httpd.conf` file:

```
Listen 443
<VirtualHost *:443>
  SSLEngine On
  SSLCertificateFile    "/usr/local/apache2/conf/server.crt"
  SSLCertificateKeyFile "/usr/local/apache2/conf/server.key"
  ProxyPass        / http://localhost:7990/ connectiontimeout=5 timeout=300
  ProxyPassReverse / http://localhost:7990/
</VirtualHost>
```

The `Listen` directive instructs Apache HTTP Server to listen for incoming requests on port 443. Actually, we could omit that directive in this case, since Apache HTTP Server listens for `https` requests on port 443 by default. Nevertheless, it's good to make one's intentions explicit.

The `VirtualHost` directive encloses a number of child directives that apply only and always to requests that arrive at port 443. Since our `VirtualHost` block does not include a `ServerName` directive, it inherits the server name from the main server configuration.

The `SSLEngine` directive toggles the use of the SSL/TLS Protocol Engine. In this case, we're using it to turn SSL on for all requests that arrive at port 443.

The `SSLCertificateFile` directive tells Apache HTTP Server where to find the PEM-encoded certificate file for the server.

The `SSLCertificateKeyFile` directive tells Apache HTTP Server where to find the PEM-encoded private
key file corresponding to the certificate file identified by the SSLCertificateFile directive. Depending on
how the certificate file was generated, it may contain a RSA or DSA private key file, making the SSLCertificateFile directive redundant; however, Apache strongly discourages that practice. The recommended approach is to separate the certificate and the private key. If the private key is encrypted, Apache HTTP Server will require a pass phrase to be entered when it starts up.

The ProxyPass and ProxyPassReverse directives should be set up in manner described in Step 5.

For more information about the support for SSL in Apache HTTP Server, refer to the Apache SSL/TLS Encryption manual. In addition, you will find lots of relevant information in the <apache directory>/conf/extra/httpd-ssl.conf file, which is included in the standard Apache distribution.

Step 8(c): Create SSL certificate and key files

In Step 8(b), you specified server.crt and server.key as the certificate file and private key file respectively. Those two files must be created before we can proceed. This step assumes that OpenSSL is installed on your server.

Generate a server key file:

```
openssl genrsa -des3 -out server.key 1024
```

You will be asked to provide a password. Make sure that the password is strong because it will form the one real entry point into the SSL encryption set-up. Make a note of the password because you'll need it when starting Apache HTTP Server later.

Generate a certificate request file (server.csr):

```
openssl req -new -key server.key -out server.csr
```

Generate a self-signed certificate (server.crt):

```
openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

The above command generates a self-signed certificate that is valid for one year. You can use the certificate signing request to purchase a certificate from a certificate authority. For testing purposes though, the self-signed certificate will suffice. Copy the certificate file and private key file to the locations you specified in Step 8(b).

```
cp server.key /usr/local/apache2/conf/
cp server.crt /usr/local/apache2/conf/
```

Step 8(d): Update the base URL for 'https'

Open a browser window and log into Stash using an administrator account. Visit Administration > Settings > General Settings, and change the base URL use 'https'.

Using a self-signed certificate
There are two implications of using the self-signed certificate:

- When you access Stash in a web browser, you can expect a warning to appear, alerting you that an un-trusted certificate is in use. Before proceeding you will have to indicate to the browser that you trust the certificate.
- When you perform a git clone operation, SSL verification will fail.

**The SSL verification error message will look something like this:**

```plaintext
error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed while accessing https://justme@mycompany/git/TP/test.git
```

It's easy to fix. Turn SSL verification off for individual git operations by setting the `GIT_SSL_NO_VERIFY` environment variable. In Unix, you can set the variable in-line with git commands as follows:

```bash
GIT_SSL_NO_VERIFY=true git clone https://justme@mycompany/git/TP/test.git
```

In Windows you have to set the variable in a separate shell statement:

```bash
set GIT_SSL_NO_VERIFY=true
git clone https://justme@mycompany/git/TP/test.git
```

Once you have purchased and installed a signed certificate from a certificate authority, you will no longer have to include the `GIT_SSL_NO_VERIFY` modifier.

**A note about application links**

When an application link is established between Stash and another Atlassian product (e.g. JIRA), and Stash is operating 'behind' Apache HTTP Server, the link from the other product to Stash must be via the proxy URL; that is, the 'reciprocal URL' from, say JIRA, to Stash must comport with the proxy name and port that you set at Step 1.

**Troubleshooting**

- On **Fedora Core 4**, people have reported 'permission denied' errors when trying to get mod_proxy (and mod_jk) working. Disabling SELinux (/etc/selinux/config) apparently fixes this.
- Some users have reported problems with user sessions being hijacked when the `mod_cache` module is enabled. If you have such problems, disable the `mod_cache` module. Note that this module is enabled by default in some Apache HTTP Server version 2 distributions.
- In general, if you are having problems:
  1. Ensure that Stash works as expected when running directly from Tomcat on `http://localhost:7990/stash`.
  2. Watch the log files (usually in /var/log/httpd/ or /var/log/apache2/). Check that you have a `LogLevel debug` directive in your httpd.conf, and turn up logging (`'LogLevel debug'`) to get more info.
  3. Check out the Stash Knowledge Base.
Scaling Stash

Hardware requirements

The type of hardware you require to run Stash depends on a number of factors:

- The number of users.
- The size of your repositories. On large repositories, many operations require more memory and more CPUs.
- The frequency of clone operations. Cloning a repository is one of the most demanding operations. One major source of clone operations is continuous integration. When your CI builds involve multiple parallel stages, Stash will be asked to perform multiple clones concurrently, putting significant load on your system.

The following are rough guidelines for choosing your hardware:

- Estimate the number of concurrent clones that are expected to happen regularly (look at continuous integration). Add one CPU for every 2 concurrent clone operations.
- Estimate or calculate the average repository size and allocate 1.5 x number of concurrent clone operations x min(repository size, 700MB) of memory.

Understanding Stash’s resource usage

Most of the things you do in Stash involve both the Stash server and one or more Git processes created by Stash. For instance, when you view a file in the Stash web application, Stash processes the incoming request, performs permission checks, creates a Git process to retrieve the file contents and formats the resulting webpage. In serving most pages, both the Stash server and Git processes are involved. The same is true for the ‘hosting’ operations: pushing your commits to Stash, cloning a repository from Stash or fetching the latest changes from Stash.

As a result, when configuring Stash for performance, CPU and memory consumption for both Stash and Git should be taken into account.
Memory

When deciding on how much memory to allocate for Stash, the most important factor to consider is the amount of memory required for Git. Some Git operations are fairly expensive in terms of memory consumption, most notably the initial push of a large repository to Stash and cloning large repositories from Stash. For large repositories, it is not uncommon for Git to use up to 500 MB of memory during the clone process. The numbers vary from repository to repository, but as a rule of thumb 1.5 x the repository size on disk (contents of the .git/objects directory) is a rough estimate of the required memory for a single clone operation for repositories up to 400 MB. For larger repositories, memory usage flattens out at about 700 MB.

The clone operation is the most memory intensive Git operation. Most other Git operations, such as viewing file history, file contents and commit lists are lightweight by comparison.

Stash has been designed to have fairly constant memory usage. Any pages that could show large amounts of data (e.g. viewing the source of a multi-megabyte file) perform incremental loading or have hard limits in place to prevent Stash from holding on to large amounts of memory at any time. In general, the default memory settings (max. 768 MB) should be sufficient to run Stash. The maximum amount of memory available to Stash can be configured in setenv.sh or setenv.bat.

⚠️ The memory consumption of Git is not managed by the memory settings in setenv.sh or setenv.bat. The Git processes are executed outside of the Java virtual machine, and as a result the JVM memory settings do not apply to Git.

CPU

In Stash, much of the heavy lifting is delegated to Git. As a result, when deciding on the required hardware to run Stash, the CPU usage of the Git processes is the most important factor to consider. And, as is the case for memory usage, cloning large repositories is the most CPU intensive Git operation. When you clone a repository, Git on the server side will create a pack file (a compressed file containing all the commits and file versions in the repository) that is sent to the client. While preparing a pack file, CPU usage will go up to 100% for one CPU.

For users that connect to Stash using SSH, the encryption of data adds to overall CPU usage. For day-to-day push and pull operations the overhead will not be significant, but when cloning repositories the overhead will be noticeable.

ℹ️ To get the maximum performance from Stash, we advise configuring automatic build tools to use the http or https protocol, if possible.
**Clones examined**

Since cloning a repository is the most demanding operation in terms of CPU and memory, it is worthwhile analyzing the clone operation a bit closer. The following graphs show the CPU and memory usage of a clone of a 220 MB repository:

**Git process (blue line)**
- CPU usage goes up to 100% while the pack file is created on the server side.
- CPU peaks at 120% when the pack file is compressed (multiple CPUs used).
- CPU drops back to 0.5% while the pack file is sent back to the client.

**Stash (red line)**
- CPU usage briefly peaks at 30% while the clone request is processed.
- CPU drops back to 0% while Git prepares the pack file.
- CPU hovers around 1% while the pack file is sent to the client.

**Git process (blue line)**
- Memory usage slowly climbs to 270 MB while preparing the pack file.
- Memory stays at 270 MB while the pack file is transmitted to the client.
- Memory drops back to 0 when the pack file transmission is complete.

**Stash (red line)**
- Memory usage hovers around 800 MB and is not affected by the clone operation.
This graph shows how concurrency affects average response times for clones:

Vertical axis: average response times.
Horizontal axis: number of concurrent clone operations.

The measurements for this graph were done on a 4 CPU server with 12 GB of memory.
Response times become exponentially worse as the number of concurrent clone operations exceed the number of CPUs.

Configuring Stash scaling options and system properties

Stash only allows a fixed number of Git commands to be executed concurrently, to prevent the performance for all clients dropping below acceptable levels. Stash has two settings to control the number of Git processes that are allowed to process in parallel: one for the web UI and one for the ‘hosting’ operations (pushing and pulling commits and cloning a repository).

The settings can be overridden by creating a stash-config.properties in STASH_HOME with the following content:

```
STASH_HOME/stash-config.properties

# The maximum number of concurrent requests using git commands using the UI or REST services (e.g. git diff via the UI). Default value is 25.
throttle.resource.scm-command=20

# Controls how long threads will wait for SCM commands to complete when the system is already running the maximum number of SCM commands. Value is in seconds. Default is 2 seconds.
throttle.resource.scm-command.timeout=2

# The maximum number of concurrent requests using "hosting" commands, git clone, git push, git pull. Default value is 1.5*cpu (1.5 times the number of available cpus/cores).
throttle.resource.scm-servlet=20

# Controls how long threads will wait for SCM commands to complete when the system is already running the maximum number of SCM commands. Value is in seconds. Default is 300 seconds (5 minutes).
throttle.resource.scm-servlet.timeout=300
```
What happens when the limits are reached?

For the given resource, the request will wait until a currently running request has completed. If no request completes within a configurable timeout, the request will be rejected.

When non-servlet requests are rejected, users will start seeing messages in the UI indicating that the server is under load.

When servlet requests are rejected, it can be a little harder to determine for a user, so Stash does a few different things:

- Stash will return either a 503 or a 501 response code to the Git client as described in [Git commands return error code 503](#) and [Git commands return error code 501](#).
- A warning message will be logged for every time a request is rejected due to the resource limits.
- For five minutes after a request is rejected, Stash will display a red banner in the UI to warn that the server is under load.

System settings

This page lists the Java VM system properties which may be set to control certain aspects of Stash's behaviour.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Values</th>
<th>Default</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>file.encoding</td>
<td>utf-8</td>
<td>utf-8</td>
<td>The Stash WAR needs to be run with the file.encoding property set to utf-8 in the JVM.</td>
</tr>
</tbody>
</table>

To set JVM system properties:

There are 2 approaches to setting JVM system properties:

- The recommended approach is to add the property to `<Stash installation directory>/system.properties`. This file can be created as a plain text file (it follows the Java properties format).
- You can also pass the -D parameter to the JVM.

Stash config properties

This page describes very specific configuration for your database connection parameters, which are set by Stash during database setup and migration. We don't expect that you will edit these except in collaboration with Atlassian Support.

In your Stash home directory, the database configuration is contained within the stash-config.properties file. For example:

```
Related pages:
- Installing Stash on Windows
- Installing Stash on Linux and Mac
- Connecting Stash to an external database
- Supported platforms
```
jdbc.url=jdbc:postgresql://localhost:5432/stash
jdbc.user=stash_user
jdbc.password=s3cr3t
jdbc.driver=org.postgresql.Driver

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jdbc.url</td>
<td>This is the JDBC url that Stash will use to connect to the database. This should include the driver subprotocol (e.g., postgresql:), the hostname, port and database that you will connect to. This string may vary depending on the database you are connecting to. Please seek specific examples for other databases from your database provider.</td>
</tr>
<tr>
<td>jdbc.user</td>
<td>This is the user that Stash will connect to the database with. The user will need to be able to create and drop tables and indexes, as well as read and write operations on the entire database schema defined in jdbc.url.</td>
</tr>
<tr>
<td>jdbc.password</td>
<td>The password that the user defined by jdbc.user will connect with.</td>
</tr>
<tr>
<td>jdbc.driver</td>
<td>The JDBC driver class that should be used by Stash to connect to the database.</td>
</tr>
</tbody>
</table>

If none of the above values are specified in stash-config.properties, then a provided HSQL database will be used.

**Securing Stash with HTTPS**

This page is intended for administrators setting up Stash for a small team, and describes how to enable SSL access for Tomcat, the webserver distributed with Stash, using a self-signed certificate. You should consider running Stash with HTTPS (HTTP over SSL) and making secure access mandatory, if Stash will be internet-facing where usernames, passwords and other proprietary data may be at risk.

Those setting up a production instance should consider using a CA certificate, briefly described below.

Be aware that you can set up Stash to run behind a web server, such as Apache HTTP Server. To secure Stash when Apache HTTP Server acts as a reverse proxy for Stash see Integrating Stash with Apache HTTP Server.

⚠️ Please note that Atlassian Support will refer SSL-related support to the issuing authority for the certificate. The documentation on this page is for reference only.
1. Generate a self-signed certificate

Self-signed certificates are useful where you require encryption but do not need to verify the website identity. They are commonly used for testing and on internal corporate networks (intranets).

Users may receive a warning that the site is untrusted and have to "accept" the certificate before they can access the site. This usually will only occur the first time they access the site.

The following approach to creating a certificate uses Java's `keytool`, for Java 1.6. Other tools for generating certificates are available.

To generate a self-signed certificate:
Log in with the user account that Stash will run under, and run the following command:

<table>
<thead>
<tr>
<th>Windows</th>
<th>&quot;$JAVA_HOME\bin\keytool&quot; -genkey -alias tomcat -keyalg RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux, MacOS and Unix</td>
<td>$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA</td>
</tr>
</tbody>
</table>

This will create (if it doesn’t already exist) a new .keystore file located in the home directory of the user you used to run the keytool command.

**Note the following:**

- When running the keytool command you will be prompted with: *What is your first and last name?*

  You must enter the **fully qualified hostname** of the server running Stash. This is the name you would type in your web browser after 'http://' (no port number) to access your Stash installation. The qualified host name should match the base URL you have set in Stash (without the port number).

- The keytool utility will also prompt you for two passwords: the keystore password and the key password for Tomcat.

  You must use the same value for both passwords, and the value must be either:
  - "changeit", which is the default value Tomcat expects, or
  - any other value, but you must also specify it in conf/server.xml by adding the following attribute to the `<Connector/>` tag: `keystorePass="<password value>"`

2. **Configure HTTPS in Tomcat**

To configure HTTPS in Tomcat:

- Edit conf/server.xml and, at the bottom, before the `</Service>` tag, add this section (or uncomment it if it already exists):

  ```xml
  <Connector port="8443"
    maxHttpHeaderSize="8192"
    SSLEnabled="true"
    maxThreads="150"
    minSpareThreads="25"
    maxSpareThreads="75"
    enableLookups="false"
    disableUploadTimeout="true"
    useBodyEncodingForURI="true"
    acceptCount="100"
    scheme="https"
    secure="true"
    clientAuth="false"
    sslProtocol="TLS" />
  ```

  This enables SSL access on port 8443 (the default for HTTPS is 443, but 8443 is used instead of 443 to avoid conflicts).
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:

SSL + Apache + IE problems

Some people have reported errors when uploading attachments over SSL using IE. This is due to an IE bug, and can be fixed in Apache by setting:

BrowserMatch ".MSIE." \
  nokeepalive ssl-unclean-shutdown \
  downgrade-1.0 force-response-1.0

Google has plenty more on this.

Can't find the keystore

java.io.FileNotFoundException: /home/user/.keystore (No such file or directory)

This indicates that Tomcat cannot find the keystore. The keytool utility creates the keystore as a file called .keystore in the current user's home directory. For Unix/Linux the home directory is likely to be /home/<username>. For Windows it is likely to be C:\User\<UserName>.

Make sure you are running Stash as the same user who created the keystore. If this is not the case, or if you are running Stash 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
Wrong certificate

```java
javax.net.ssl.SSLException: No available certificate corresponds to the SSL cipher suites which are enabled.
```

If the Keystore has more than one certificate, Tomcat will use the first returned unless otherwise specified in the SSL Connector in `conf/server.xml`.

Add the `keyAlias` attribute to the Connector tag you uncommented, with the relevant alias, for example:

```xml
<Connector port="8443"
    maxHttpHeaderSize="8192"
    maxThreads="150"
    minSpareThreads="25"
    maxSpareThreads="75"
    enableLookups="false"
    disableUploadTimeout="true"
    useBodyEncodingForURI="true"
    acceptCount="100"
    scheme="https"
    secure="true"
    clientAuth="false"
    sslProtocol="TLS"
    keystoreFile="/opt/local/.keystore"
    keystorePass="removed"
    keyAlias="tomcat"/>
```

Using Apache Portable Runtime

APR uses a different SSL engine, and you will see an exception like this in your logs

```java
SEVERE: Failed to initialize connector [Connector[HTTP/1.1-8443]]
LifecycleException: Protocol handler initialization failed: java.lang.Exception: No Certificate file specified or invalid file format
```

The reason for this is that the APR Connector uses OpenSSL and cannot use the keystore in the same way. You can rectify this in one of two ways:

**Use the Http11Protocol to handle SSL connections**

Edit the `server.xml` so that the SSL Connector tag you just uncommented specifies the Http11Protocol instead of the APR protocol:
<Connector port="8443"
    maxHttpHeaderSize="8192"
    SSLEnabled="true"
    keystoreFile="${user.home}/.keystore"
    maxThreads="150"
    enableLookups="false"
    disableUploadTimeout="true"
    acceptCount="100"
    scheme="https"
    secure="true"
    clientAuth="false"
    sslProtocol="TLS"
    useBodyEncodingForURI="true" />

Configure the Connector to use the APR protocol

This is only possible if you have PEM encoded certificates and private keys. If you have used OpenSSL to generate your key, then you will have these PEM encoded files - in all other cases contact your certificate provider for assistance.

<Connector port="8443"
    maxThreads="200"
    scheme="https"
    secure="true"
    SSLEnabled="true"
    SSLCertificateFile="${user.home}/certificate.pem"
    SSLCertificateKeyFile="${user.home}/key.pem"
    clientAuth="optional"
    SSLProtocol="TLSv1"/>

Enabling client authentication

To enable client authentication in Tomcat, ensure that the value of the clientAuth attribute in your Connector element of your Tomcat's server.xml file is true.

<Connector
    ... clientAuth="true"
    ... />

For more information about Connector element parameters, please refer to the 'SSL Support' section of the Tomcat 6.0 documentation.

Exporting the self-signed certificate

If Stash will run as the user who ran the keytool --genkey command, you do not need to export the certificate.

You may need to export the self-signed certificate, so that you can import it into a different keystore, if Stash will not be run as the user executing keytool --genkey. You can do so with the following command:
If you generate the certificate as one user and run Stash as another, you'll need to do the certificate export as the generating user and the import as the target user.

**Requesting a CA certificate**

Digital certificates that are issued by trusted 3rd party CAs (Certification Authorities) provide verification that your website does indeed represent your company.

When running Stash in a production environment, you will need a certificate issued by a CA, such as [VeriSign](http://www.verisign.com) or [TrustCenter](http://www.trustcenter.com). The instructions below are adapted from the [Tomcat documentation](http://tomcat.apache.org/tomcat-6.0/appendix)viewed.

First, you will generate a local certificate and create a 'certificate signing request' (CSR) based on that certificate. You then submit the CSR to your chosen certificate authority. The CA will use that CSR to generate a certificate for you.

1. Use Java's `keytool` utility to generate a local certificate, as described in the [section above](http://tomcat.apache.org/tomcat-6.0/appendix/viewed).
2. Use the `keytool` utility to generate a CSR, replacing the text `<MY_KEYSTORE_FILENAME>` with the path to and file name of the .keystore file generated for your local certificate:

   **Windows**
   ```
   "%JAVA_HOME%\bin\keytool" -certreq
   -keyalg RSA -alias tomcat -file certreq.csr -keystore <MY_KEYSTORE_FILENAME>
   ```

   **Linux, MacOS and Unix**
   ```
   $JAVA_HOME/bin/keytool -certreq
   -keyalg RSA -alias tomcat -file certreq.csr -keystore <MY_KEYSTORE_FILENAME>
   ```

3. Submit the generated file called `certreq.csr` to your chosen certificate authority. Refer to the documentation on the CA's website to find out how to do this.
4. The CA will send you a certificate.
5. Import the new certificate into your local keystore. Assuming your certificate is called "file.cer" whether obtained from a CA or self-generated, the following command will add the certificate to the keystore:

   **Windows**
   ```
   "%JAVA_HOME%\bin\keytool" -import
   -alias tomcat -file file.cer
   ```

   **Linux, MacOS and Unix**
   ```
   $JAVA_HOME/bin/keytool -import -alias
tomcat -file file.cer
   ```

**Enabling SSH access to Git repositories in Stash**

Enabling SSH access to Git repositories in Stash allows your Stash users to:

- add their own SSH keys to Stash
- use those SSH keys to secure Git operations between their computer and the Stash server.

Stash users must each add their own SSH key pairs to Stash to be able to use SSH access.
Supported key types are DSA and RSA2. Note that RSA1 is not supported. We've tested key sizes of 768, 1024, 2048, 4096 and 8192 bytes.

There are performance implications for Stash when using SSH. When users connect to Stash using SSH, the encryption of data adds to overall CPU usage. For day-to-day push and pull operations the overhead will not be significant, but when cloning repositories the overhead will be noticeable.

To get the maximum performance from Stash, we advise configuring automatic build tools to use the http or https protocol, if possible. See Scaling Stash for more information.

On this page:
- Enabling SSH access
- SSH base URL
- When running Stash behind a proxy

Related pages:
- Setting up SSH port forwarding
- Setting up SSH on Windows
- Setting up SSH on Linux and Mac

Enabling SSH access

To enable SSH access:

1. Go to Administration > Settings > Server settings.
2. Under 'SSH access', check SSH enabled.
3. Enter values for SSH port and SSH base URL, according to the information in the sections below.
4. Click Save.

SSH access

- SSH enabled
- SSH port
- SSH base URL

SSH base URL

The SSH base URL is the base URL with which users can access the SSH push/pull/clone functionality of Stash.

This is the base URL that Stash will use when displaying SSH URLs to users. If you do not set this, it will default to the host that is set in Stash base URL, with the port that SSH is listening on.
If you set up port forwarding, you will need to set the **SSH base URL** to the machine and port that is being forwarded to Stash. However, you do not need to specify the port portion of the URL if the default SSH port (port 22) is being forwarded to Stash.

![Diagram showing client and Stash with port forwarding](image)

<table>
<thead>
<tr>
<th>Port forwarding</th>
<th>SSH base URL</th>
<th>Stash base URL</th>
<th>SSH port</th>
<th>Resulting SSH URL for a repo</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>Not set</td>
<td><a href="https://stash">https://stash</a>. atlassian.com</td>
<td>7999</td>
<td>ssh://git@stash. atlassian. com:7999/&lt;projectname&gt;/&lt;reponame&gt;.git</td>
</tr>
<tr>
<td>✔ Port 22 -&gt; 7999</td>
<td><a href="https://stash">https://stash</a>. atlassian.com</td>
<td><a href="https://stash">https://stash</a>. atlassian.com</td>
<td>7999</td>
<td>ssh://git@stash.backend. atlassian.com:7999/ATLASSIAN/jira.git</td>
</tr>
</tbody>
</table>

**When running Stash behind a proxy**

If you run Stash behind a http proxy such as Apache (e.g. as per our instructions), and if Apache runs on a different host, SSH will not be available on that host. Instead, you will need to set the SSH base URL to the machine Stash is actually running on (and the URL should include the SSH port Stash is serving from).

![Diagram showing client and Stash with proxy](image)

For example, if the **SSH base URL** is set to `ssh://stash.backend. atlassian.com:7999`, the SSH URL for the repository Jira in the project Atlassian will be `ssh://git@stash.backend. atlassian.com:7999/ATLASSIAN/jira.git`.

If you set up port forwarding, you will need to set the **SSH base URL** to the proxy machine and port that is being forwarded to Stash. However, you do not need to specify the port portion of the URL if the default SSH port (port 22) is being forwarded to Stash.
For example, if you set up port forwarding from your http proxy host, stash.atlassian.com, port 22, to stash.backend.atlassian.com port 7999, set the **SSH base URL** to ssh://stash.atlassian.com. Then, the SSH URL for the repository Jira in the project Atlassian will be ssh://git@stash.atlassian.com/ATLASSIAN/jira.git.

<table>
<thead>
<tr>
<th>Port forwarding</th>
<th>SSH base URL</th>
<th>SSH port</th>
<th>Stash base URL</th>
<th>Resulting SSH URL for a repo</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>ssh://stash.backend.atlassian.com:7999</td>
<td>7999</td>
<td><a href="https://stash.backend.atlassian.com">https://stash.backend.atlassian.com</a></td>
<td>ssh://git@stash.backend.atlassian.com:7999/PROJECTNAME/REPONAME.git</td>
</tr>
<tr>
<td>✔ Port 22 -&gt; 7999</td>
<td>ssh://stash.atlassian.com</td>
<td>7999</td>
<td><a href="https://stash.backend.atlassian.com">https://stash.backend.atlassian.com</a></td>
<td>ssh://git@stash.atlassian.com/PROJECTNAME/REPONAME.git</td>
</tr>
<tr>
<td>✔ Port 44 -&gt; 7999</td>
<td>ssh://stash.atlassian.com:44</td>
<td>7999</td>
<td><a href="https://stash.backend.atlassian.com">https://stash.backend.atlassian.com</a></td>
<td>ssh://git@stash.atlassian.com:44/PROJECTNAME/REPONAME.git</td>
</tr>
</tbody>
</table>

**Setting up SSH port forwarding**

**Why set up port forwarding?**

There are two scenarios where you might want to set up port forwarding.

*Remove port numbers from your SSH URLs*

Stash listens for SSH connections on port 7999 by default.
Your users will need to include the port in the URL they use to clone from Stash, for example:

```
$ git clone ssh://git@stash.yourcompany.com:7999/PROJECT/repo.git
```

Rather than have the port number in the URL, you may wish to set up port forwarding so that connections to the default SSH port are forwarded to the port Stash is listening on (e.g. you could forward port 22 to port 7999).

This would allow your users to use a URL without a port number in it, like this:

```
$ git clone ssh://git@stash.yourcompany.com/PROJECT/repo.git
```

**Stash is running behind a reverse proxy on a separate machine**

You may be following our instructions for setting up Stash behind an Apache front-end.

In this case, your users may not be able to access Stash directly for SSH connections, or if they can, you may wish to make the SSH and HTTP/S URLs consistent.

![Diagram](https://via.placeholder.com/150)

For example, if you have the above topology, without port forwarding (and assuming the default port of 7999), your users will need to clone Stash directly from the backend, like this:

```
$ git clone ssh://git@stash.backend.atlassian.com:7999/PROJECT/repo.git
```

In your network, the `stash.backend.atlassian.com` machine may not be accessible directly, or you may want the URL to be consistent with the HTTP/S URL of `https://stash.atlassian.com/scm/PROJECT/repo.git`.

In this case, you need to set up port forwarding on the `stash.atlassian.com` machine to accept connections and forward them to port 7999 on the `stash.backend.atlassian.com` machine.

**How to set up port forwarding**

**HAProxy**

Atlassian recommends the use of HAProxy for forwarding SSH connections through to Stash.

HAProxy is supported on Linux, Solaris and FreeBSD.
Installing HAProxy

Your Operating System may support installing HAProxy via its system package manager, such as apt-get, yum or rpm. This will be the easiest way.

Alternatively, you may build HAProxy yourself and install it.

1. Download the latest version of HAProxy from http://haproxy.1wt.eu/#down.
2. Extract the archive and cd into the directory:

   ```
   tar xzvf haproxy-1.4.21.tar.gz
   cd haproxy-1.4.21
   ```

3. Read the instructions in the README for how to build on your system. This is generally quite simple - on a Linux 64 bit 2.6 Kernel, the command is:

   ```
   make TARGET=linux26 ARCH=x86_64
   ```

4. If it completes successfully, install it following the instructions in the README:

   ```
   sudo make install
   ```

Configuring HAProxy

HAProxy is extremely powerful - it is designed as a http/s load balancer, but also can serve as a port forwarder for ssh.

The full documentation for version 1.4 is here. More documentation is available on the HAProxy web site.

An example simple configuration is as follows:

```
global
  daemon
  maxconn 10000

defaults
  timeout connect 500s
  timeout client 5000s
  timeout server 1h

frontend sshd
  bind *:7999
  default_backend ssh
  timeout client 2h

backend ssh
  mode tcp
  server localhost-stash-ssh 127.0.0.1:7999 check port 7999
```

The above configuration will listen on port 7999 (indicated by the bind directive) on all network interfaces. As indicated by the server directive, traffic is forwarded to 127.0.0.1, port 7999. You will need to replace 127.0.0
You can check your configuration by running:

```bash
haproxy -f haproxyconf.txt -c
```

To run haproxy, simply start it using

```bash
haproxy -f haproxyconf.txt
```

If you use HAPerxy to additionally proxy HTTP traffic, ensure that the running `mode configuration` is set to `http`:

```bash
backend http
  mode http
  bind *:80
  server localhost-stash-http 127.0.0.1:7990
```

Using the default SSH port

You can configure HAPerxy to listen on the default SSH port instead, so that the port does not need to be specified in the clone URL.

By default, the normal ssh daemons are running on port 22. You have several options:

- Configure HAPerxy to listen on an alternate port as in the previous example.
- Configure multiple network interfaces on the physical machine and force the default ssh daemon to listen on all but the interface for accessing Stash. Configure HAPerxy to only listen on that interface.
- Move the default ssh daemon to listen on another port and let HAPerxy bind on port 22.

We do not provide instructions on the last two options, except for how to configure HAPerxy.

Use the same configuration as the last example, but change the bind port to 22, e.g.

```bash
...  
  frontend sshd
  bind *:22
  ...  
```

You will have to run this configuration as the `root` user, using `sudo`, because it specifies a port to listen on that is less than 1024.

```bash
sudo haproxy -f haproxyconf.txt
```

Configuring the SSH base URL
Once port forwarding is set up, you will need to configure the SSH base URL in Stash so that the clone urls presented in Stash indicate the correct host and port to clone from. See the SSH base URL section in Enabling SSH access to Git repositories in Stash.

**Releases**

**Stash 1.2**

07 August 2012

- MySQL, PostgreSQL, SQL Server and Oracle support
- Database migration
- File search
- Add-ons ecosystem
- Small improvements

Read the Stash 1.2 release notes.

See the change log for Stash 1.2.x minor releases.

**Stash 1.1**

19 June 2012

- SSH support
- Fast browsing
- Simple permissions
- Image diffs

Read the Stash 1.1 release notes.

See the change log for Stash 1.1.x minor releases.

**Stash 1.0 is released!**

1st May 2012

Atlassian Stash is a repository management solution that allows everyone in your organisation to easily collaborate on all your Git repositories.

In Stash you can:

- Create Git repositories and organize them into projects
- Browse your repositories and your commits
- View the changesets, diffs, blame and history of your files
- Create new users and organize them into groups
- Manage permissions at a global and at a project level
- Integrate with JIRA

Read the Stash 1.0 release notes.

See the change log for Stash 1.0.x minor releases.

**Stash upgrade guide**

The instructions on this page describe how to upgrade Stash from a previous version.

- For details of the latest Stash 1.2.x release, see the Stash 1.2 change log.
- For the latest and greatest Stash release, see Releases.
Please read the Supported platforms page for the full list of supported platforms for Stash.

### Upgrade notes

These upgrade notes are general instructions for upgrading Stash. We strongly recommend that you upgrade Stash by following these steps:

1. **Stop Stash!**

   To stop Stash, change directory in a terminal or command prompt to the `<Stash installation directory>` and run:

   - **Windows:**
     
     ```
     bin\stop-stash.bat
     ```

   - **Linux and Mac:**
     
     ```
     bin/stop-stash.sh
     ```

2. **Back up your Stash data!**

   - Back up the Stash home directory. This is where your Stash data is stored. The home directory location is defined:
     
     - On Windows: by the STASH_HOME environment variable, or by the STASH_HOME line of `<Stash installation directory>/bin/setenv.bat`.
     - On Linux and Mac: by the STASH_HOME line of `<Stash installation directory>/bin/setenv.sh`.

     - If you are using an external database, back up this database. Follow the directions provided by the database vendor to do this.

3. **Download and install Stash as usual**

   In particular, you must redefine the Stash home directory. See the following for more information:

   - Installing Stash on Windows
   - Installing Stash on Linux and Mac

   If you made custom changes to the configuration of your existing Stash installation, for example for the port or context path, you will have to make these changes for the new installation as well.
If you are using MySQL

Stash does not ship with the MySQL database driver.

You will need to reinstall the driver in the new installation, or copy the previous driver from the old <Stash installation directory>/lib to the new <Stash installation directory>/lib.

4. Start Stash

To start Stash, change directory in a terminal or command prompt to the <Stash installation directory> and run:

- **Windows:**
  
  `bin\start-stash.bat`

- **Linux and Mac:**
  
  `bin/start-stash.sh`

Upgrading from Stash 1.0.x to 1.1 or higher

SSH

When you restart Stash after upgrading to 1.1, Stash will automatically enable SSH access to your repositories, on the default port of 7999.

If you want to change the port, or are hosted behind a proxy or firewall, you may also need to change the SSH base URL so the clone URL's in Stash are correct. See the [SSH admin instructions](#).

Developing for Stash

If you are a Stash plugin developer, please refer to our [Stash developer documentation](#).

Checking for known issues and troubleshooting the Stash upgrade

If something is not working correctly after you have completed the steps above to upgrade your Stash installation, please check for known Stash issues and try troubleshooting your upgrade as described below:

- **Check for known issues.** Sometimes we find out about a problem with the latest version of Stash after we have released the software. In such cases we publish information about the known issues in the [Stash Knowledge Base](#).

- If you encounter a problem during the upgrade and cannot solve it, please create a [support ticket](#) and one of our support engineers will help you.

**Stash 1.2 release notes**

7th August 2012
Atlassian is proud to present Stash 1.2, which provides greatly improved support for your enterprise database, rapid file searching and new Stash add-ons available from the Atlassian Marketplace.

See the change log for Stash 1.2.x minor releases.

**Highlights of this release:**

- MySQL, PostgreSQL, SQL Server and Oracle support
- Database migration
- File search
- Add-ons ecosystem
- Merge filter

**Providing feedback:**

Please log your votes and issues. They help us decide what needs doing, and are much appreciated!

1

**MySQL, PostgreSQL, SQL Server and Oracle support**

Stash now has support for all these major databases: MySQL, Oracle, PostgreSQL and Microsoft SQL Server. Choose the database that best fits your needs, or your system administrator is most familiar with, or that your company is already using. More...

2

**Database migration**

Switch easily from the database embedded in Stash to your organisation's existing technology stack, and migrate painlessly if your system administrators change the infrastructure. Stash scales and adapts as your requirements change. More...
File search

Stash's new file search ensures that you can quickly find any file in your repository, without needing to check out the source. Just start typing any part of the file name into the search field and you'll get a list of matches, fast. And you can filter by path, CamelCase (for example, AttrM to match AttributeMap) and file extension.
Add-ons ecosystem

Visualise information about your Git repository, comment on your code in-line, collect achievements when committing code or receive change notifications in HipChat. With almost a dozen add-ons available on the Atlassian Marketplace, you can extend Stash to suit your needs.

Merge filter

Merges can be faded out in the commit list, as in this screenshot, to make it easier to see the important details. Use the 't' keyboard shortcut to toggle this effect.

The Stash 1.2 team

Development

Core team
Adam Ahmed
Bryan Turner
David Pinn
Federico Silva Armas
Jason Hinch
Jonathan Poh
Kostya Marchenko
Michael McGlynn
Michael Studman
Pierre-Etienne Poirot
Tim Pettersen
Xu-Heng Tjhin

**Team leads**

Matt Watson
Seb Ruiz

**Architect**

Michael Heemskerk

**Project manager**

Anton Mazkovoi

**Support**

Ajay Sridhar
Armen Khachatryan
Daniel Rohan
Douglas Fabretti
Felipe Kraemer
Gurleen Anand
Renan Battaglin
Rene Verschoor
Zed Yap

**Others**

**Product management**

Jens Schumacher

**Design and user experience**

Matt Bond

**Product marketing**

Giancarlo Lionetti
Jeff Park

**Technical writing**

Paul Watson

**Operations**
James Fleming

Stash 1.2 change log

This page will contain information about the Stash 1.2 minor releases as these become available. These releases will be free to all customers with active Stash software maintenance.

Don't have Stash 1.2 yet?

Take a look at all the features in the Stash 1.2 release notes and see what you are missing out on!

Upgrading from a previous version of Stash

If you are upgrading, please read the Stash upgrade guide.

Stash 1.2.1

17 August 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

<table>
<thead>
<tr>
<th>JIRA Issues (1 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>------</td>
</tr>
</tbody>
</table>

Stash 1.1 release notes

19th June 2012

Atlassian is proud to present Stash 1.1, which provides a simple and secure solution for managing your Git repositories in the Enterprise.

See the change log for Stash 1.1.x minor releases.

Highlights of this release:

- SSH support
- Fast browsing
- Simple permissions
SSH support

Developed from the ground up with enterprise level security as a #1 priority, Stash now supports SSH in addition to HTTPS. You can either use standard HTTPS authentication, or set up your public keys and connect to Stash using SSH. This resolves Stash's #1 feature request, focused on adding support for SSH security.

Developers can manage their own SSH keys, and add as many as they like. Stash administrators can grant or revoke the SSH keys of any user.

More...

Fast browsing

Stash 1.1 greatly improves productivity by making it easier to navigate and work with Git repositories.

Recent repositories

The new Repositories item in the Stash header is for those developers who work with several repositories and want a quick way to get back to one of those repositories.

Mouse-less productivity

Stash helps you to work even more efficiently without a mouse. Whether viewing changesets, browsing directories or jumping through your commit list, simply press ‘J’ or ‘K’ to move to the next or previous.
Simple permissions

Stash keeps you and your developers productive by providing a way to structure your repositories and manage permissions with a simple, yet powerful, user interface.

- **Global permissions** – delegate administration of projects to key users and groups, to give your developers the freedom to create and manage repositories
- **Projects permissions** – use simple permissions at the project level to control access to repositories for users and groups

The new permission screens provide a great overview of who has access to your projects, and managing permissions is even faster.

More...

Global Permissions

<table>
<thead>
<tr>
<th>Individual Users</th>
<th>System Administrator</th>
<th>Administrator</th>
<th>Project Creator</th>
<th>Stash User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adam Ahmed</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Administrator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anton Maksavoi</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extranet Bamboo User</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>David Pinn</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extranet Crucible User</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Federico Silva Armas</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geoff Cram</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(more users)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Image diffs

Stash makes diffs more accessible to everyone on your team, not just the back-end coders.

Have you ever tried to find the subtle difference between two images? That difference may be small like a text change or as large as a page redesign. Web designers, front-end developers, and maybe a few QA folks, rejoice and check out Stash's interactive image diff viewer.

Maybe even more useful is ediffs. When viewing a diff it can sometimes be difficult to distinguish textual changes. Stash solves this with the addition of ediffs so you can clearly see the textual changes added or
The
Stash
1.1 team

Development

Core team

Adam
Ahmed
Bryan Turner
David Pinn
Federico Silva Armas
Jason Hinch
Jonathan Poh
Kostya Marchenko
Michael McGlynn
Michael Studman
Pierre-Etienne Poirot
Tim Pettersen
Xu-Heng Tjhin

Team leads

Matt Watson
Seb Ruiz

Architect

Michael Heemskerk

Project manager

Anton Mazkovoi

Support

Ajay Sridhar
Armen Khachatryan
Daniel Rohan
Douglas Fabretti
Felipe Kraemer
Gurleen Anand
Renan Battaglin
Rene Verschoor
Zed Yap
Stash 1.1 change log

This page contains information about the Stash 1.1 minor releases. These releases are, of course, free to all customers with active Stash software maintenance.

Don't have Stash yet?

Take a look at all the features in the Stash 1.1 release notes and see what you are missing out on!

Upgrading from a previous version of Stash

If you are upgrading, please read the Stash upgrade guide.

Stash 1.1.2

13 July 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

<table>
<thead>
<tr>
<th>JIRA Issues (4 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>------</td>
</tr>
</tbody>
</table>
Stash 1.1.1

22 June 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

### JIRA Issues (2 issues)

<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Summar y</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
</table>

Stash 1.0 release notes

Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
1st May 2012

Atlassian is proud to present Stash 1.0, which provides a central place to create and manage Git repositories. It’s the place where all that distributed code comes back together, where you can find the latest official version of your project, and where you can keep track of what’s going on.

See the change log for Stash 1.0.x minor releases.

**Highlights of this release:**

- Git repository management
- Projects and permissions
- Built for Git, focused on Enterprise
- Browse your source and history
- JIRA integration

**Providing feedback:**

Please log your votes and issues. They help us decide what needs doing, and are much appreciated!

1

**Git repository management**

Stash provides a simple and powerful interface to create and manage Git repositories. Create repositories in a couple of clicks, and quickly choose those of your users and groups who will be contributors to the project, and those who will be just observers.

2

**Projects and permissions**

Since projects rarely consist of a single repository, Stash provides a convenient Project structure. This helps you to organise and manage repositories, and makes managing access to your repositories really simple.

With Stash you can empower end users to manage repositories themselves, while keeping control of the key...
administration functions. And because we want to make it easy for you to manage teams, Stash has a group
management feature to help you grant permissions across your organisation.

Built for Git, focused on Enterprise

Stash has everything you need to create and manage Git repositories efficiently behind the safety of your own firewall.

Stash doesn’t force administrators to use a pre-packaged appliance and so give up control. Whether on Windows, Linux or MacOS X, Stash will feel right at home on all platforms.

With LDAP, Crowd and JIRA support, you can manage your team easily, whether they are a small number of users in Stash’s internal directory, or 500 developers managed in your corporate directory.

User Directories

The table below shows the user directories currently configured for Stash.

<table>
<thead>
<tr>
<th>Directory Name</th>
<th>Type</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stash Internal Directory</td>
<td>Internal</td>
<td>3</td>
</tr>
<tr>
<td>Extranet Crowd</td>
<td>Atlassian Crow by</td>
<td>3</td>
</tr>
<tr>
<td>caviar-internal-directory (inactive)</td>
<td>Internal</td>
<td></td>
</tr>
</tbody>
</table>

Add Directory

Browse your source and history

Keep track of commits to the repositories you’re involved with and dive into the changesets to see exactly what has changed in the source. Use Stash’s user interface to quickly navigate your repository and view annotated changes in an easily digestible way.

JIRA integration

Created in 2012 by Atlassian. Licensed under a Creative Commons Attribution 2.5 Australia License.
JIRA integration saves users time when tracking and checking their development. Stash keeps track of all issues that are associated with commits. This allows users to navigate straight to the JIRA issues that are linked to the commits, and to see in JIRA an aggregate of all code changes related to an issue.

The Stash 1.0 team

Development

Core team

Adam Ahmed
Brendan Humphreys
Bryan Turner
Conor MacNeill
David Pinn
Federico Silva Armas
Geoff Crain
Jason Hinch
Jonathan Poh
Michael McGlynn
Michael Studman
Nick Pellow
Pierre-Etienne Poirot
Xu-Heng Tjhin

Team leads

Matt Watson
Seb Ruiz
Tim Pettersen

Architect

Michael Heemskerk

Project manager

Anton Mazkovoi

Support

Ajay Sridhar
Armen Khachatryan
Daniel Rohan
Douglas Fabretti
Felipe Kraemer
Gurleen Anand
Stash 1.0 change log

This page contains information about the Stash 1.0 minor releases. These releases are, of course, free to all customers with active Stash software maintenance.

Don't have Stash yet?

Take a look at all the features in the Stash 1.0 release notes and see what you are missing out on!

Upgrading from a previous version of Stash

If you are upgrading, please read the Stash 1.0 upgrade guide.

On this page:

- Stash 1.0.3
- Stash 1.0.2
- Stash 1.0.1

Stash 1.0.3

17 May 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.
## JIRA Issues (4 issues)

<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>📅</td>
<td>STASH-2 509</td>
<td>MD5 hash of email for Gravatar needs to use a lower-case version of email address</td>
<td>Jason Hinch [Atlassian]</td>
<td>Mary Whooley</td>
<td>🐈</td>
<td>Close</td>
<td>Fixed</td>
<td>May 08, 2012</td>
<td>May 15, 2012</td>
</tr>
</tbody>
</table>

### Stash 1.0.2

This was an internal release only.

### Stash 1.0.1

#### 7 May 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

## JIRA Issues (5 issues)

<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>📅</td>
<td>STASH-2 503</td>
<td>Improve</td>
<td>Michael</td>
<td>Michael</td>
<td>🐈</td>
<td>Close</td>
<td>Fixed</td>
<td>May 06, 2012</td>
<td>May 06, 2012</td>
</tr>
</tbody>
</table>
### Stash 1.0.1 release notes

**07 May 2012**

The Atlassian Stash team is proud to announce the release of Stash 1.0.1.

We've fixed several bugs in this release. Please see the 'Updates and fixes in this release' section below for details.

Stash 1.0.1 is, of course, free to all customers with active Stash software maintenance.

**Don't have Stash yet?**

Take a look at all the features in the Stash 1.0 release notes and see what you are missing out on!

---

<table>
<thead>
<tr>
<th>#</th>
<th>Issue</th>
<th>Description</th>
<th>Assignee</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>scalabilty by queueing of scm hosting operation under high load</td>
<td>Heemske rk [Atlassia n]</td>
<td>d</td>
<td>2012</td>
<td>2012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>🚨</td>
<td>STASH-2478</td>
<td>InvalidCharacterException in JIRA logs when application link to Stash is present</td>
<td>Tim Pettersen [Atlassia n]</td>
<td>⚠️ Close</td>
<td>Fixed</td>
<td>Apr 30, 2012</td>
<td>May 06, 2012</td>
</tr>
</tbody>
</table>
### Upgrading from a previous version of Stash

If you are upgrading, please read the [Stash 1.0 upgrade guide](#).

### Updates and fixes in this release

The issues addressed in Stash 1.0.1 are shown below.

#### JIRA Issues (5 issues)

<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Priority</th>
<th>Status</th>
<th>Resolution</th>
<th>Created</th>
<th>Updated</th>
</tr>
</thead>
</table>
Stash 1.0 upgrade guide

The instructions on this page describe how to upgrade Stash from a previous version.

- For details of the latest Stash 1.0.x release, see the Stash 1.0 change log.
- For the latest and greatest Stash release, see Releases.

Please read the Supported platforms page for the full list of supported platforms for Stash.

**On this page:**
- Upgrade notes
- Upgrading from Stash 1.0
- Developing for Stash
- Checking for known issues and troubleshooting the Stash upgrade

**Related pages:**
- Releases
- Getting started
- Administering Stash

Upgrade notes

These upgrade notes are specific to Stash 1.0.x. We strongly recommend that you upgrade Stash by following these steps:

1. Stop Stash!

To stop Stash, run the following command in a terminal:

- Windows:
  <Stash installation directory>\bin\stop-stash.bat

- Linux and Mac:
  <Stash installation directory>/bin/stop-stash.sh

2. Back up your Stash instance!

   - Back up the Stash home directory. This is where your Stash data is stored. The home directory is specified either in <Stash installation directory>\bin\setenv.bat or in the STASH_HOME environment variable (on Windows).
   - If you are using an external database, back up this database. Follow the directions provided by the database vendor to do this.

3. Download and install Stash as usual

   In particular, you must redefine the Stash home directory, either in <Stash installation directory>\bin\setenv.bat or in the STASH_HOME environment variable (on Windows). See the following for more information:
   - Installing Stash on Windows
   - Installing Stash on Linux and Mac
If you made custom changes to the configuration of your existing Stash installation, for example for the port or context path, you will have to make these changes for the new installation as well.

4. Start Stash

See the following for more information:

- Installing Stash on Windows
- Installing Stash on Linux and Mac

Upgrading from Stash 1.0

There are no known issues associated with upgrading from Stash 1.0 to 1.0.1.

Developing for Stash

If you are a Stash plugin developer, please refer to our Stash developer documentation.

Checking for known issues and troubleshooting the Stash upgrade

If something is not working correctly after you have completed the steps above to upgrade your Stash installation, please check for known Stash issues and try troubleshooting your upgrade as described below:

- Check for known issues. Sometimes we find out about a problem with the latest version of Stash after we have released the software. In such cases we publish information about the known issues in the Stash Knowledge Base.
- If you encounter a problem during the upgrade and cannot solve it, please create a support ticket and one of our support engineers will help you.

Stash security advisories

Finding and reporting a security vulnerability

Atlassian's approach to releasing patches is detailed in How to report a security issue.

Publication of Bamboo security advisories

Atlassian's approach to publishing security advisories is detailed in Security advisory publishing policy.

Severity levels

Atlassian's scale for measuring security issues is detailed in Severity levels for security issues.

Our patch policy

Atlassian's approach to releasing patches is detailed in our Security patch policy.

Security advisories

Git resources

Get Git

Mac: http://code.google.com/p/git-osx-installer/downloads/list?can=3
Basic Git

Basic Git commands

Git cheat sheets and other resources

http://rogerdudler.github.com/git-guide/

http://byte.kde.org/~zrusin/git/git-cheat-sheet-medium.png

http://nvie.com/posts/a-successful-git-branching-model/


http://ndpsoftware.com/git-cheatsheet.html#loc=workspace:


Stash FAQ

On this page:

- Repositories
  - Q: I'm getting a "broken pipe" error when pushing my commits.
  - Q: Does Stash support Mercurial (Hg)? What about other version control systems?
  - Q: What about Git repository management in FishEye and Crucible?
  - Q: Why did you create a new product for Git repository management? Couldn't you build this into FishEye?
  - Q: Does FishEye require Stash? Does Stash require FishEye? Can they be used together?

- Integration
  - Q: Does Stash work with JIRA? If so, what version of JIRA do I need to run Stash?
  - Q: Will Stash be available for Atlassian OnDemand?

- Licensing
  - Q: The user tiers only go to 500 users - what do I do if I need more?
  - Q: I'm evaluating/have a quote for Stash and want to buy, but the promotion just ended. Am I still eligible for the discount?
  - Q: How long does the promotional discount last?

Related pages:

- Stash Knowledge Base Home
- Support policies
Repositories

Q: I'm getting a "broken pipe" error when pushing my commits.

A: This error occurs when the amount of data you’re trying to push in one go exceeds Git’s http post buffer. Just run the following command to increase it to 500MB.

```bash
git config http.postBuffer 524288000
```

See [Git push fails with 'fatal: The remote end hung up unexpectedly'](https://help.github.com/articles/github-pages/).

Q: Does Stash support Mercurial (Hg)? What about other version control systems?

A: Currently Stash does not support Mercurial. We will be gauging demand for Mercurial support as we move forward - [STASH-2469](https://jira.atlassian.com/view/issue/stash?issue=S) - Authenticate to see issue details.

Q: What about Git repository management in FishEye and Crucible?

A: The current Git repository management feature in FishEye will be deprecated in the near future. We encourage those interested in Git repository management to check out Stash. There is a promotional discount of 50% for active FishEye customers until June 30, 2012.

Q: Why did you create a new product for Git repository management? Couldn’t you build this into FishEye?

A: In FishEye 2.7 we added basic capabilities to host and manage Git repositories within FishEye. However, as we were planning future releases, we realized that the architecture of FishEye, built to index, browse and search across various SCMs, was not adequate for a DVCS repository management tool.

Therefore we have made the decision to build a new product, with a clear focus: hosting and managing Git repositories. Instead of a "Jack of all trades", we will have two products that are focused on 2 very different tasks:

1. Stash – Host, manage and collaborate on Git repositories
2. FishEye – Track, search and browse Subversion, Perforce, Git, Mercurial and CVS repositories in one place.
Q: Does FishEye require Stash? Does Stash require FishEye? Can they be used together?

A: FishEye and Stash are two separate standalone products that do not require each other. If you are using multiple source code management systems (SCM) at your organization it makes sense to use both FishEye and Stash. While you are managing your Git repositories with Stash, you can use FishEye to browse, search and reference code from other SCMs including Subversion.

Also, if you are using Git, Stash will provide your Git repository management, and FishEye will be a central place to keep track of changes and search for code across your repositories.

Integration

Q: Does Stash work with JIRA? If so, what version of JIRA do I need to run Stash?

A: Stash works with JIRA 4.3+. However, you will require the latest version of the JIRA/FishEye plugin to view commits in JIRA. See our documentation on JIRA integration.

Q: Will Stash integrate with any other Atlassian Tools? Crowd? Bitbucket? SourceTree?

A: Stash currently integrates with the JIRA issues tracker, SourceTree DVCS Mac client and Crowd user management solution. You can also connect to Stash via Bamboo to run your builds and deployments and we are planning even tighter integrations in the future.

Q: Will Stash be available for Atlassian OnDemand?

A: Atlassian Stash will not be available in OnDemand. If you are looking for a distributed version control solution to use with Atlassian OnDemand, we recommend using Bitbucket, our cloud based Git and Mercurial source code hosting solution. Bitbucket connects to Atlassian OnDemand via the JIRA DVCS connector.

Licensing

Q: The user tiers only go to 500 users - what do I do if I need more?

A: We have plans to release larger licenses in the not too distant future. Please contact us if you are interested in more than 500 users.

Q: I'm evaluating/have a quote for Stash and want to buy, but the promotion just ended. Am I still eligible for the discount?

A: All quotes will be honored up to 90 days past the end date (June 30, 2012) of the promotion period.

Q: How long does the promotional discount last?

A: The promotional discount for Stash has ended on June 30, 2012. To review our current pricing please go here.

How do I change the external database password
You can change the password the Stash uses to connect to an external database, however you don’t do this from the Stash Administration area – you must follow the procedure described below.

**Related pages:**
- Connecting Stash to an external database

To change the password that Stash uses when connecting to an external database:

1. Stop Stash, on [Windows](#), or on [Linux and Mac](#).
2. Get your database administrator to change the password on your database.
3. Go to your Stash home directory.

   Edit the `stash-config.properties` file to change the line that looks like:

```
jdbc.password=MY_PASSWORD
```

replacing MY_PASSWORD with your new database password.

4. Restart Stash, on [Windows](#), or on [Linux and Mac](#).

**Installation troubleshooting guide**

⚠️ There are currently no topics related to installation issues.

If you do encounter issues when installing Stash please [raise a support request](#). We will update this page based on the feedback that you give us.

**Stash home directory**

**Where is the Stash home directory?**

The Stash home directory is where your Stash data is stored. The home directory location is defined either by the STASH_HOME environment variable, or in the STASH_HOME line of:

- `<Stash installation directory>/bin/setenv.bat`, on Windows
- `<Stash installation directory>/bin/setenv.sh`, on Linux and Mac.

**What is in the Stash home directory?**

Your Stash home directory contains the following directories and files:

<table>
<thead>
<tr>
<th>Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Contains all repository data and the embedded HSQL database if an external database is not configured.</td>
</tr>
<tr>
<td>export</td>
<td>Contains database dump files produced during migrations between databases.</td>
</tr>
<tr>
<td>log</td>
<td>Contains logging files for Stash.</td>
</tr>
<tr>
<td>plugins</td>
<td>Contains plugin related data (such as externally uploaded plugins) for Stash.</td>
</tr>
<tr>
<td>tmp</td>
<td>Temporary directory for run-time related operations. Can be safely deleted when the Stash is <em>not running</em>.</td>
</tr>
</tbody>
</table>
stash-config.properties  File which contains configuration properties for Stash.

Securing the Stash home directory

The internal database files, the migration dump files and the stash-config.properties files all contain information that may be considered secret (server settings, salted and hashed user passwords, database passwords, etc).

For production use, we strongly recommend that you secure this directory against unauthorised access.

We recommend the following precautions:

- Assign a separate restricted user account on the machine for running Stash (not a root/administrator user)
  - If you wish to run Stash on port 80, use a separate http front end as described in Integrating Stash with Apache HTTP Server (do not run as root/Administrator if security of the home directory is important to you)
- Ensure that only the user running Stash can access the Stash home directory, by setting file system permissions appropriately for your operating system.

Support policies

Welcome to the support policies index page. Here, you’ll find information about how Atlassian Support can help you and how to get in touch with our helpful support engineers. Please choose the relevant page below to find out more.

- Bug fixing policy
- How to report a security issue
- New features policy
- Patch policy
- Security advisory publishing policy
- Security patch policy
- Severity levels for security issues

To request support from Atlassian, please raise a support issue in our online support system. To do this, visit support.atlassian.com, log in (creating an account if need be) and create an issue under Stash. Our friendly support engineers will get right back to you with an answer.

Bug fixing policy

Summary

- Atlassian Support will help with workarounds and bug reporting.
- Critical bugs will generally be fixed in the next maintenance release.
- Non critical bugs will be scheduled according to a variety of considerations.

Raising a Bug Report

Atlassian Support is eager and happy to help verify bugs — we take pride in it! Please open a support request in our support system providing as much information as possible about how to replicate the problem you are experiencing. We will replicate the bug to verify, then lodge the report for you. We’ll also try to construct workarounds if they’re possible.
Customers and plugin developers are also welcome to open bug reports on our issue tracking systems directly. Use http://jira.atlassian.com for the stand-alone products and http://studio.atlassian.com for JIRA Studio and Atlassian OnDemand.

When raising a new bug, you should rate the priority of a bug according to our JIRA usage guidelines. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

**How Atlassian Approaches Bug Fixing**

Maintenance (bug fix) releases come out more frequently than major releases and attempt to target the most critical bugs affecting our customers. The notation for a maintenance release is the final number in the version (ie the 1 in 3.0.1).

If a bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions) then it will be fixed in the next maintenance release provided that:

- The fix is technically feasible (i.e. it doesn't require a major architectural change).
- It does not impact the quality or integrity of a product.

For non-critical bugs, the developer assigned to fixing bugs prioritises the non-critical bug according to these factors:

- How many of our supported configurations are affected by the problem.
- Whether there is an effective workaround or patch.
- How difficult the issue is to fix.
- Whether many bugs in one area can be fixed at one time.

The developers responsible for bug fixing also monitor comments on existing bugs and new bugs submitted in JIRA, so you can provide feedback in this way. We give high priority consideration to security issues.

When considering the priority of a non-critical bug we try to determine a 'value' score for a bug which takes into account the severity of the bug from the customer's perspective, how prevalent the bug is and whether roadmap features may render the bug obsolete. We combine this with a complexity score (i.e. how difficult the bug is). These two dimensions are used when developers self serve from the bug pile.

**Further reading**

See Atlassian Support Offerings for more support-related information.

**How to report a security issue**

**Finding and Reporting a Security Vulnerability**

If you find a security bug in the product, please open an issue on http://jira.atlassian.com in the relevant project.

- Set the priority of the bug to 'Blocker'.
- Provide as much information on reproducing the bug as possible.
- Set the security level of the bug to 'Developer and Reporters only'.

All communication about the vulnerability should be performed through JIRA, so that Atlassian can keep track of the issue and get a patch out as soon as possible.
Further reading

See Atlassian Support Offerings for more support-related information.

New features policy

Summary

- We do not publish roadmaps.
- Product Managers review our most popular voted issues on a regular basis.
- We schedule features based on a variety of factors.
- Our Atlassian Bug Fixing Policy is distinct from our Feature Request process.
- Atlassian provides consistent updates on the top 20 feature/improvement requests (in our issue tracker systems).

How to Track what Features are Being Implemented

When a new feature or improvement is scheduled, the ‘fix-for’ version will be indicated in the JIRA issue. This happens for the upcoming release only. We maintain roadmaps for more distant releases internally, but because these roadmaps are often pre-empted by changing customer demands, we do not publish them.

How Atlassian Chooses What to Implement

In every major release we aim to implement highly requested features, but it is not the only determining factor. Other factors include:

- Direct feedback from face to face meetings with customers, and through our support and sales channels.
- Availability of staff to implement features.
- Impact of the proposed changes on the application and its underlying architecture.
- How well defined the requested feature is (some issues gain in popularity rapidly, allowing little time to plan their implementation).
- Our long-term strategic vision for the product.

How to Contribute to Feature Development

Influencing Atlassian’s release cycle

We encourage our customers to vote on feature requests in JIRA. The current tally of votes is available online in our issue tracking systems, http://jira.atlassian.com and http://studio.atlassian.com. Find out if your improvement request already exists. If it does, please vote for it. If you do not find it, create a new feature or improvement request online.

Extending Atlassian Products

Atlassian products have powerful and flexible extension APIs. If you would like to see a particular feature implemented, it may be possible to develop the feature as a plugin. Documentation regarding the plugin APIs is available. Advice on extending either product may be available on the user mailing-lists, or at 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.

Patch policy

Patch Policy

Atlassian will only provide software patches in extremely unusual circumstances. If a problem has been fixed in a newer release of the product, Atlassian will request that you upgrade your instance to fix the issue. If it is deemed necessary to provide a patch, a patch will be provided for the current release and the last maintenance release of the last major version only.

Patches are issued under the following conditions:

- The bug is critical (production application down or major malfunction causing business revenue loss or high numbers of staff unable to perform their normal functions).
- A patch is technically feasible (i.e., it doesn't require a major architectural change)
  OR
- The issue is a security issue, and falls under our Security Patch Policy.

Atlassian does not provide patches for non-critical bugs.

Provided that a patch does not impact the quality or integrity of a product, Atlassian will ensure that patches supplied to customers are added to the next maintenance release. Customers should watch a filed bug in order to receive e-mail notification when a "Fix Version" is scheduled for release.

Patches are generally attached to the relevant http://jira.atlassian.com issue.

Further reading

See Atlassian Support Offerings for more support-related information.

Security advisory publishing policy

Publication of Security Advisories

When a security vulnerability in an Atlassian product is discovered and resolved, Atlassian will inform customers through the following mechanisms:

- We will post a security advisory in the latest documentation of the affected product at the same time as releasing a fix for the vulnerability. This applies to all security advisories, including severity levels of critical, high, medium and low.
- We will send a copy of all security advisories to the Technical Alerts mailing list for the product concerned.
  Note: To manage your email subscriptions and ensure you are on this list, please go to my.atlassian.com and click 'Email Prefs' near the top right of the page.
- If the person who reported the vulnerability wants to publish an advisory through some other agency, such as CERT, we will assist in the production of that advisory and link to it from our own.

Early warning of critical security vulnerabilities:

- If the vulnerability is rated critical (see our criteria for setting severity levels) we may send an early warning to the Technical Alerts mailing list approximately one week before releasing the fix. This early warning is in addition to the security advisory itself, described above.
- However, if the vulnerability is publicly known or being exploited, we will release the security advisory and patches as soon as possible, potentially without early warning.

Further reading
Security patch policy

Product Security Patch Policy

Atlassian makes it a priority to ensure that customers’ systems cannot be compromised by exploiting vulnerabilities in Atlassian products.

Scope

This page describes when and how we release security patches and security upgrades for our products. It does not describe the whole of disclosure process that we follow. It also excludes JIRA Studio, since JIRA Studio will always be patched by Atlassian without additional notifications.

Critical vulnerabilities

When a Critical security vulnerability is discovered by Atlassian or reported by a third party, Atlassian will do all of the following:

- Issue a new, fixed release for the current version of the affected product as soon as possible, usually in a few days.
- Issue a binary patch for the current release.
- Issue a binary patch for the latest maintenance release of the previous version of the product.
- Patches for older versions or releases normally will not be issued.

Patches will be attached to the relevant JIRA issue. You can use these patches as a "stop-gap" measure until you upgrade your installation in order to fully fix the vulnerability.

Non-critical vulnerabilities

When a security issue of a High, Medium or Low severity is discovered, Atlassian will do all of the following:

- Include the fix into the next scheduled release, both for the current and previous maintenance versions.
- Where practical, provide new versions of plugins or other components of the product that can be upgraded independently.

You should upgrade your installation in order to fix the vulnerability.

Other information

Severity level of vulnerabilities is calculated based on Severity Levels for Security Issues.

Visit our general Atlassian Patch Policy as well.

Examples

Example 1: A critical severity vulnerability is found in a (hypothetical current release) JIRA 5.3.2. The last bugfix release in 5.2.x branch was 5.2.3. In this case, a patch will be created for 5.3.2 and 5.2.3. In addition, new bugfix releases, 5.3.3 and 5.2.4, which are free from this vulnerability, will be created in a few days.

Example 2: A high or medium severity vulnerability is found in the same release as in the previous example. The fix will be included into the currently scheduled releases 5.3.3 and 5.2.4. Release schedule will not be brought forward and no patches will be issued. If the vulnerability is in a plugin module, then a plugin upgrade package may still be supplied.

Further reading
See Atlassian Support Offerings for more support-related information.

# Severity levels for security issues

## Severity Levels

Atlassian security advisories include a severity level. This severity level is based on our self-calculated CVSS score for each specific vulnerability. CVSS is an industry standard vulnerability metric. You can learn more about CVSS at FIRST.org web site.

CVSS scores are mapped into the following severity ratings:

- Critical
- High
- Medium
- Low

An approximate mapping guideline is as follows:

<table>
<thead>
<tr>
<th>CVSS score range</th>
<th>Severity in advisory</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 – 2.9</td>
<td>Low</td>
</tr>
<tr>
<td>3 – 5.9</td>
<td>Medium</td>
</tr>
<tr>
<td>6.0 – 7.9</td>
<td>High</td>
</tr>
<tr>
<td>8.0 – 10.0</td>
<td>Critical</td>
</tr>
</tbody>
</table>

Below is a summary of the factors which illustrate types of vulnerabilities usually resulting in a specific severity level. Please keep in mind that this rating does not take into account details of your installation.

### Severity Level: Critical

Vulnerabilities that score in the critical range usually have the following characteristics:

- Exploitation of the vulnerability results in root-level compromise of servers or infrastructure devices.
- The information required in order to exploit the vulnerability, such as example code, is widely available to attackers.
- Exploitation is usually straightforward, in the sense that the attacker does not need any special authentication credentials or knowledge about individual victims, and does not need to persuade a target user, for example via social engineering, into performing any special functions.

For critical vulnerabilities, is advised that you patch or upgrade as soon as possible, unless you have other mitigating measures in place. For example, if your installation is not accessible from the Internet, this may be a mitigating factor.

### Severity Level: High

Vulnerabilities that score in the high range usually have the following characteristics:

- The vulnerability is difficult to exploit.
- Exploitation does not result in elevated privileges.
- Exploitation does not result in a significant data loss.

### Severity Level: Medium

Vulnerabilities that score in the medium range usually have the following characteristics:
- Denial of service vulnerabilities that are difficult to set up.
- Exploits that require an attacker to reside on the same local network as the victim.
- Vulnerabilities that affect only nonstandard configurations or obscure applications.
- Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics.
- Vulnerabilities where exploitation provides only very limited access.

**Severity Level: Low**

Vulnerabilities in the low range typically have very little impact on an organisation's business. Exploitation of such vulnerabilities usually requires local or physical system access.

**Further reading**

See [Atlassian Support Offerings](https://developer.atlassian.com/display/STASHDEV/Building+from+Source+Code) for more support-related information.

**Building Stash from source**

**This page has moved!**


But you really wanted to build a plugin anyway, right?