



Documentation for Stash 2.9

Contents

Stash Documentation Home	5
Getting started	5
Using Stash in the enterprise	7
Supported platforms	8
Installing Stash on Windows	10
Running Stash as a Windows service	14
Installing Stash on Linux and Mac	16
Running Stash as a Linux service	20
Configuring JIRA integration in the Setup Wizard	24
Getting started with Git and Stash	30
Importing code from an existing project	35
Basic Git commands	37
Using Stash	38
Creating projects	39
Creating repositories	40
Creating personal repositories	41
Using repository hooks	43
Controlling access to code	44
Using branch permissions	45
Branch permission patterns	47
Using repository permissions	48
Using project permissions	49
Allowing public access to code	50
Workflow strategies in Stash	51
Using branches in Stash	52
Automatic branch merging	56
Using forks in Stash	58
Keeping forks synchronized	60
Using pull requests in Stash	62
Checks for merging pull requests	67
Permanently authenticating with Git repositories	68
Using SSH keys to secure Git operations	70
Creating SSH keys	73
Notifications	76
HipChat notifications	76
Markdown syntax guide	77
Requesting add-ons	81
Integrating Stash with Atlassian applications	83
JIRA integration	84
Bamboo integration	87
Administering Stash	88
Users and groups	89
External user directories	95
Connecting Stash to an existing LDAP directory	97
Connecting Stash to JIRA for user management	103
Delegating Stash authentication to an LDAP directory	105
Connecting Stash to Crowd	111
Global permissions	114
Setting up your mail server	115
Linking Stash with JIRA	116
JIRA FishEye-Stash Plugin compatibility	118
Using custom JIRA issue keys with Stash	120
Connecting Stash to an external database	121
Connecting Stash to MySQL	123
Connecting Stash to Oracle	125
Connecting Stash to PostgreSQL	128

Connecting Stash to SQL Server	131
Transitioning from jTDS to Microsoft's JDBC driver	135
Migrating Stash to another server	136
Specifying the base URL for Stash	138
Configuring the application navigator	138
Managing add-ons	139
POST service webhook for Stash	139
Audit logging in Stash	143
Audit events in Stash	144
Advanced actions	148
Scaling Stash	148
Scaling Stash for Continuous Integration performance	151
Stash production server data	153
Stash debug logging	157
Stash config properties	158
Enabling SSH access to Git repositories in Stash	178
Setting up SSH port forwarding	181
Proxying and securing Stash	184
Securing Stash with Tomcat using SSL	187
Integrating Stash with Apache HTTP Server	193
Securing Stash with Apache using SSL	197
Securing Stash behind nginx using SSL	201
Moving Stash to a different context path	203
Changing the port that Stash listens on	204
Running Stash with a dedicated user	205
Data recovery and backups	205
Releases	210
Stash upgrade guide	213
End of support announcements for Stash	222
Stash 2.9 release notes	223
Stash 2.8 release notes	227
Stash 2.7 release notes	234
Stash 2.6 release notes	239
Stash 2.5 release notes	244
Stash 2.4 release notes	249
Stash 2.3 release notes	254
Stash 2.2 release notes	258
Stash 2.1 release notes	260
Stash 2.1 changelog	264
Stash 2.0 release notes	265
Stash 2.0 changelog	269
Stash 1.3 release notes	273
Stash 1.3 changelog	278
Stash 1.2 release notes	280
Stash 1.2 change log	284
Stash 1.1 release notes	287
Stash 1.1 change log	290
Stash 1.0 release notes	292
Stash 1.0 change log	295
Stash 1.0.1 release notes	298
Stash 1.0 upgrade guide	299
Stash security advisories	301
Stash security advisory 2012-09-04	301
Git resources	302
Stash FAQ	303
How do I change the external database password	305
Stash home directory	305
Raising a request with Atlassian Support	307
Support policies	308
Bug fixing policy	309
How to report a security issue	309
New features policy	310

Patch policy	311
Security advisory publishing policy	312
Security patch policy	312
Severity levels for security issues	313
Building Stash from source	314
Contributing to the Stash documentation	314

Stash Documentation Home

Getting started

[Using Stash in the enterprise](#)

[Installing Stash on Linux and Mac](#)

[Installing Stash on Windows](#)

[Getting started with Git and Stash](#)

[Pull requests](#)

[Git resources](#)

See our [Git tutorials...](#)

Administering

[All administration topics](#)

[Supported platforms](#)

[Global permissions](#)

[External user directories](#)

[Securing Stash with HTTPS](#)

[Connecting to an external database](#)

See admin [Answers...](#)

Extending Stash

[Integrating with JIRA](#)

[Integrating with Bamboo](#)

[APIs and add-ons](#)

[Scaling Stash](#)

See add-on developer [Answers...](#)

[Latest release notes](#)

[Stash upgrade guide](#)

[See the QuickStart movie](#)

Getting started

Atlassian Stash is the on-premises Git repository management solution for enterprise teams. It allows everyone in your organisation to easily collaborate on your Git repositories.



This section describes how to install, set up and start using Stash.

Related pages:

- [Using Stash](#)
- [Administering Stash](#)
- [Stash FAQ](#)
- [Stash upgrade guide](#)

System requirements

- Stash is a Java application, and relies on the Git distributed version control system (DVCS). See our [Supported platforms](#) page to find out about system requirements.

Download and install Stash

- [Windows](#)
- [Mac](#)
- [Linux](#)

Use Git

Stash is all about managing Git repositories. If you still need to install Git locally, see the [Stash install](#) pages.

We have some information here to help get you up and running with Git:

- [Git Tutorials and Training](#)
- [Basic Git commands](#)
- [Permanently authenticating with Git repositories](#)
- [Using SSH keys to secure Git operations](#)
- [Git resources](#)

Work with projects

Stash manages related repositories as projects. Find out how to [set up projects](#) and [give your teams access](#) to those.

If you have existing projects that you want to manage in Stash, then you'll want to read [Importing code from an existing project](#).

Integrate Stash with other Atlassian applications

As a first step, see [JIRA integration](#) for information about using Stash with JIRA. To link Stash with JIRA, see [Configuring JIRA integration in the Setup Wizard](#).

If you want to see results from your continuous integration or build server in Stash, see [Bamboo integration](#).

Read more about using Stash

You are looking at the Stash documentation. Browse using the tree in the panel on the left, or use the search at the top right.

Atlassian blog posts:

- [Getting social with pull requests](#)
- [Enterprise Git the way you want it](#)
- [Linking your Stash Git repositories to Crucible for code reviews](#)
- [Simple, secure Git repository management for the enterprise](#)

Using Stash in the enterprise

Atlassian Stash is the Git code management solution for enterprise teams. It allows everyone in your organisation to easily collaborate on your Git repositories, while providing enterprise-grade support for:

- user authentication
- repository security
- integration with your existing databases and development environment.

This page describes best practice for using Stash in enterprise environments, that is with 500+ user licenses. Of course, much of this information is also applicable to other Stash installations.

On this page:

- [Platform requirements for hosting Stash](#)
- [Performance considerations with Stash](#)
- [Setting up Stash in a production environment](#)
- [Administering Stash in a production environment](#)

Watch the movie »

Platform requirements for hosting Stash

Although Stash can be run on Windows, Linux and Mac systems, for enterprise use we only recommend, and support, [Linux](#). This recommendation is based on our own testing and experience with using Stash.

Please see the [Supported platforms](#) page for details of the supported versions of Java, external databases, web browsers and Git.

Performance considerations with Stash

In general, Stash is very stable and has low memory consumption. There are no scalability limits other than for Git hosting operations (clone in particular). We know this is the scalability limit of the product; the limit is proportional to the number of cores on the system.

As an example, data collected from an internal Stash instance indicate that for a team of approximately 50, with associated continuous integration infrastructure, we see a peak concurrency of 30 simultaneous clone operations and a mean of 2 simultaneous clone operations. We conservatively expect that a customer with similar usage patterns would be capable of supporting 1000 users on a machine with 40 cores and a supporting amount of ram. While we expect a peak concurrency larger than 40, Stash is designed to queue incoming requests so as to avoid overwhelming the server.

Please see [Scaling Stash](#) for more information about Stash performance and hardware requirements.

Setting up Stash in a production environment

When setting up Stash for a production or enterprise environment, please follow the instructions on the [Installing Stash on Linux and Mac](#) page. We highly recommend that you configure the following aspects:

Use an external database

- For production environments Stash should use an external database, rather than the embedded database. Set up your external DBMS (for example MySQL) before starting Stash for the first time. This allows you to connect Stash to that DBMS using the Setup Wizard that launches when you first run Stash.

See [Connecting Stash to an external database](#).

Connect to your existing user directory

- Connect Stash to your existing user directory (for example Active Directory). See [External user directories](#).

Secure the Stash home directory

- For production environments the Stash home directory should be secured against unauthorised access. See [Stash home directory](#).

Secure Stash with HTTPS

- 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](#).

Enable SSH access to Git repositories

- Enable SSH access for your Stash users to Git repositories in Stash so that they can add their own SSH keys to Stash, and then use those SSH keys to secure Git operations between their computer and the Stash server. See [Enabling SSH access to Git repositories in Stash](#).

Change the context path for Stash

- If you are running Stash behind a proxy, or you have another Atlassian application (or any Java web application), available at the same hostname and context path as Stash, then you should set a unique context path for Stash. See [Moving Stash to a different context path](#).

Administering Stash in a production environment

Upgrading Stash

- For production environments we recommend that you test the Stash upgrade on a QA server before deploying to production. See the [Stash upgrade guide](#).

Backups and recovery

- 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. See [Data recovery and backups](#) for information about using the Stash backup client.

Logging

- Stash server logs can be found in <STASH_HOME>/log. Logs for the bundled Tomcat webserver can be found in <Stash installation directory>/log. See [Stash debug logging](#).
- Stash displays recent audit events for each repository and project (only visible to Stash admins and system admins), and also creates full audit log files that can be found in the <Stash home directory>/audit/logs directory. Note that Stash has an upper limit to the number of log files it maintains, and deletes the oldest file when a new file is created – we recommend an automated backup of log files. See [Audit logging in Stash](#).



























Supported platforms

This page lists the supported platforms for **Stash 2.9.x**.

See [JIRA FishEye-Stash Plugin compatibility](#) for information about supported versions of JIRA.

Key:  = Supported  = Deprecated  = Not Supported

Java Version		
Oracle JDK	 1.7	<ul style="list-style-type: none"> • For the OracleJDK, you can download the Java SE Development Kit (JDK) from the Oracle website.
	 1.6	
	 1.5	

OpenJDK	<div><div> 1.7</div><div> 1.6</div><div> 1.5</div></div>	<ul style="list-style-type: none">For the OpenJDK, download and install instructions for Linux flavours are at http://openjdk.java.net/install/.Once the JDK is installed, you will need to set the <code>JAVA_HOME</code> environment variable, pointing to the root directory of the JDK. See Installing Stash on Windows or Installing Stash on Linux and Mac for details. As of Stash 2.4, support for Java 6 is now <i>deprecated</i>. Stash 3.0+ will require Java 7. If you are using Java 6, Stash requires at least 1.6.0_4 (See STASH-3708).
Operating Systems		
Apple Mac OS X	<div> (evaluation only)</div>	<ul style="list-style-type: none">Stash is a pure Java application and should run on any platform, provided all the JDK requirements are satisfied.In production environments Stash should be run from a dedicated user account. Although Stash can be run in virtualised environments, Atlassian is not yet able to provide technical support for performance-related problems in a virtualised environment. If you do chose to run Stash in a virtual machine, please ensure that you choose a VM with good IO throughput.
Linux	<div></div>	
Microsoft Windows	<div> (Not supported for 500+ Enterprise tiers)</div>	
Databases		
HSQLDB	<div> (bundled; for evaluation only)</div>	<ul style="list-style-type: none">Please see connecting Stash to an external database. MySQL 5.6.x: Note that Stash <i>does not support</i> MySQL 5.6.x, because of bugs in its query optimizer (#68424, #69005). See Connecting Stash to MySQL for more information.
Microsoft SQL Server / Microsoft SQL Server Express	<div><div> 2005</div><div> 2008</div><div> 2008 R2</div><div> 2012</div></div>	
MySQL	<div><div> 5.1.x</div><div> 5.5.x</div><div> 5.6.x</div></div>	
Oracle	<div> 11g</div>	
PostgreSQL	<div> 8.2, 8.3, 8.4, 9.1, 9.2, 9.3</div>	
Web Browsers		
Chrome	<div> Latest stable version supported</div>	<ul style="list-style-type: none"> As of Stash 2.0, support for Internet Explorer 8 is <i>deprecated</i>. Some screens may not render correctly, and some functionality may not be available. Stash 3.0+ will require Internet Explorer 9+.
Firefox	<div> Latest stable version supported</div>	
Internet Explorer	<div><div> 10</div><div> 9</div><div> 8</div></div>	
Safari	<div> Latest stable version supported</div>	
DVCS Clients		

Git - server side	<div><div>✔</div>1.7.6+, 1.8.0 - 1.8.4.2</div> <div><div>✖</div>1.8.4.3+</div> <div><div>✖</div>Cygwin</div>	<ul style="list-style-type: none">• 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.• ⚠ Git 1.7.1 is not supported.• <div><div>✖</div>1.8.4.3+ and higher are not supported due to a critical bug in how symbolic refs are handled.</div>• <div><div>✖</div>Cygwin: Cygwin Git <i>is not supported</i> for use on Windows servers, regardless of version.</div>
Git - client side	<div><div>✔</div>1.6.6+</div>	
Additional tools		
Perl	<div><div>✔</div>5.8.8+</div>	
Mail Clients		
Apple Mail	<div><div>✔</div>Apple Mail 4</div>	
Gmail	<div><div>✔</div>Latest</div>	
iOS Devices	<div><div>✔</div>iPhone, iPad</div>	
Microsoft Outlook	<div><div>✔</div>Express, 2007, 2010</div>	
Outlook.com / Hotmail / Windows Live Mail	<div><div>✔</div>Latest</div>	

Hardware requirements

See [Scaling Stash](#) regarding the hardware requirements for your Stash installation.

Notes:

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.

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.

For [production environments](#), see also the additional steps at the end of this page.

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.



Cygwin Git is *not supported*. No internal testing is done on that platform, and many aspects of Stash's functionality (pull requests and forks among them) have known issues. When running Stash on Windows, *always* use `msysGit`.

Related pages:

- [Running Stash as a Windows service](#)
- [Installing Stash on Linux and Mac](#)
- [Getting started with Git and Stash](#)
- [Supported platforms](#)
- [Stash upgrade guide](#)

[Download latest version](#) ➔

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](#).

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

Stash 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 root directory of the Java installation. When running Stash on Windows, unlike Linux or Unix, JAVA_HOME paths with spaces are just fine.

▼ [If you don't see a path...](#)

If you don't see a path, or if you just see %JAVA_HOME%, then set JAVA_HOME 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 "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, and don't wrap the value in quotes.

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 versions of Git and Perl

In a command prompt, run:

```
git --version
perl --version
```

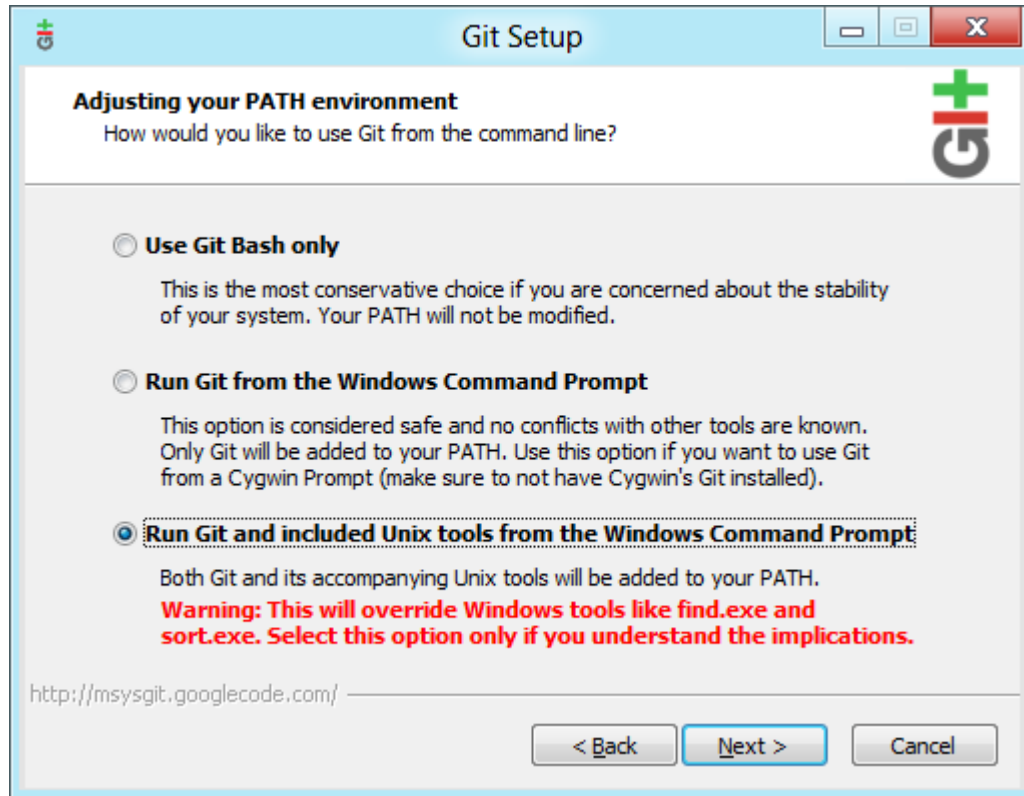
The version of Git should be **1.7.6** or higher. The version of Perl should be **5.8.8** or higher.

▼ [If you don't see supported versions of Git and Perl, then get them...](#)

Download the [Full installer for official Git for Windows](#). Installing Git for Windows (msysGit) also installs a supported version of Perl.

Run the Git installer. Ensure that `git.exe` is available in the path:

- Option 2, "Run Git from the Windows Command Prompt", or Option 3, "Run Git and included Unix tools from the Windows Command Prompt", will both work with Stash.
- Do *not* select Option 1, "Use Git Bash only" -- *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.

⚠ **msysGit is the *only supported distribution* when running Stash on Windows. Cygwin Git is *not supported* and has known issues.**

5. Now it's time to get Stash

Download Stash from the Atlassian download site. Looking for the [Stash WAR file](#)?

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.

Don't use spaces in the path to the Stash installation directory.

6. Tell Stash where to store your data

The Stash [home directory](#) is where your Stash data is stored.

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.

There are a few things to know about setting up the Stash home directory on Windows that will make life easier:

- 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.
- Keep the path length to the Stash home directory as short as possible. See [Pull request merges can fail when Stash is hosted on Windows](#) for an explanation.
- Don't use spaces in the path to the Stash home directory.

7. Start Stash!

The user that runs Stash shouldn't have admin privileges (see this [performance issue](#)), but must have read, write and execute access to the Stash home directory and the <Stash installation directory>.

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

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

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](#).
- Set up JIRA integration now, or 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. Add users and repositories

Now is the time to set up your users in Stash, and to tell Stash about any existing repositories you have. Please see the following pages for the details:

- [Getting started with Git and Stash](#)
- [Importing code from an existing project](#)

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. Please see [Using Stash in the enterprise](#) for more information about best practice.

Run Stash as a dedicated user

- For production environments Stash should be run from a dedicated user account with restricted privileges. See [Running Stash with a dedicated user](#).

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 Tomcat using SSL](#).

Run Stash as a Windows service

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

Connect to your existing user directory

- See [External user directories](#).

Change the context path for Stash

- Where you are running Stash behind a proxy, or you have another Atlassian application, or any Java web application, available at the same hostname and context path as Stash, then you should set a unique context path for Stash. See [Moving Stash to a different context path](#).

Stopping Stash (optional)

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

```
bin\stop-stash.bat
```

Uninstalling Stash

To uninstall Stash, stop Stash as described above and then delete the `<Stash installation directory>` and [Stash home directory](#).

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.

On this page:

- [Setting up Stash as a Windows service](#)
- [Troubleshooting](#)

Related pages:

- [Installing Stash on Windows](#)
- [Running Stash as a Linux service](#)

Before you start

- If you are using a 64-bit version of Windows, first ensure that Stash uses a 64-bit JVM (check by running `java -version` in a Command Prompt, and ensure that the `JAVA_HOME` system environment variable points to the 64-bit JVM), and then replace the 32-bit Tomcat binaries with their 64-bit counterparts in the `<Stash installation directory>/bin` directory:

```
cd <STASH-INST/bin>
rename tomcat7.exe tomcat7.exe.x86
rename tcnative-1.dll tcnative-1.dll.x86
rename tomcat7.exe.x64 tomcat7.exe
rename tcnative-1.dll.x64 tcnative-1.dll
```

- 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.
- Stash should be run from a local [dedicated user account](#) that does not have admin privileges and that has read, write and execute access to the Stash home directory and the `<Stash installation directory>`. See [Git Push Operations Extremely Slow on Windows](#).
- 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.
- If you upgraded Stash from version 1.x to 2.x and Stash stopped running as a service you will need to reinstall the service according to instructions in the [Stash upgrade guide](#).

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
> tomcat7 //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
> tomcat7 //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 768 Mb):

```
> tomcat7 //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-2.0.0\bin>service.bat install
Installing the service 'STASH' ...
Using CATALINA_HOME:      "C:\Program Files (x86)\atlassian-stash-2.0.0"
Using CATALINA_BASE:      "C:\Program Files (x86)\atlassian-stash-2.0.0"
Using JAVA_HOME:          "C:\Java\jre6"
Using JVM:                  "auto"
The service 'STASH' has been installed.
C:\Program Files (x86)\atlassian-stash-2.0.0\bin>tomcat7.exe //US//STASH --Startup
auto
C:\Program Files (x86)\atlassian-stash-2.0.0\bin>tomcat7.exe //US//STASH --JvmMx
1024

C:\Program Files (x86)\atlassian-stash-2.0.0\bin>net start STASH
The Atlassian Stash service is starting.
The Atlassian Stash service was started successfully.
```

Troubleshooting

- Problems may occur when trying to set up 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.

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.

If you are installing Stash for production or enterprise use, please read [Using Stash in the enterprise](#) first.

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](#)
- [Running Stash as a Linux service](#)
- [Getting started with Git and Stash](#)
- [Supported platforms](#)
- [Stash upgrade guide](#)
- [Using Stash in the enterprise](#)

See also:

- [Chef recipe for Stash](#)
- [Puppet/Vagrant How-To](#)

[Download latest version](#) ➞

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...](#)

Install 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.

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/`.

Set your JAVA_HOME if you don't see a path

Linux	Mac
<p>Do either of the following:</p> <ul style="list-style-type: none"> If <code>JAVA_HOME</code> is not set, log in with 'root' level permissions and run: <pre>echo JAVA_HOME="path/to/JAVA_HOME" >> /etc/environment</pre> <p>where <code>path/to/JAVA_HOME</code> may be like: <code>/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/</code></p> <ul style="list-style-type: none"> If <code>JAVA_HOME</code> needs to be changed, open the <code>/etc/environment</code> file in a text editor and modify the value for <code>JAVA_HOME</code> to: <pre>JAVA_HOME="path/to/JAVA_HOME"</pre> <p>It should look like:</p> <pre>JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/</pre>	<p>Open your <code>~/.profile</code> file in a text editor and insert:</p> <pre>JAVA_HOME="path/to/JAVA_HOME" export JAVA_HOME</pre> <p>where <code>path/to/JAVA_HOME</code> may be like: <code>/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home/</code></p> <p>Refresh your <code>~/.profile</code> in the terminal and confirm that <code>JAVA_HOME</code> is set:</p> <pre>source ~/.profile \$JAVA_HOME/bin/java -version</pre> <p>You should see a version of Java that is 1.6.0 or higher, like this:</p> <pre>java version "1.6.0_24"</pre>

3. Check your versions of Git and Perl

In a terminal, run this:

```
git --version
perl --version
```

The version of Git should be **1.7.6** or higher. The version of Perl should be **5.8.8** or higher.

▼ If you don't see supported versions of Git and Perl...

Download and install the latest stable Git release from the [Git website](#).

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](#) 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>.
- 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
```

Download and install the latest stable Perl release from <http://www.perl.org/get.html>, or use your favourite package manager.

Now try running 'perl --version' again. The version of Perl should be **5.8.8** or higher.

4. Now it's time to get Stash

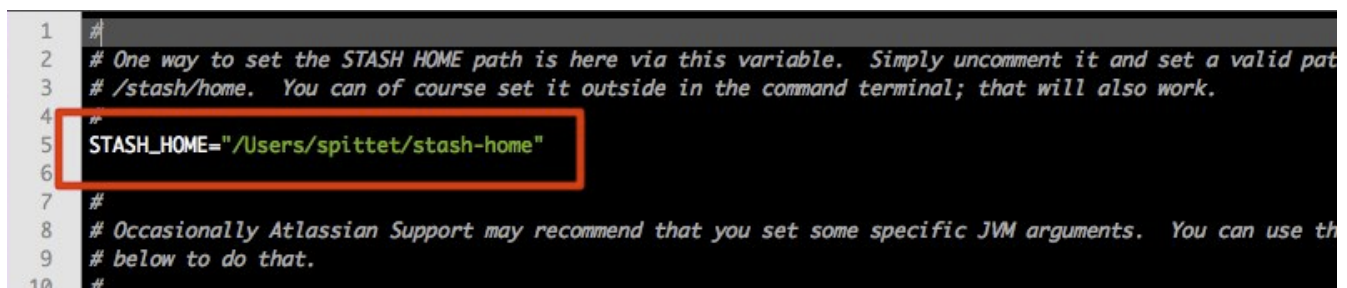
Download [Stash](#) from the Atlassian download site. Looking for the [Stash WAR file](#)?

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.

5. Tell Stash where to store your data

The Stash [home directory](#) is where your Stash data is stored.

Create your Stash home directory (without spaces in the name), 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:



```
1 #
2 # One way to set the STASH HOME path is here via this variable. Simply uncomment it and set a valid path
3 # /stash/home. You can of course set it outside in the command terminal; that will also work.
4 #
5 STASH_HOME="/Users/spittet/stash-home"
6 #
7 #
8 # Occasionally Atlassian Support may recommend that you set some specific JVM arguments. You can use them
9 # below to do that.
10 #
```

⚠ 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.

6. 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](#).

7. 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](#).

8. Add users and repositories

Now is the time to set up your users in Stash, and to tell Stash about any existing repositories you have. Please see the following pages for the details:

- [Getting started with Git and Stash](#)
- [Importing code from an existing project](#)

Additional steps for production environments

For production or enterprise environments we recommend that you configure the additional aspects below. These are not necessary when installing for evaluation purposes. Please see [Using Stash in the enterprise](#) for more information about best practice.

Run Stash as a dedicated user

- For production environments Stash should be run from a dedicated user account with restricted privileges. See [Running Stash with a dedicated user](#).

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 Tomcat using SSL](#).

Connect to your existing user directory

- See [External user directories](#).

Change the context path for Stash

- Where you are running Stash behind a proxy, or you have another Atlassian application (or any Java web application) available at the same hostname and context path as Stash, then you should set a unique context path for Stash. See [Moving Stash to a different context path](#).

Install Stash as a service

- See [Running Stash as a Linux service](#)

Stopping Stash (optional)

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


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

Uninstalling Stash

To uninstall Stash, stop Stash as described above and then delete the <Stash installation directory> and [Stash home directory](#).

Running Stash as a Linux service

 System administration tasks are **not supported by Atlassian**. These instructions are only provided as a guide and may not be up to date with the latest version of your operating system.

For production use on a Linux server, Stash should be configured to run as a Linux service, that is, as a daemon process. This has the following advantages:

- Stash can be automatically restarted when the operating system restarts.
- Stash can be automatically restarted if it stops for some reason.
- Stash is less likely to be accidentally shut down, as can happen if the terminal Stash was manually started in is closed.
- Logs from the Stash JVM can be properly managed by the service.

Related pages:

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

There are different approaches to running Stash as a service on Linux:

- Use the Java Service Wrapper which a Java Application to be run as a UNIX Daemon.
- Use an `init.d` script to start Stash at boot time - this doesn't restart Stash if it stops for some reason.

Note that Stash assumes that the external database is available when it starts; these approaches do not support service dependencies, and the startup scripts will not wait for the external database to become available.

On this page:

- [Using the Java Service Wrapper](#)
- [Using an `init.d` script](#)

Using the Java Service Wrapper

Stash can be run as a service on Linux using the [Java Service Wrapper](#). The Service Wrapper is [known to work with](#) Debian, Ubuntu, and Red Hat.

The Service Wrapper provides the following benefits:

- Allows Stash, which is a Java application, to be run as a service.
- No need for a user to be logged on to the system at all times, or for a command prompt to be open and running on the desktop to be able to run Stash.
- The ability to run Stash in the background as a service, for improved convenience, system performance and security.
- Stash is launched automatically on system startup and does not require that a user be logged in.
- Users are not able to stop, start, or otherwise tamper with Stash unless they are an administrator.
- Can provide advanced failover, error recovery, and analysis features to make sure that Stash has the maximum possible uptime.

Please see <http://wrapper.tanukisoftware.com/doc/english/launch-nix.html> for wrapper installation and configuration instructions.

The service wrapper supports the standard commands for SysV init scripts, so it should work if you just create a symlink to it from `/etc/init.d`.

Using an `init.d` script

The usual way on Linux to ensure that a process restarts at system restart is to use an `init.d` script. This approach does not restart Stash if it stops by itself.

1. [Stop Stash](#) .
2. Create a stash user, set the permissions to that user, create a home directory for Stash and create a symlink to make upgrades easier:

```
$> export VERSION="2.8.1" # Set this to the version you want to install
$> curl -OL
http://downloads.atlassian.com/software/stash/downloads/atlassian-stash-$VERSION.tar.gz
$> tar xz -C /opt -f atlassian-stash-$VERSION.tar.gz
$> ln -s /opt/atlassian-stash-$VERSION /opt/atlassian-stash-latest

# Create a home directory
$> mkdir /opt/stash-home

# ! Update permissions and ownership accordingly
```

3. Create the [startup script](#) in `/etc/init.d/stash` with the following contents (Ensure the script is executable by running `chmod 755 stash`) *Note: this is only an example, please adjust the script for your platform:*

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:          stash
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Initscript for Atlassian Stash
# Description:       Automatically start Atlassian Stash when the system starts up.
#                   Provide commands for manually starting and stopping Stash.
### END INIT INFO
# Adapt the following lines to your configuration
# RUNUSER: The user to run Stash as.
RUNUSER=stash
# STASH_INSTALLDIR: The path to the Stash installation directory
STASH_INSTALLDIR="/opt/atlassian-stash"
# STASH_HOME: Path to the Stash home directory
STASH_HOME="/opt/stash-home"
#
=====
====
#
=====
====
#
=====
====
# PATH should only include /usr/* if it runs after the mountnfs.sh script
PATH=/sbin:/usr/sbin:/bin:/usr/bin
DESC="Atlassian Stash"
NAME=stash
PIDFILE=$STASH_INSTALLDIR/work/catalina.pid
SCRIPTNAME=/etc/init.d/$NAME
# Read configuration variable file if it is present
[ -r /etc/default/$NAME ] && . /etc/default/$NAME
# Define LSB log_* functions.
# To be replaced by LSB functions
# Defined here for distributions that don't define
# log_daemon_msg
log_daemon_msg () {
    echo $@
```

```

}
# To be replaced by LSB functions
# Defined here for distributions that don't define
# log_end_msg
log_end_msg () {
    retval=$1
    if [ $retval -eq 0 ]; then
        echo "."
    else
        echo " failed!"
    fi
    return $retval
}
# Depend on lsb-base (>= 3.0-6) to ensure that this file is present.
. /lib/lsb/init-functions

run_with_home() {
    if [ "$RUNUSER" != "$USER" ]; then
        su - "$RUNUSER" -c "export
STASH_HOME=${STASH_HOME};${STASH_INSTALLDIR}/bin/$1"
    else
        export STASH_HOME=${STASH_HOME};${STASH_INSTALLDIR}/bin/$1
    fi
}
#
# Function that starts the daemon/service
#
do_start()
{
    run_with_home start-stash.sh
}
#
# Function that stops the daemon/service
#
do_stop()
{
    if [ -e $PIDFILE ]; then
        run_with_home stop-stash.sh
    else
        log_failure_msg "$NAME is not running."
    fi
}

case "$1" in
start)
    [ "$VERBOSE" != no ] && log_daemon_msg "Starting $DESC" "$NAME"
    do_start
    case "$?" in
        0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
        2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
    esac
    ;;
stop)
    [ "$VERBOSE" != no ] && log_daemon_msg "Stopping $DESC" "$NAME"
    do_stop
    case "$?" in
        0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
        2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
    esac
    ;;
status)
    if [ ! -e $PIDFILE ]; then
        log_failure_msg "$NAME is not running."
    fi

```

```
        return 1
    fi
    status_of_proc -p $PIDFILE "" $NAME && exit 0 || exit $?
    ;;
restart|force-reload)
    #
    # If the "reload" option is implemented then remove the
    # 'force-reload' alias
    #
    log_daemon_msg "Restarting $DESC" "$NAME"
do_stop
case "$?" in
    0|1)
        do_start
        case "$?" in
            0) log_end_msg 0 ;;
            1) log_end_msg 1 ;; # Old process is still running
            *) log_end_msg 1 ;; # Failed to start
        esac
        ;;
    *)
        # Failed to stop
        log_end_msg 1
        ;;
esac
;;
*)
    echo "Usage: $SCRIPTNAME {start|stop|status|restart|force-reload}" >&2
```

```
    exit 3
;;
esac
```

4. To start on system boot, add the script to the start up process. For Ubuntu (and other Debian derivatives) use:

```
update-rc.d stash defaults
```

For RHEL (and derivatives) use:

```
chkconfig --add stash --level 0356
```

Note: You may have to install the `redhat-lsb` package on RHEL (or derivatives) to provide the LSB functions used in the script.

You may also be interested in an alternative startup script optimised for usage in RHEL based systems: <https://answers.atlassian.com/questions/218946/stash-system-v-init-script-for-rhel>

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

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 Stash 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:

JIRA base URL	The web address of your JIRA server. Examples are: <code>http://www.example.com:8080/jira/</code> <code>http://jira.example.com</code>
JIRA admin username	The credentials for a user with the 'JIRA System Administrators' global permission in JIRA.
JIRA password	
Stash base URL	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.

We recommend that you make use of the automatic back-linking from JIRA to Stash.

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.

Symptom	Cause	Solution
<p>The setup wizard displays one of the following error messages:</p> <ul style="list-style-type: none"> Failed to create application link from JIRA server at <URL> to this <application> server at <URL>. Failed to create application link from this <application> server at <URL> to JIRA server at <URL>. Failed to authenticate application link from JIRA server at <URL> to this <application> server at <URL>. Failed to authenticate application link from <application> server at <URL> to this JIRA server at <URL>. 	<p>The setup wizard failed to complete registration of the peer-to-peer application link with JIRA. JIRA integration is only partially configured.</p>	<p>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>
<p>The setup wizard displays one of the following error messages:</p> <ul style="list-style-type: none"> Failed to register <application> configuration in JIRA for shared user management. Received invalid response from JIRA: <response> Failed to register <application> configuration in JIRA for shared user management. Received: <response> 	<p>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.</p>	<p>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>

<p>The setup wizard displays the following error message:</p> <ul style="list-style-type: none"> Error setting Crowd authentication 	<p>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 <code>config.xml</code> file. This may be caused by a problem in your environment, such as a full disk.</p>	<p>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.</p>
<p>The setup wizard displays the following error message:</p> <ul style="list-style-type: none"> Error reloading Crowd authentication 	<p>The setup wizard has completed the integration of your application with JIRA, but is unable to start synchronizing the JIRA users with your application.</p>	<p>Restart your application. You should then be able to continue with the setup wizard. If this solution does not work, please contact Atlassian Support.</p>
<p>The setup wizard displays the following error message:</p> <ul style="list-style-type: none"> An error occurred: <code>java.lang.IllegalStateException: Could not create the application in JIRA/Crowd (code: 500)</code>. Please refer to the logs for details. 	<p>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 <code><application></code> URL.</p>	<p>Remove the partial configuration if it exists, try the 'Connect to JIRA' step again, and then continue with the setup. Detailed instructions are below.</p>
<p>No users can log in after you have set up the application with JIRA integration.</p>	<p>Possible causes:</p> <ul style="list-style-type: none"> 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. 	<p>Go to JIRA and add some usernames to the group.</p> <ul style="list-style-type: none"> 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. <p>If this solution does not work, please contact Atlassian Support.</p>

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:

- Log in to JIRA as a user with the '**JIRA System Administrators**' global permission.
- Click the '**Administration**' link on the JIRA top navigation bar.
- Remove the application link from JIRA, if it exists:
 - Click '**Application Links**' in the JIRA administration menu. The 'Configure Application Links' page will appear, showing the application links that have been set up.
 - 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 '**Applic**

- ation 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:

```
<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.
- 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 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 '**A**

- **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](#).
 - **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** (under 'Security Settings').
 - b. Click **Setup JIRA/Crowd authentication**. Note, if LDAP authentication has already been set up, you will need to remove that before connecting to JIRA for user management.
 - c. Make the following settings:

Authenticate against	Select a JIRA instance
Application name and password	Enter the values that you defined for your application in the settings on JIRA.

JIRA URL	The web address of your JIRA server. Examples: <div><code>http://www.example.com:8080/jira/ http://jira.example.com</code></div>
Auto-add	Select Create a FishEye user on successful login so that your JIRA users will be automatically added as a FishEye user when they first log in.
Periodically synchronise users with JIRA	Select Yes to ensure that JIRA will synchronize all changes in the user information on a regular basis. Change the value for Synchronise Period if required.
When Synchronisation Happens	Select an option depending on whether you want to allow changes to user attributes from within FishEye.
Single Sign On	Select Disabled . SSO is not available when using JIRA for user management and if enabled will make the integration fail.

- d. Click **Next** and select at least one user group to be synchronised from JIRA. If necessary, you could create a new group in JIRA, such as 'fisheye-users', and select this group here.
- e. Click **Save**.
- For Stash:
 - a. Go to the Stash administration area.
 - 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 Stash 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.

Getting started with Git and Stash

Atlassian Stash is the Git repository management solution for enterprise teams. It allows everyone in your organisation to easily collaborate on 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 for interacting with them.

On this page:

- [Assumptions](#)
- [Add users to Stash and grant permissions](#)
- [Create your first project and share it with collaborators](#)
- [Create a repository and get your code into Stash](#)

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](#).
- You have Stash installed and running. See [Installing Stash on Linux and Mac](#) or [Installing Stash on Windows](#).

Please read [Git resources](#) or check out our [Git tutorials](#) for tips on getting started with Git.

Related pages:

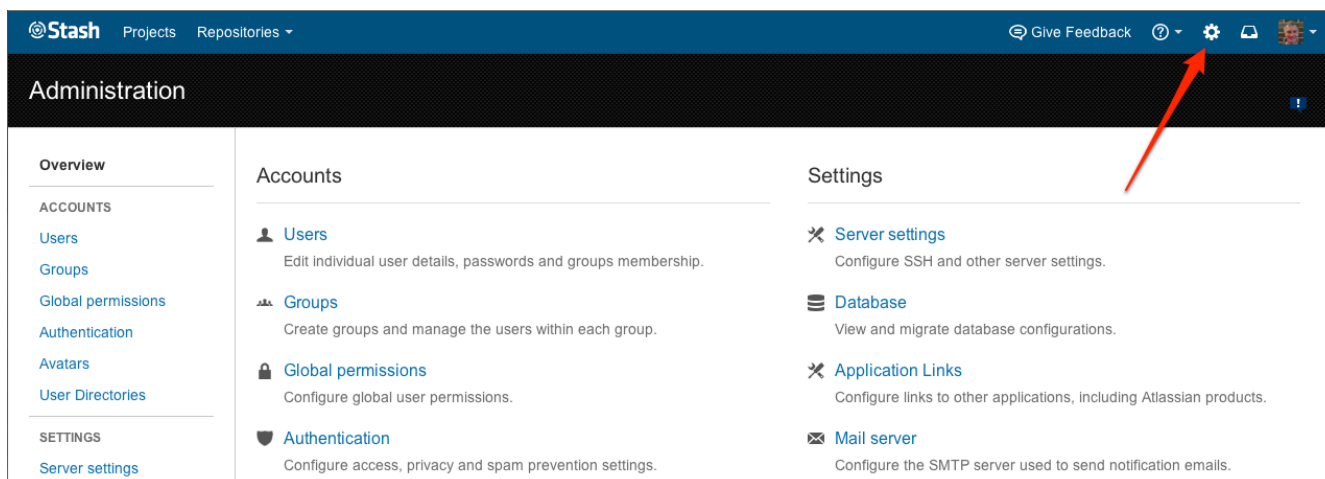
- [Installing Stash on Windows](#)
- [Installing Stash on Linux and Mac](#)
- [Importing code from an existing project](#)
- [Basic Git commands](#)
- [Git tutorials and training](#)

Watch the movie »

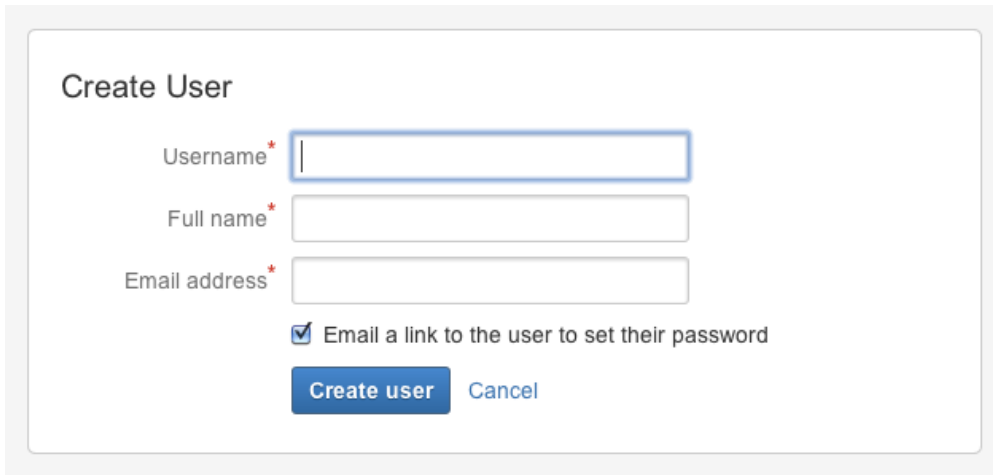
Add users to Stash and grant permissions

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

Go to the Stash administration area, by clicking the 'cog' menu in the header, and then click **Users** (under 'Accounts'):



Click **Create User** to go directly to the user creation form:



Create User

Username*

Full name*

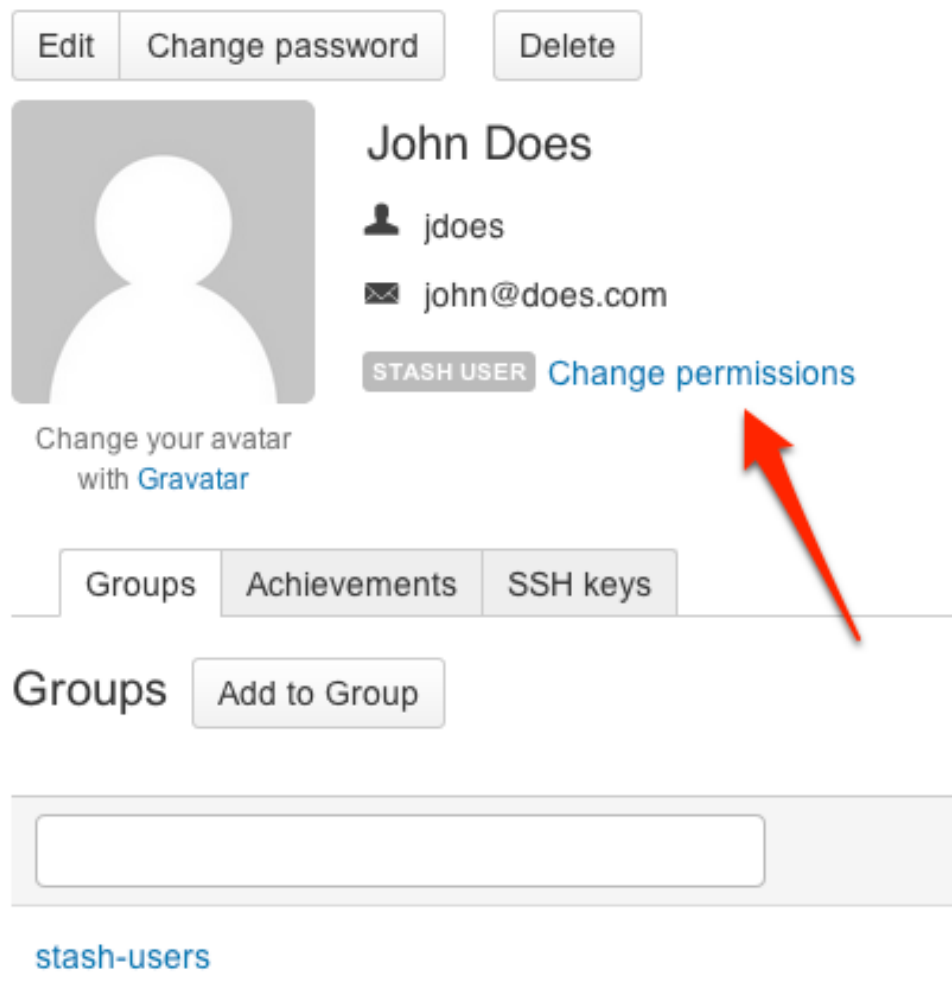
Email address*

☒ 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:


[← Back to Users](#)



Edit Change password Delete

 **John Does**

 jdoes

 john@does.com

STASH USER [Change permissions](#)

Change your avatar with [Gravatar](#)

Groups Achievements SSH keys

Groups [Add to Group](#)

[stash-users](#)

There are 4 levels of user authentication:

- **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.

See [Users and groups](#) for more information about authentication.

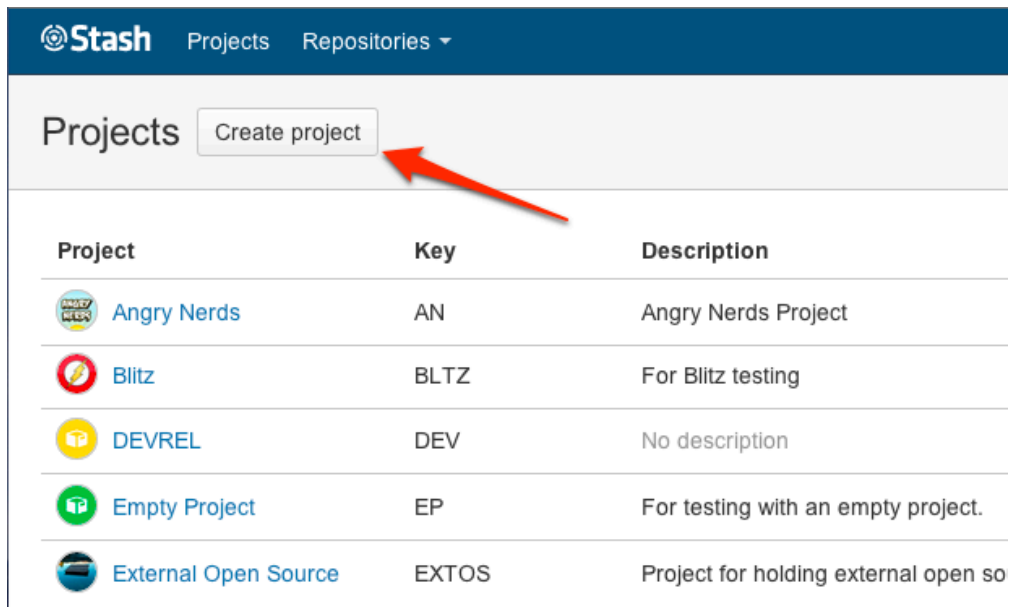
See [External user directories](#) if you have existing user identities you wish to use with Stash.

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 go to 'Projects' and click **Create project**. (Initially, you won't see as many projects as shown in this screenshot.)



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

The screenshot shows the 'Create a Project' form. It has the following fields and controls:

- Project name***: A text input field containing 'Angry Nerds Mobile'.
- Project key***: A text input field containing 'ANM'. Below it is a hint: 'Eg. AT (for a project named Atlassian)'.
- Description**: A text area containing 'Spiking the mobile version of Angry Nerds.'.
- Project Avatar**: A circular icon of a box and a 'Change avatar' button.
- At the bottom, there are two buttons: 'Create project' (in blue) and 'Cancel'.

See [Creating projects](#) for more information.

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:

Angry Nerds Mobile [Create repository](#)

[Repositories](#) **Permissions** [Settings](#)

Project permissions

Default permission for this project: ☐ Write ☐ Read ☒ No access

Individual Users

Name	Admin	Write	Read
<input type="text" value="Add Users"/> Read Add			
Paul Watson	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Groups

Name	Admin	Write	Read
<input type="text" value="Add Groups"/> Read Add			

No groups have been given explicit permissions to this project

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

There are 3 levels of project access:

- **Admin** — can create, edit and delete repositories and projects, and configure permissions for projects.
- **Write** — can push to and pull from all the repositories in the project.
- **Read** — can only browse code and comments in, and pull from, the repositories in the project.

See [Using project permissions](#) for more information.

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:

Angry Nerds Mobile [Create repository](#)

[Repositories](#) **Permissions** [Settings](#)

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.

See [Creating repositories](#) for more information.

A simple clone and push

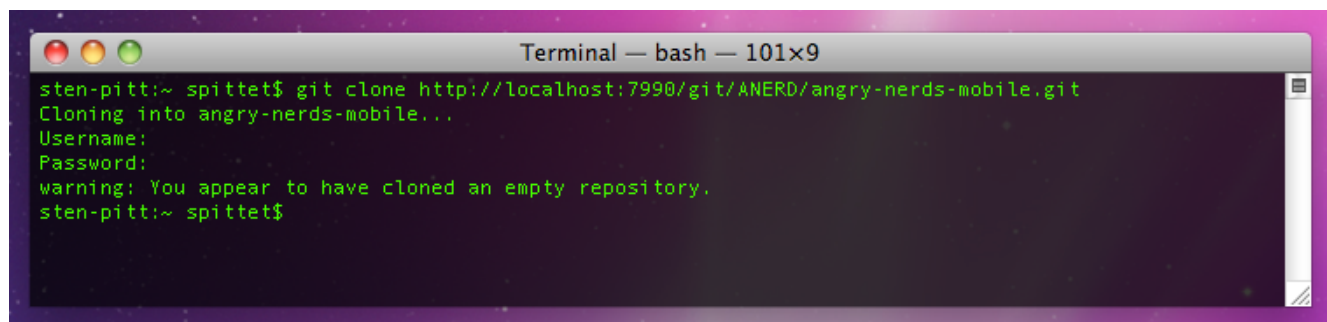
This section describes how to [clone the repository you just created](#) and then [push a commit](#) back to it. You can see the clone URL to use at the top right of the screen. [SSH access](#) may be available.

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

```
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

```
cd <reponame>
git add .
git commit -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.

For more information about getting your code into Stash, see [Importing code from an existing project](#). Note that huge Git repositories (larger than a few GBs) are likely to impact the performance of the Git client – see [this discussion](#).

Check out our [Git tutorials and training](#) for more information, and have a look at this list of [basic Git commands](#) that you will probably use often.

Importing code from an existing project

When creating a new repository, you can import code from an existing project into Stash. You can do this by first cloning the repository to your local system and then pushing to an empty Stash repository.

On this page:

- [Import an existing, unversioned code project to an empty repository](#)
- [Import a Git project to an empty repository](#)
- [Mirror an existing Git repository](#)

Import an existing, unversioned code project to an empty repository

If you have code on your local machine that is not under source control, you can put it under source control and import it into Stash. To do this:

1. Locally, change to the root directory of your existing source.
2. [Initialise the project](#) by running the following commands in the terminal:

```
git init
git add --all
git commit -m "Initial Commit"
```

3. Log into Stash and [create a new repository](#).
4. Locate the clone URL at the top right (eg.: `https://username@stash.atlassian.com/scm/PROJECT/repo.git`).
5. Push your files to the repository by running the following commands in the terminal (change the URL accordingly):

```
git remote add origin
https://username@stash.atlassian.com/scm/PROJECT/repo.git
git push -u origin master
```

6. Done! Your repository is now available in Stash.

Import a Git project to an empty repository

You can import an existing repository into an empty project in Stash. When you do this, Stash maintains your commit history.

1. [Check out the repository from your existing Git host](#). Use the `--mirror` parameter to include all branches and tags:

```
git clone --mirror
```

2. [Change the remote origin](#) in your local repository to point to Stash (change the URL accordingly):

```
git remote set-url origin
https://username@stash.atlassian.com/scm/PROJECT/repo.git
```

3. Then [push all branches](#) to Stash:

```
git push origin
```

Mirror an existing Git repository

You can mirror an existing repository into a repository hosted in Stash.

1. Check out the repository from your existing Git host. Use the `--mirror` parameter to include all branches and tags:

```
git clone --mirror
```

2. Add Stash as another remote in your local repository:

```
git remote add stash https://username@stash.atlassian.com/scm/PROJECT/repo.git
```

3. Then push all branches and tags to Stash


```
git push --all stash
git push --tags stash
```

4. Using `git fetch origin`, and the `git push` commands listed in step 3, the repository in Stash can be updated with changes from the upstream repository.

Basic Git commands

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

For more detail, check out the [Atlassian Git Tutorials](#) for a visual introduction to Git commands and workflows, including examples.

Git task	Notes	Git commands
Create a new local repository		<code>git init</code>
Check out a repository	Create a working copy of a local repository:	<code>git clone /path/to/repository</code>
	For a remote server, use:	<code>git clone username@host:/path/to/repository</code>
Add files	Add one or more files to staging (index):	<code>git add <filename></code> <code>git add *</code>
Commit	Commit changes to head (but not yet to the remote repository):	<code>git commit -m "Commit message"</code>
	Commit any files you've added with <code>git add</code> , and also commit any files you've changed since then:	<code>git commit -a</code>
Push	Send changes to the master branch of your remote repository:	<code>git push origin master</code>
Status	List the files you've changed and those you still need to add or commit:	<code>git status</code>
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:	<code>git remote add origin <server></code>
	List all currently configured remote repositories:	<code>git remote -v</code>
Branches	Create a new branch and switch to it:	<code>git checkout -b <branchname></code>
	Switch from one branch to another:	<code>git checkout <branchname></code>

	List all the branches in your repo, and also tell you what branch you're currently in:	<code>git branch</code>
	Delete the feature branch:	<code>git branch -d <branchname></code>
	Push the branch to your remote repository, so others can use it:	<code>git push origin <branchname></code>
	Push all branches to your remote repository:	<code>git push --all origin</code>
	Delete a branch on your remote repository:	<code>git push origin :<branchname></code>
Update from the remote repository	Fetch and merge changes on the remote server to your working directory:	<code>git pull</code>
	To merge a different branch into your active branch:	<code>git merge <branchname></code>
	View all the merge conflicts: View the conflicts against the base file:	<code>git diff</code> <code>git diff --base <filename></code>
	Preview changes, before merging:	<code>git diff <sourcebranch> <targetbranch></code>
	After you have manually resolved any conflicts, you mark the changed file:	<code>git add <filename></code>
Tags	You can use tagging to mark a significant changeset, such as a release:	<code>git tag 1.0.0 <commitID></code>
	CommitID is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:	<code>git log</code>
	Push all tags to remote repository:	<code>git push --tags origin</code>
Undo local changes	If you mess up, you can replace the changes in your working tree with the last content in head: Changes already added to the index, as well as new files, will be kept.	<code>git checkout -- <filename></code>
	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:	<code>git fetch origin</code> <code>git reset --hard origin/master</code>
Search	Search the working directory for <code>foo()</code> :	<code>git grep "foo()"</code>

Using Stash

Stash is the on-premises Git repository management solution for enterprise teams. It allows everyone in your organisation to easily collaborate on your Git repositories.

This section describes the essentials of using Stash.

If you are setting up Stash, see the [Getting started](#) section. If you want to configure Stash, see the [Administering Stash](#) section.

See [Getting started with Git and Stash](#) for an overview of how to work with Stash.

Related pages:

- [Getting started](#)
- [Git Tutorials and Training](#)
- [Git resources](#)
- [Administering Stash](#)
- [Stash FAQ](#)

Working with projects

Stash manages related repositories as projects. Find out how to [set up projects](#) and then [give your teams access](#) to those.

Working with repositories

If you have existing projects that you want to manage in Stash, then you'll want to read [Importing code from an existing project](#).

See also:

- [Creating repositories](#)
- [Controlling access to code](#)
- [Using pull requests in Stash](#)

Git resources

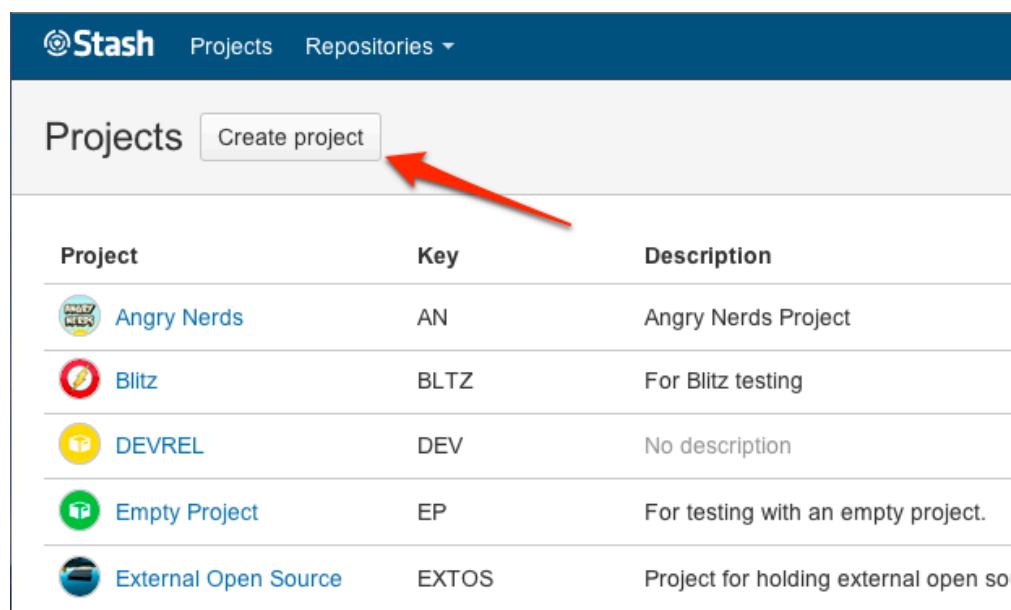
For those who are new to using Git:

- [Using pull requests in Stash](#)
- [Basic Git commands](#)
- [Permanently authenticating with Git repositories](#)






Creating projects

Projects allow you to group repositories and to [manage permissions](#) for them in an aggregated way.

To create a project, click on **Create project**:



The screenshot shows the Stash web interface. At the top, there's a navigation bar with the Stash logo and tabs for 'Projects' and 'Repositories'. Below this, the 'Projects' section is active, displaying a 'Create project' button with a red arrow pointing to it. Below the button is a table listing existing projects.

Project	Key	Description
 Angry Nerds	AN	Angry Nerds Project
 Blitz	BLTZ	For Blitz testing
 DEVREL	DEV	No description
 Empty Project	EP	For testing with an empty project.
 External Open Source	EXTOS	Project for holding external open so

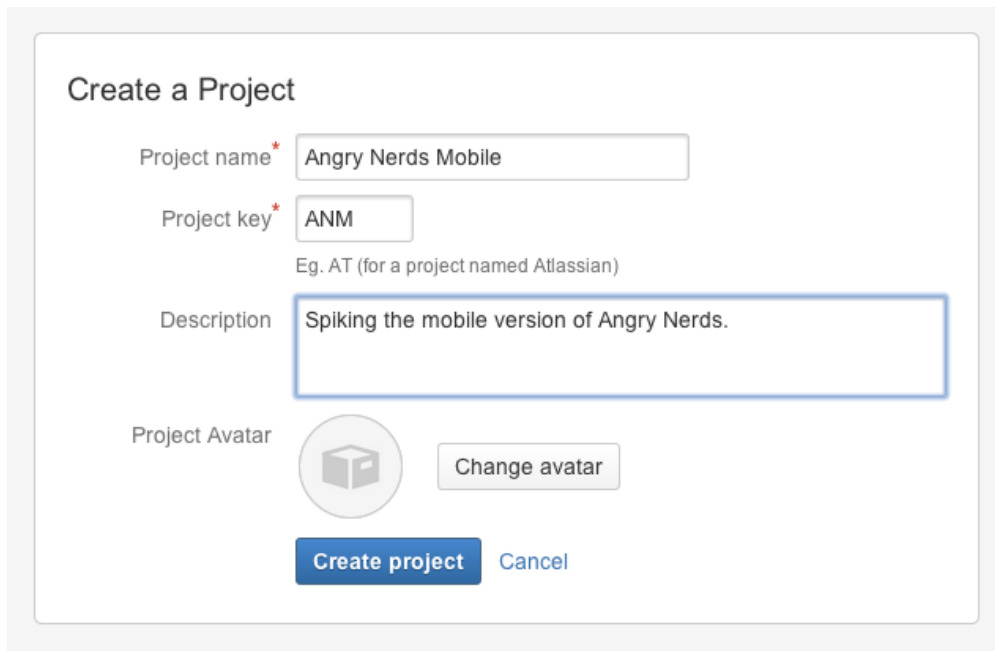
Related pages:

- [Getting started with Git and Stash](#)
- [Using project permissions](#)
- [Creating repositories](#)
- [Global permissions](#)

Fill out the form. 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.

Optionally, you can choose an avatar for the project. This is displayed throughout Stash and helps to identify your project.

Click **Create project** when you're done.




Create a Project

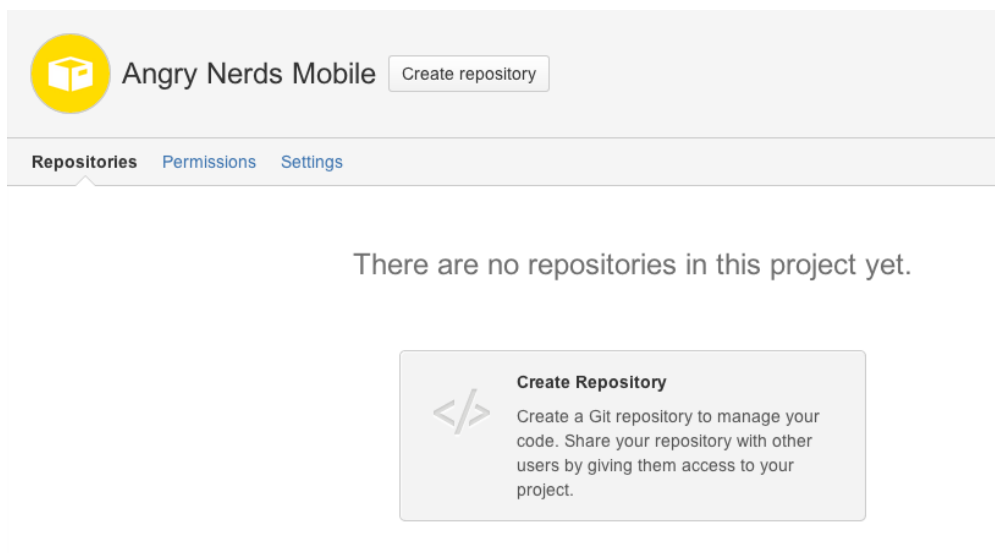
Project name*


Project key*
Eg. AT (for a project named Atlassian)

Description

Project Avatar 


You'll want to add repositories to the project. See [Creating repositories](#) for details.



 **Angry Nerds Mobile**

[Repositories](#) [Permissions](#) [Settings](#)

There are no repositories in this project yet.

**Create Repository**
Create a Git repository to manage your code. Share your repository with other users by giving them access to your project.

Creating repositories

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

In order to create repositories you need to have [Project Admin permission](#) for the project to which you want to add a repository.

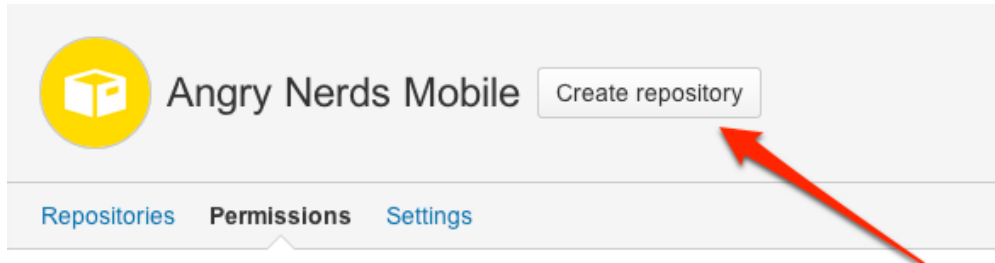
Once a repository is created, the project permissions are applied to the repository. That means all repositories

created in a project share the same access and permission settings.

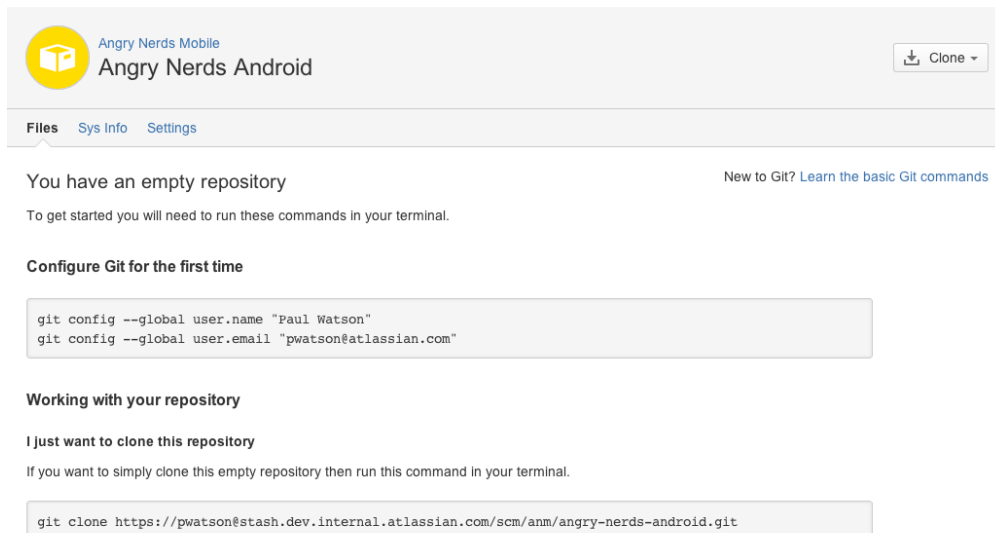
Related pages:

- [Creating personal repositories](#)
- [Using repository permissions](#)
- [Creating projects](#)
- [Importing code from an existing project](#)

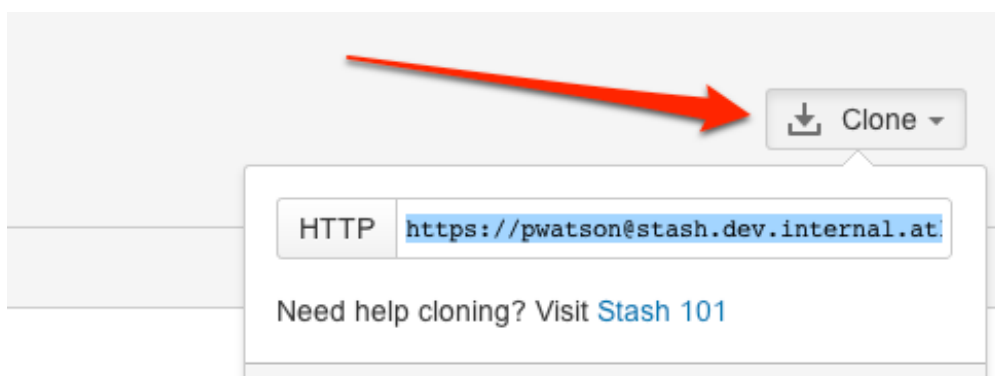
Click **Create repository** to open the repository creation form:



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 permissions at the parent project level. More information is available on [Creating projects](#).

Creating personal repositories

Stash allows you to create personal repositories, unrelated to other projects, that you can use for such purposes

as storing private snippets of work, kick-starting your own project, or contributing a bug-fix for a project you are not a member of.

By default, personal repositories are not visible to other Stash users. However, you can:

- use [repository permissions](#) to open up access to other Stash users and groups, for collaboration or review.
- allow [public access](#) (read-only) to your project, for anonymous users.

You can create personal repositories in 2 ways:

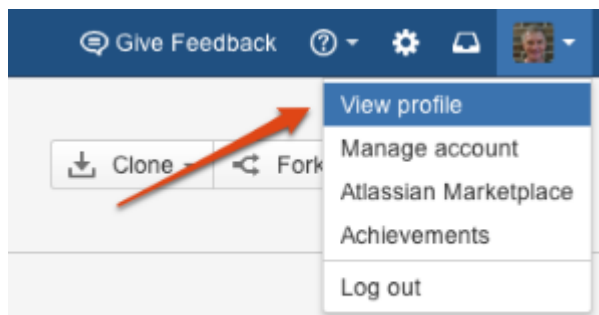
- [Directly](#), from your profile.
- By [forking](#) another repository.

Your personal repositories are listed on the **Repositories** tab of your profile page. Every Stash user can see your profile page, but they can only see those repositories that you have given them permission to view.

Directly creating a personal repository

You can create a personal repository at any time from your Stash profile:

1. Choose **View profile** from your user menu in the header.

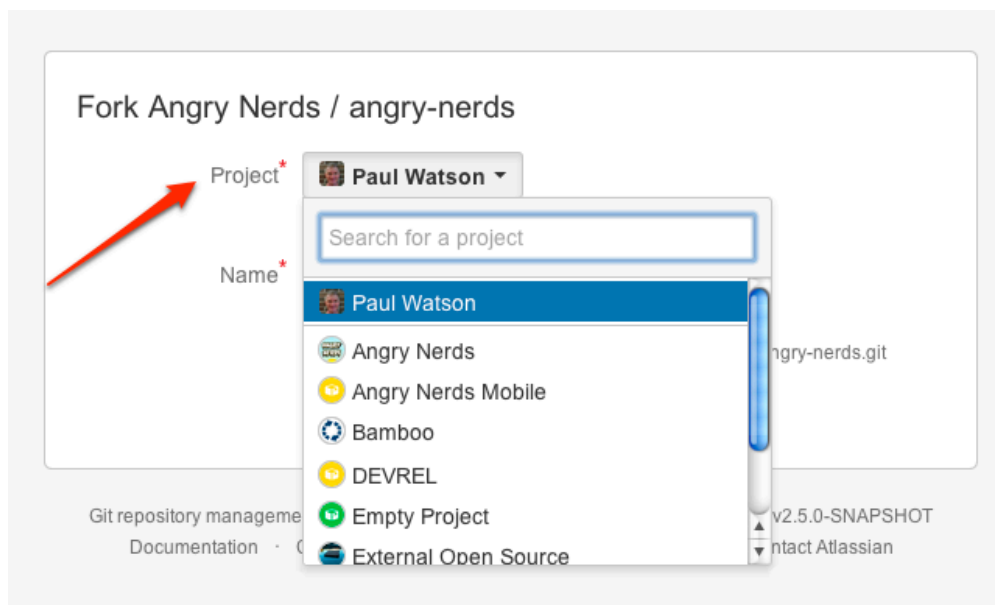


2. Click **Create repository**.
3. Set [repository permissions](#) on the new repository, if required.

Forking another repository

You can create a personal fork of any other repository in Stash for which you have [permission](#):

1. Go to the repository that you wish to fork.
2. Click **Fork**.
3. Choose your own profile (this is selected by default) from the **Project** list:



4. Click **Fork repository**.
5. Set [repository permissions](#) on the new repository, if required.

Using repository hooks

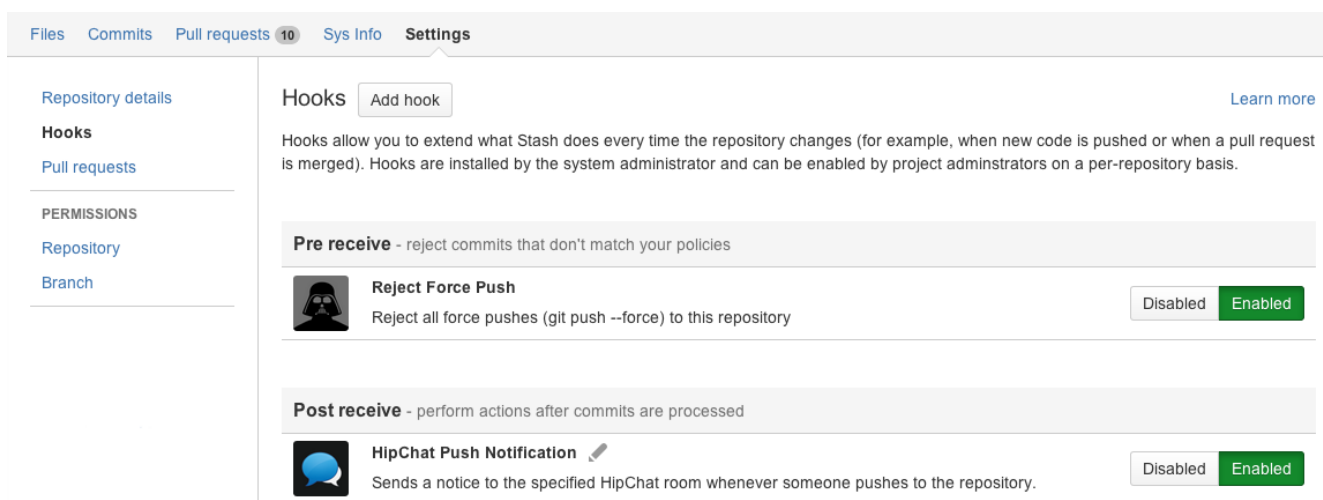
Hooks in Stash provide a way to customise a team's workflow and integrate with other systems. Stash currently supports two types of hooks, **pre** and **post-receive**.

On this page:

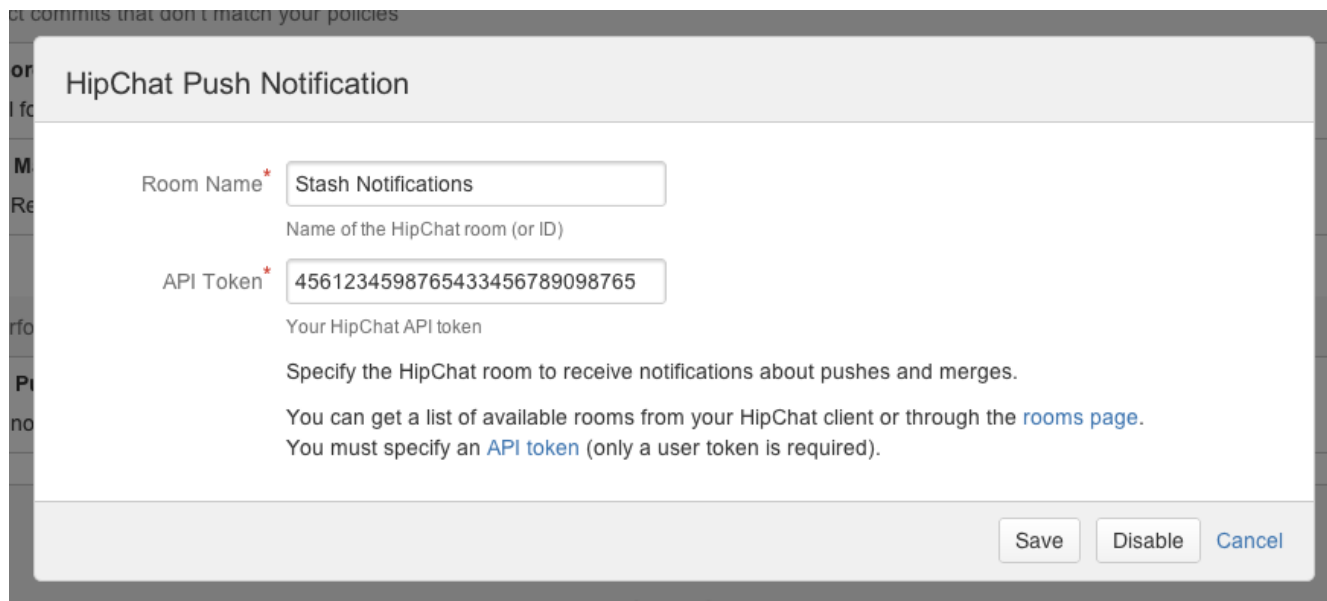
- [Managing hooks](#)
 - [Pre-receive hooks](#)
 - [Post-receive hooks](#)
- [Getting hooks from the Atlassian Marketplace](#)
- [Creating your own hooks](#)
- [Learn more about Git hooks](#)

Managing hooks

Administrators can see the hooks that are available in Stash by going to **Settings > Hooks** for a Stash repository. Once installed, hooks are available across all repositories in a Stash instance, but are enabled separately on each repository in a project.



Click the 'pen' icon beside the name of a hook to edit configuration details for the hook.



at commits that don't match your policies

HipChat Push Notification

Room Name*

Name of the HipChat room (or ID)

API Token*

Your HipChat API token

Specify the HipChat room to receive notifications about pushes and merges.

You can get a list of available rooms from your HipChat client or through the [rooms page](#).

You must specify an [API token](#) (only a user token is required).

Stash currently ships with the following hooks:

- **Reject Force Push** – block all Git force pushes (`git push -- force`).
- **HipChat Push Notifications** – send a message to a HipChat room when someone pushes to the repository.

Pre-receive hooks

The first hook to run when handling a push from a client is the pre-receive hook. It can reject pushes to the repository if certain conditions are not fulfilled. You can use this hook to prevent force pushes to the repository or check whether all commits contain a valid JIRA issue key.

Post-receive hooks

The post-receive hook runs after the commits have been processed and can be used to update other services or notify users. For example a post-receive hook can be used to send a message to a chat server or notify a continuous integration server of the newly pushed changes.

Getting hooks from the Atlassian Marketplace

A number of hooks are available from the [Atlassian Marketplace](#). You can find and install these from within Stash – simply use the **Add hook** button on the hooks settings page to view available hooks from the marketplace. See [Managing add-ons](#) for details.

Creating your own hooks

Developers can write receive hook plugins for Stash using a simple API that provides a simple way to create a configuration interface, and stores the hook's configuration settings on a per-repository basis.

For information about how to write your own hooks please see the [Stash developer docs](#).

In particular, these pages will be helpful:

- [Repository hooks](#)
- [Repository hook plugin module](#)

Learn more about Git hooks

Hooks in Stash are executed on the server, but Git itself also provides support for client-side hooks meant for client operations such as committing and merging. Learn more about Git hooks in the [Git documentation](#).

Note: Git post-receive hooks won't be triggered after a [pull request](#) merge. The mechanism that performs the pull request merge is actually based on a `git fetch into` the repository, which doesn't trigger Git post-receive hooks. If you would like to trigger functionality based on a pull request merge, you should write a post-receive repository hook.

Controlling access to code

Stash provides the following types of permissions to allow fully customisable control of access to code:

Global permissions

- Control user and group access to Stash projects and to the Stash server configuration.
- For example, these can be used to control the number of user accounts that can access Stash for licensing purposes.
- See [Global permissions](#).

Project permissions

- Apply the same access permissions to all repositories in a project.
- For example, these can be used to define the core development team for a project.
- See [Using project permissions](#).

Repository permissions

- Extend access to a particular repository for other, non-core, users.
- For example, these can be used to allow external developers or consultants access to a repository for special tasks or responsibilities.
- See [Using repository permissions](#).

Branch permissions

- Control commits to specific branches within a repository.
- For example, these can provide a way to enforce workflow roles such as the Release Manager, who needs to control merges to the release branch.
- See [Using branch permissions](#).

Note that you can also allow public (anonymous) access to projects and repositories. See [Allowing public access to code](#).

Using branch permissions

Branch permissions allow you to control who can commit to specific branches in a repository. Branch permissions provide another level of security within Stash (along with [user authentication](#) and [project, repository and global permissions](#)) that provides a way to control, or enforce, your own workflow or process.



Branch permissions:

- are based on users or groups.
- are actually restrictions, independent of project and repository level permissions, that limit branch access to specific people.
- prevent unauthorised users pushing to or deleting the branch.
- are based on explicit branch names, or you can use advanced branch permissions to match multiple branches (or tags) using [pattern matching](#).

For example, if two developers Xavier and Yves have write access to repository R, but only Xavier has branch permissions on branch B, then Yves won't be able to push to B.

If a user does not have commit access to the branch, an error message will be shown on the Git command line when they try to push a change to the branch.

Note that if no branch permissions are defined then anyone with commit access to the repository can push to any branch. Also, if there are conflicting permissions, the most permissive one applies; for example if one permission restricts a particular users access but another permission allows it, then the user will be allowed commit access.

On this page:

- [Setting branch permissions](#)
- [Advanced branch permissions](#)

Related pages:

- [Using pull requests in Stash](#)
- [Controlling access to code](#)
- [Global permissions](#)

Setting branch permissions

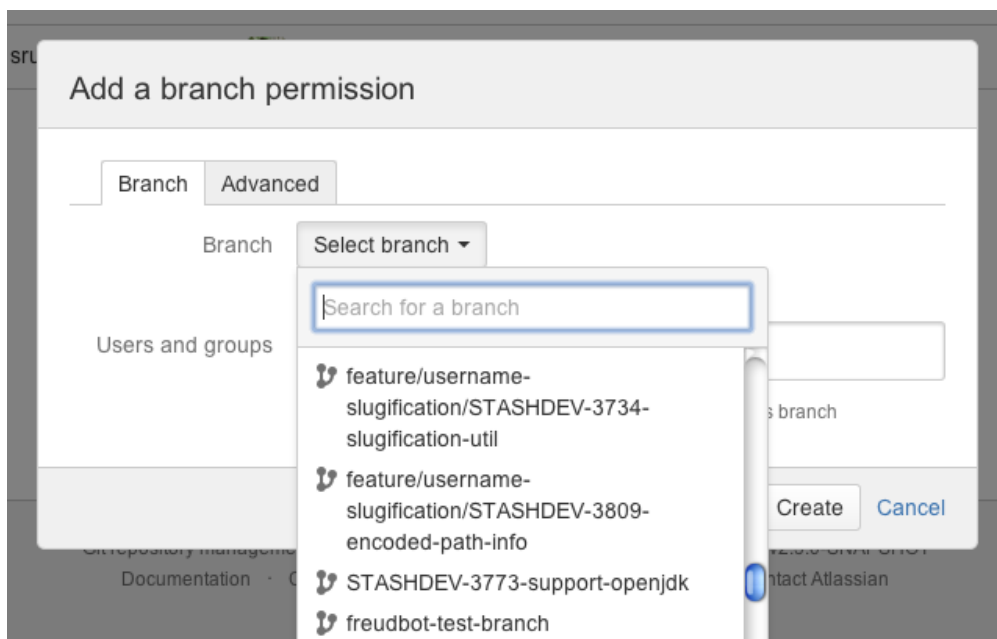
Branch permissions in Stash are set on a per-repository basis. Makes sense – branch permissions control access to *repository* branches, right?

You'll need either project admin, admin or sys-admin [permissions](#) to set branch permissions.

So, to set branch permissions:

1. Go to a repository in a project.
2. Choose **Settings** > **Branch** (under 'Permissions').
3. Click **Add permission**.
4. On the **Branch** tab, choose the branch for which you want to control access.
5. Add (or remove) users or groups that you want to have (or not have) commit access to the branch.
6. Click **Create** to finish.

You can always change the permissions for a branch later, if necessary.



Advanced branch permissions

Advanced branch permissions specify a pattern that is matched against branches *and* tags being pushed to Stash; this allows you to restrict any pushes to branches that match the pattern.

Advanced branch permission also apply to attempts to create new branches; if a push to Stash attempts to create a new branch that matches a pattern, the user must be authorised for the operation to proceed.

To set advanced branch permissions, choose **Settings > Branch**, and click **Add permission**, as described above.

On the **Advanced** tab, enter a [glob pattern](#) to match the names of multiple branches for which you want to control access.

Branch permission patterns

Stash supports a powerful type of pattern syntax for matching branch names (similar to pattern matching in Apache Ant).

These expressions use the following wild cards:

?	Matches one character (any character except path separators)
*	Matches zero or more characters (not including path separators)
**	Matches zero or more <i>path segments</i> .

Pattern used in branch permissions match against all refs pushed to Stash (i.e. branches and tags).

In git, branch and tag names can be nested in a namespace by using directory syntax within your branch names, e.g. `stable/1.1`. The `'**'` wild card selector enables you to match arbitrary directories.

- A pattern can contain any number of wild cards.
- If the pattern ends with `/` then `**` is automatically appended - e.g. `foo/` will match any branches or tags containing a `foo` path segment
- Patterns only need to match a suffix of the fully qualified branch or tag name. Fully qualified branch names look like `refs/heads/master`, whilst fully qualified tags look like `refs/tags/1.1`.

Also see the [Ant documentation](#).

Examples

*	Matches everything
PROJECT-*	Matches and branch or tag named PROJECT-*, even in a name space. e.g. <code>refs/heads/PROJECT-1234</code> , <code>refs/heads/stable/PROJECT-new</code> or <code>refs/tags/PROJECT-1.1</code>

?.?	Matches any branch or tag of 2 characters separated by a '.'. e.g. refs/heads/1.1, refs/heads/stable/2.X or refs/tags/3.1
tags/ or tags/**	Matches all tags and any branches with 'tags' as a namespace. e.g. refs/heads/stable/tags/some_branch, refs/tags/project-1.1.0
heads/**/master	Matches all branches called master. e.g. refs/heads/master, refs/heads/stable/master

Using repository permissions

Stash allows you to manage the permissions for just a single repository, or for a group of repositories together from the [project](#).

Repository permissions allow you to extend access to a repository, for those who don't have project permissions. For example, you might use repository permissions to allow external developers or consultants access to a repository for special tasks or responsibilities.

Stash supports 3 levels of permissions for repositories:

- Admin
- Write
- Read

Depending on the permission level for the repository that has been granted to you, you can perform different actions in the repository:

Related pages:

- [Using project permissions](#)
- [Using branch permissions](#)
- [Global permissions](#)
- [Allowing public access to code](#)

	Browse	Clone, fork, pull	Create, browse or comment on a pull request	Merge a pull request	Push	Edit settings and permissi ons
Admin	✓	✓	✓	✓	✓	✓
Write	✓	✓	✓	✓	✓	✗
Read	✓	✓	✓	✗	✗	✗

Granting access to a repository

To modify its permissions, go to the repository's settings and click on **Repository** (under 'Permissions'). Click in the **Add Users** or **Add Groups** field in the relevant section to search for, and bulk add, users or groups. Now choose a permission from the list and click **Add**.

Once added, you can use the checkboxes to edit specific permissions for an individual user or a particular group.

Granting access to all repositories within a project

If you have a large number of repositories in a project, [project level permissions](#) provide a convenient way

to grant access to *all* repositories within that project. For example you can grant a group, say "Team A", *Write* access at the project level, which will automatically give them *Write* access to all existing repositories in the project, as well as any repositories that are subsequently created in the project.

To modify permissions for a project, click the **Permissions** tab when viewing the project. You can add, or modify, permissions for individual users, and groups, in the same way as described above for a single repository.

Granting permission to create repositories

Only users with [project administration](#) permission can create *new repositories*.

Using project permissions

Stash allows you to manage the permissions for the repositories in a project in an aggregated way.

There are 3 levels of project permission that you can assign to a user or group for a project: *Admin*, *Write* and *Read*.

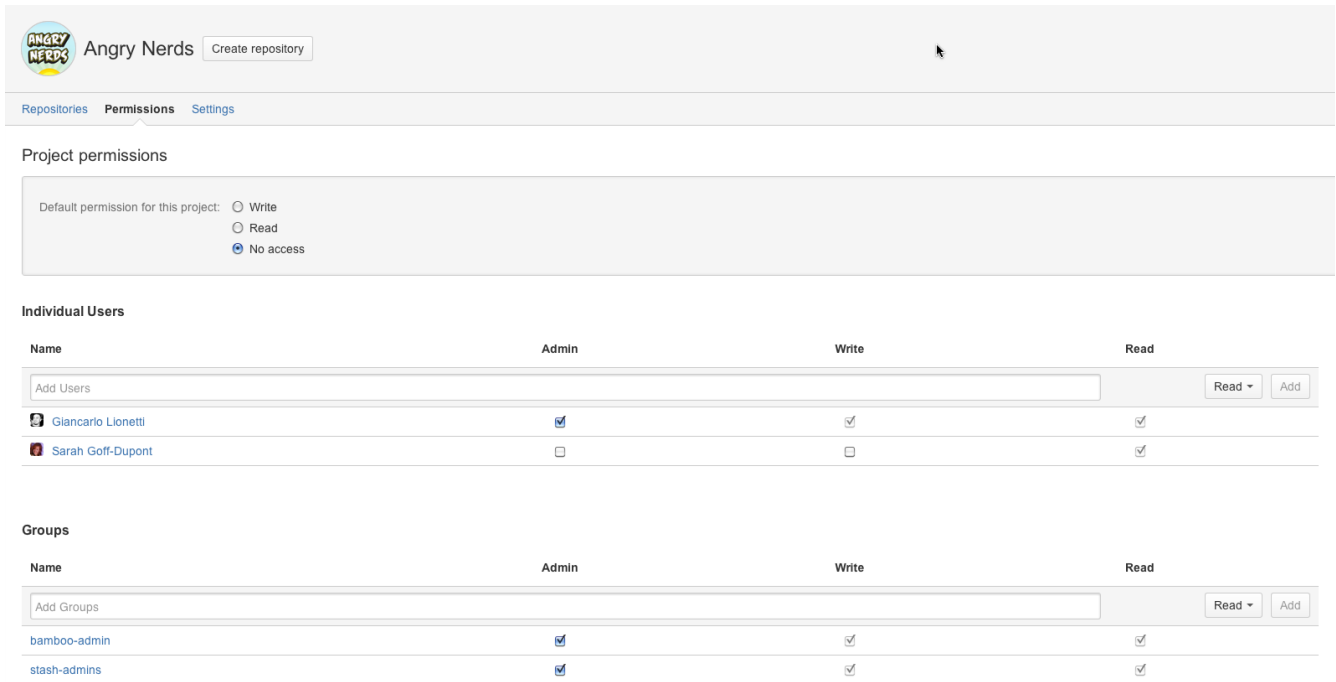
Related pages:

- [Creating projects](#)
- [Global permissions](#)
- [Using branch permissions](#)
- [Using repository permissions](#)
- [Allowing public access to code](#)

	Browse	Clone / Pull	Create, browse, comment on pull request	Merge pull request	Push	Create repositories	Edit settings / permissions
Project Admin	✓	✓	✓	✓	✓	✓	✓
Write	✓	✓	✓	✓	✓	✗	✗
Read	✓	✓	✓	✗	✗	✗	✗

To modify permissions for a project, click the **Permissions** tab when viewing the project. Click in the **Add Users** or **Add Groups** field in the relevant section to search for, and bulk add, users or groups. Now choose a permission from the list and click **Add**.

Once added, you can use the checkboxes to edit specific permissions for individual users or particular groups.



Angry Nerds [Create repository](#)

[Repositories](#) [Permissions](#) [Settings](#)

Project permissions

Default permission for this project: ☐ Write ☐ Read ☒ No access

Individual Users

Name	Admin	Write	Read
<input type="text" value="Add Users"/>			Read Add
Giancarlo Lionetti	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Sarah Goff-Dupont	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Groups

Name	Admin	Write	Read
<input type="text" value="Add Groups"/>			Read Add
bamboo-admin	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
stash-admins	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Allowing public access to code

You can open up public access for anonymous (unauthenticated) users to projects and repositories in Stash. This allows you to:

- Broadcast your repositories to a wider audience who generally don't have access to your source.
- Utilise unauthenticated cloning of repositories when setting up continuous integration servers to work with Stash.
- Link from other systems, for example JIRA or Confluence, to give users access to code without requiring authentication.
- Create open-source projects or repositories.

Public access allows anonymous users to browse the files, pull requests and commits for a specific repository or an entire project, and to clone repositories, without needing to log in, or have an account in Stash.

In Stash, you can:

- Configure a specific repository for public access.
- Configure a project to allow public access to all repositories in the project.
- Disable anonymous access by setting a global system property.

On this page:

- [Making a repository publicly accessible](#)
- [Making a project publicly accessible](#)
- [Viewing public repositories](#)
- [Disabling public access globally](#)

Related pages:

- [Using project permissions](#)
- [Using repository permissions](#)

Making a repository publicly accessible

You can open up an specific repository for public (anonymous) access.

You need admin permission for the repository.

Go to the repository and choose **Settings**, then **Repository** (under 'Permissions'). Check **Allow public access** to allow users to clone and browse the repository.

Making a project publicly accessible

You can open up a whole project (but not a private project) for public (anonymous) access.

You need admin permission for the repository, or its project.

Go to the project and choose **Settings**, then **Permissions**. Check **Allow public access** to allow users to clone and browse any repository in the project.

Viewing public repositories

Stash displays a list of repositories for which anonymous access has been enabled.

Anonymous and logged-in users can choose **Repositories** > **View all public repositories** to see these.

Disabling public access globally

Stash provides a [system property](#) that allows you to turn off public access for the whole instance.

To do this, set the `feature.public.access` property to `false` in the `stash.config.properties` file in your [Stash home directory](#).

Workflow strategies in Stash

Various Git workflows are supported by Stash:

- [Centralized Workflow](#)
- [Feature Branch Workflow](#)
- [Gitflow Workflow](#)
- [Forking Workflow](#)

For information about setting up Git workflows in Stash see [Using branches in Stash](#) and [Using forks in Stash](#).

Centralized Workflow

Like Subversion, the Centralized Workflow uses



a central repository to serve as the single point-of-entry for all changes to the project. Instead of `trunk`, the default development branch is called `master` and all changes are committed into this branch. This workflow doesn't require any other branches besides `master`.

[Read more about the Centralized Workflow...](#)

Feature Branch Workflow

The core idea behind the Feature Branch Workflow is that all feature development should take place in a dedicated branch instead of the `master` branch. This encapsulation makes it easy for multiple developers to work on a particular feature without disturbing the main codebase. It also means the `master` branch will never contain broken code, which is a huge advantage for continuous integration environments.

[Read more about the Feature Branch Workflow...](#)

Gitflow Workflow

The Gitflow Workflow defines a strict branching model designed around the project release. While somewhat more complicated than the Feature Branch Workflow, this provides a robust framework for managing larger projects.

[Read more about the Git Workflow...](#)

Forking Workflow

The Forking Workflow is fundamentally different than the other workflows discussed in this tutorial. Instead of using a single server-side repository to act as the "central" codebase, it gives every developer a server-side repository. This means that each contributor has not one, but two Git repositories: a private local one and a

public server-side one.

[Read more about the Forking Workflow...](#)

Using branches in Stash

Stash makes it easy for each member of your team to use a [branching workflow](#) for your Git development process. Your workflow can be mapped to branches in the Stash 'branching model', allowing Stash to:

- guide your developers into making consistent naming decisions when creating branches.
- identify the type of each branch and apply actions like automatic merging accordingly.



On this page:

- [Configuring the branching model](#)
- [Creating branches](#)
- [Automating the branch workflow](#)
- [Managing all your branches](#)

See also [Using branch permissions](#) for information about restricting access to branches in Stash.

Configuring the branching model

Stash uses a 'branching model' to define the branch workflow for each repository. As a project administrator, configuring the model lets you:

- enable the branch types that will be available in your workflow.
- specify the naming convention to be used for each branch type.

The naming convention simply adds prefixes to branch names, so that branches of the same type get the same prefix.

To configure the branching model for a repository, go to **Settings > Branching model** for the repository and click **Enable branching model**. Note that for *new* repositories, the branching model is enabled by default, and uses the default branch prefixes.

Stash makes a number of branch types available, as described below. Use the checkboxes to enable just those branch types that map to your workflow. Note that several branch types have default branch naming prefixes (for example the default prefix for the 'feature' branch type is `/feature`), as shown:



Development

This is generally the integration branch for feature work and is often the default branch (e.g. `master`) or a named branch such as `develop`. In a workflow using pull requests, this is usually the branch where new feature branches are targeted. In other cases, developers might commit directly to this branch.



Feature

`/feature`

Feature branches are used for specific feature work or improvements. They generally branch from, and merge back into, the development branch, by means of pull requests. See [Feature branch workflow](#).



Production

The production branch is used while deploying a release. It branches from, and merges back into, the development branch. In a Gitflow-based workflow it is used to prepare for a new production release.

Release

`/release`



Release branches are used for release task and long-term maintenance of software versions. Typically, they branch from, and fixes are merged back into, the development branch. Merging into an older release branch allows for [automatic merging](#) to newer release branches as well as the development branch.



Bugfix

/bugfix

Bugfix branches are typically used to fix release branches.





Hotfix

/hotfix

Hotfix branches are used to quickly fix the production branch without interrupting changes in the development branch. In a Gitflow-based workflow, changes are usually merged into the production and development branches.

Note that:

- Prefixes can't be empty.
- Prefixes can't be longer than 30 characters.
- Prefixes can't overlap; for example PROD and PRODUCT would be overlapping prefixes.
- For Microsoft SQL Server, prefixes can't use non-ASCII characters. See

 **STASH-3884** - Non-ASCII values used as branch model prefixes/branch names don't work in MSSQL ( [Open](#))

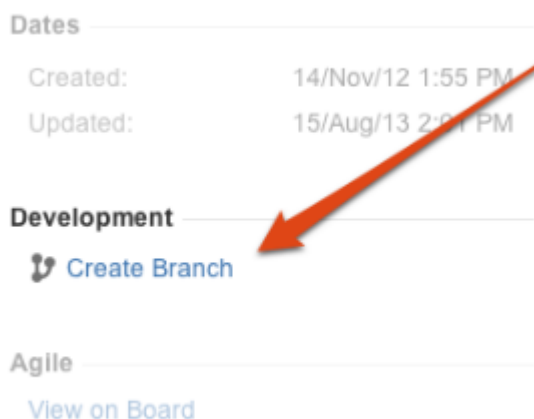
Creating branches

You can create a new branch when [in JIRA](#) (version 6.1 and above) or [in Stash](#). Either way, you can [override the settings](#) that Stash suggests for the repository, branch type, branching point and branch name.

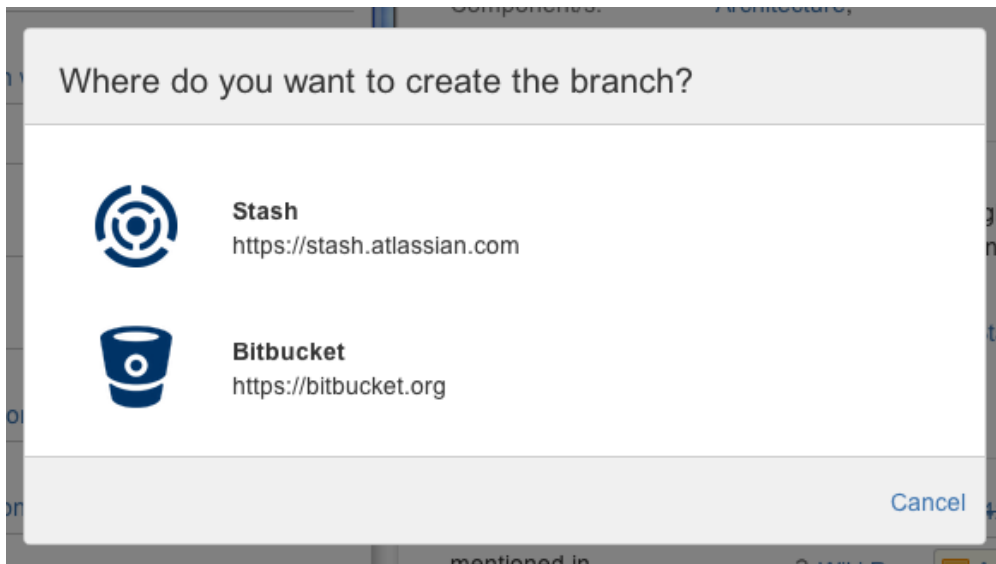
From a JIRA issue

 JIRA must be connected with Stash by an [application link](#) for this functionality to be available.

When viewing an issue in JIRA of JIRA Agile, click **Create Branch** (under 'Development' – you'll need the 'View Development Tools' project permission in JIRA to see this):



Choose the SCM, if more than one is available, where you want to create the branch:



Stash suggests the **Branch type** and **Branch name** based on the JIRA issue type and summary. Change the [settings](#) suggested by Stash, if necessary:


Create branch for STASHDEV-2493

Repository

Branch type [Learn about branch types](#)

Branch from

Branch name



From in Stash

In Stash, choose **Create branch** from the **Actions** menu when viewing the **Files**, **Commits** or **Branches** tabs, or when viewing a particular branch.

Stash suggests the **Branch type** and **Branch name** based on the JIRA issue type and summary. Notice that Stash displays the current [build status](#) beside the source branch picker. Change the [settings](#) suggested by Stash if necessary:

Create branch

Repository: Paul Watson / stash

Branch type: Bugfix

[Learn about branch types](#)

Branch from: STASH-2867-context-lines

Branch name: bugfix/STASH-2887-context-lines

Diagram: STASH-2867-context-lines (parent) → bugfix/STASH-2887-context-lines (child)

Create branch Cancel

Creating the branch

You can specify:

- the **Repository**
- the **Branch type**, if a [branching model](#) has been previously configured – choose **Custom** if you need an *ad hoc* branch type
- the branch point
- the **Branch name** – the prefix is based on the branch type you selected, and as defined by the [branching model](#). Note that the branch name should follow your team's convention for this.

Note that Stash suggests a **Branch type** based on the JIRA issue type, when a [branching model](#) is configured. The mapping is:

JIRA issue type	Stash branch type
Bug	Bugfix
Story	Feature
New Feature	Feature

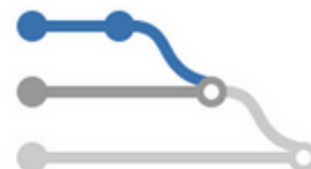
Once the new branch is created, Stash takes you to the file listing for that. You can now pull to your local repository and switch to the new branch.

Automating the branch workflow

Stash can automate some merges in the branch workflow, based on the branching model for the repository. This allows merges to be cascaded to newer branches of the same parent, subject to a few conditions, so reducing the need for manual maintenance of branches.

As a project administrator you can turn on automatic merging for a particular repository. Go to **Settings > Branching model** for the repository, and select **Enable automatic merging** (under 'Automatic merge').

If Stash cannot perform an automatic merge, perhaps because branch permissions prevent it, Stash creates a new pull request for that merge, and the automatic merge operation stops. This allows you to resolve the conflict locally before approving the new pull request, which may involve further cascading merges.



See [Automatic branch merging](#) for more information about the conditions for automatic merging, and how Stash determines the ordering of branches.

Managing all your branches

The branch listing page makes it easy to keep track of all the branches in your repository. It allows you to:

- See how many commits behind or ahead your branch is compared to a chosen 'base branch'.
- See the latest status for pull requests originating from branches.
- See the build status of branches at a glance.
- Easily find branches when you have a ton of them. Furthermore, if you're using the Stash [branch model](#), you can filter by branch type simply by searching for the prefix – for example, search for "feature/" to see all your feature branches.
- See the feature and bugfix branches that haven't yet been merged into a particular release (for example, "release/2.10") by changing the 'base branch'.

The **Behind/Ahead** column shows by how many commits a branch has diverged from the 'base branch' (for example, `master`). Use the branch selector (arrowed in the screenshot below) to change the base branch.

The **Pull requests** column shows the most relevant status from the pull requests against each branch – click an icon to see details. The status is:

- OPEN if there is at least one open pull request.
- MERGED if there are no open pull requests, and at least one pull request has been merged.
- DECLINED if there are no open or merged pull requests, and at least one pull request has been declined.

The **Builds** column shows the status of the latest build results published to Stash by an [integrated build server](#). The overall status is 'passed' if all the different builds (for example, unit tests, functional tests, deploy to staging) succeeded and 'failed' if at least one run failed for any of those. Click an icon to see details of the builds.

The **Actions** menus include tasks for working with the branches in the repository, such as:

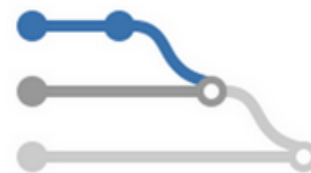
- Check out in SourceTree
- Create a pull request
- Edit permissions
- Delete branch

Choose **Keyboard shortcuts** from the Stash Help menu to see shortcuts to help you navigate quickly around the branch listing.

Files Commits Compare Branches Pull requests 14 Settings					
<div> <div> <div>↗ master ▾</div> <div>⋮</div> </div> <div> <input type="text" value="Search branches"/> </div> </div> <div>Learn more</div>					
Branch	Behind/Ahead	Updated	Pull requests	Builds	Actions
↗ bugfix/STASHDEV-2623-branch-perms-max-pattern-validation	3	6 mins ago	OPEN		⋮
↗ STASHDEV-5560-ssh-urls-access-keys	8 1	5 hours ago	OPEN	✓	⋮
↗ master <small>DEFAULT BRANCH</small>		10 hours ago	DECLINED	!	⋮
↗ bugfix/STASHDEV-5573-changeset-attribute	63 3	15 hours ago	OPEN	✓	⋮
↗ STASHDEV-5461-platform-upgrade-2.10	27 3	17 hours ago	OPEN	!	⋮
↗ release/2.9	46	Yesterday			⋮
↗ lower-case-slugs	60 1	4 days ago		!	⋮
↗ release/2.8	323	4 days ago	MERGED	✓	⋮
↗ release/2.7	359	4 days ago	MERGED	✓	⋮
↗ release/2.6	385	4 days ago	MERGED	✓	⋮

Automatic branch merging

Stash can automatically merge changes to newer release branches, thus reducing the need for manual maintenance of branches. To be able to do this, Stash has to be able to determine the [ordering of branches](#), and relies on [semantic versioning](#) of branch names – for example Stash will order these branch names like this: 1.0.0 < 2.0.0 < 2.1.0 < 2.1.1.



Note that:

- Automatic branch merging is subject to a few [conditions](#).
- Automatic merging is off by default for new and existing repositories.
- You must explicitly enable automatic merging for each repository.
- The commit message will indicate that the merge was automatic.
- Stash records full audit log entries for automatic merges.

As a project administrator, turn on automatic merging by going to **Settings** > **Branching model** for a repository, and selecting **Enable automatic merging** (under 'Automatic merge').

On this page:

- [Conditions for automatic merging](#)
 - [What happens if the automatic merge fails?](#)
- [Branch ordering algorithm](#)
- [Ordering examples](#)

Conditions for automatic merging

The following conditions must be satisfied for Stash to be able to automatically cascade changes:

- The Stash [branching model](#) must be configured for the repository.
- The 'release' branch type must be enabled for the repository.
- The merge must go via a pull request.
- The pull request must be made to a branch that is of the 'release' type.
- The target branch of the pull request must have branches that are [newer](#) than it.

What happens if the automatic merge fails?

The automatic merge can fail for reasons such as:

- Branch permissions prevent cascading changes to a particular branch.
- Stash detects a conflict that prevents the merge.
- There is already an open pull request with the same source and target that the automatic merge would close.

For the first two cases, Stash creates a new pull request for the failed merge, and the automatic merge operation stops. This allows you to resolve the conflict locally before approving the new merge, which may start a new series of cascading merges. Note that pull request that gets automatically opened when a merge fails won't trigger the continuation of the initial merge chain if resolved locally (which is the approach that we recommend).

Branch ordering algorithm

Stash is able to automatically merge changes to newer release branches, as long as Stash can determine the ordering of those branches. Ordering is based on [semantic versioning](#) in the naming pattern for branches.

Stash uses the following ordering algorithm to determine the branches in the merge chain:

- Branches are selected and ordered on the basis of the name of the branch that started the cascade (i.e. the target of the pull request for the merge).
- Branch names are split into tokens using any of these characters: underscore '_', hyphen '-', plus '+', or period '.'.
- Only branches *matching* the name of the pull request target are added into the merge path. Matching means that every token before the first numeric token must be equal to the corresponding tokens of the target branch's name.
- Branches are ordered by number, if a given token is numeric. When comparing a numeric token with

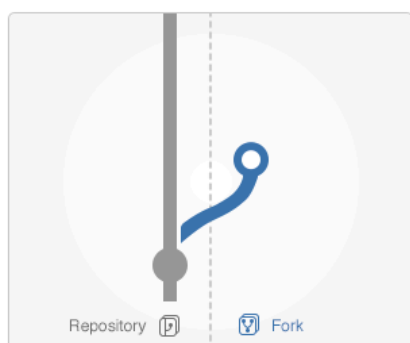
- an ASCII token, the numeric is ranked higher (i.e. is considered as being a newer version).
- If both tokens are numeric, a simple ASCII comparison is used.
- In the unlikely case of the above algorithm resulting in equality of 2 branch names, a simple string comparison is performed on the whole branch name.
- There is a limit of 30 merges.

Ordering examples

The table below provides examples of branch naming patterns that Stash is able, and not able, to order correctly:

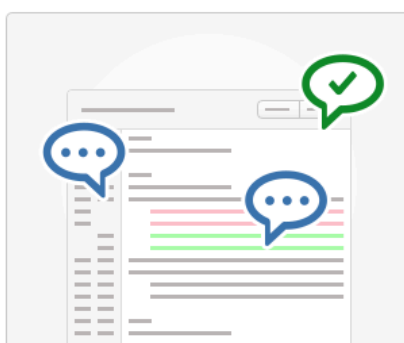
GOOD	<ul style="list-style-type: none"> release/1.0 release/1.1-rc1 release/1.1 release/1.2 release/2.0 	Stash tokenises on the '.' and the '-' of '1.1-rc1' and is able to order these branch names correctly.
GOOD	<ul style="list-style-type: none"> release/stash_1.1 release/stash_1.2 release/stash_2.0 	Stash tokenises on the '.' and the '_' and orders the numeric parts of these branch names correctly.
BAD	<ul style="list-style-type: none"> release/1.0 release/stash_1.1 	Stash tokenises on the '.' and the '_' but cannot recognise that 'stash_1.1' should follow '1.0'.

Using forks in Stash



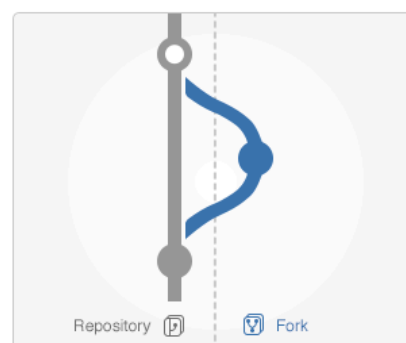
Fork

Develop features on a branch and create a pull request to get changes reviewed.



Discuss

Discuss and approve code changes related to the pull request.



Merge

Merge the branch with the click of a button.

Forks provide an alternative workflow to using branches, for where particular developers have restricted (read-only) access to a repository. See [Workflow strategies in Stash](#) for more information.

You can fork a repository into any other project in Stash for which you have admin access. You can also create [personal forks](#) and give other developers access to that using repository permissions.

Creating a fork

You can create a fork for any repository that you can see in Stash (that is, for which you have 'read' permission).

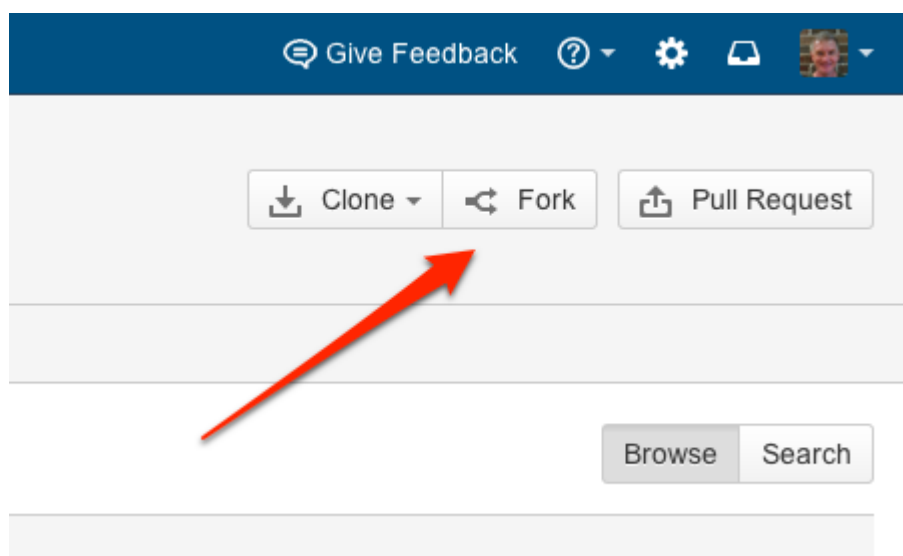
Simply click **Fork**, near the top right of the screen, when viewing a repository. You can choose the location for the newly forked repository. Note that when a repository is forked into another project it will get that project's permissions, which may be less restrictive.

On this page:

- [Creating a fork](#)
- [Issuing a pull request for a fork](#)
- [Merging a fork](#)
- [Synchronizing with upstream](#)
- [Disabling forking](#)
- [Pre-receive hooks and forks](#)

Related pages:

- [Workflow strategies in Stash](#)
- [Controlling access to code](#)
- [Creating personal repositories](#)



When creating the fork you can enable [fork syncing](#) to have Stash automatically keep your fork up-to-date with changes in the upstream repository.

Issuing a pull request for a fork

Pull requests for forks in Stash work just the way you'd expect. See [Using pull requests in Stash](#).

When creating the pull request, you can choose the fork and the branch that contains the source to be pulled, as well as the destination fork and branch.

Merging a fork

Once a pull request has been approved by reviewers, it can be merged as usual. See [Using pull requests in Stash](#).

Synchronizing with upstream

Once you fork a repository, your fork can be kept up-to-date with changes in the upstream repo either automatically by Stash or you can synchronize manually. You will still need to keep your remote working repository synced with your fork in Stash yourself. See [Keeping forks synchronized](#) for more details.

Disabling forking

Forking of repositories is available by default. However, you can turn off forking, on a per-repository basis, if this helps you to control your development process. You can do this on the **Repository details** tab of the repository settings.

Note that disabling forking on the parent repo doesn't delete any existing forks, and doesn't prevent those existing forks from being forked. Pull requests will still work from the existing forks. Furthermore, commits in the parent are viewable via the fork if the SHA1 hash is known to the user.

Pre-receive hooks and forks

Pre-receive hooks aren't copied with the fork and so are not run when code is merged in a pull-request. This means that custom hooks are unable to prevent certain changes from being merged by pull requests from forks. Instead, the hook would have to also implement a merge-check (see <https://developer.atlassian.com/stash/docs/latest/how-tos/repository-hooks.html>).

Keeping forks synchronized

Fork syncing helps you to keep your fork in Stash up-to-date with changes in the upstream repository. Stash can do this automatically for all branches and tags you haven't modified in the fork.

If you have modified branches or tags in the fork, Stash will offer syncing strategies. Stash will never update your branch or tag in your fork if this means that your changes would be lost.

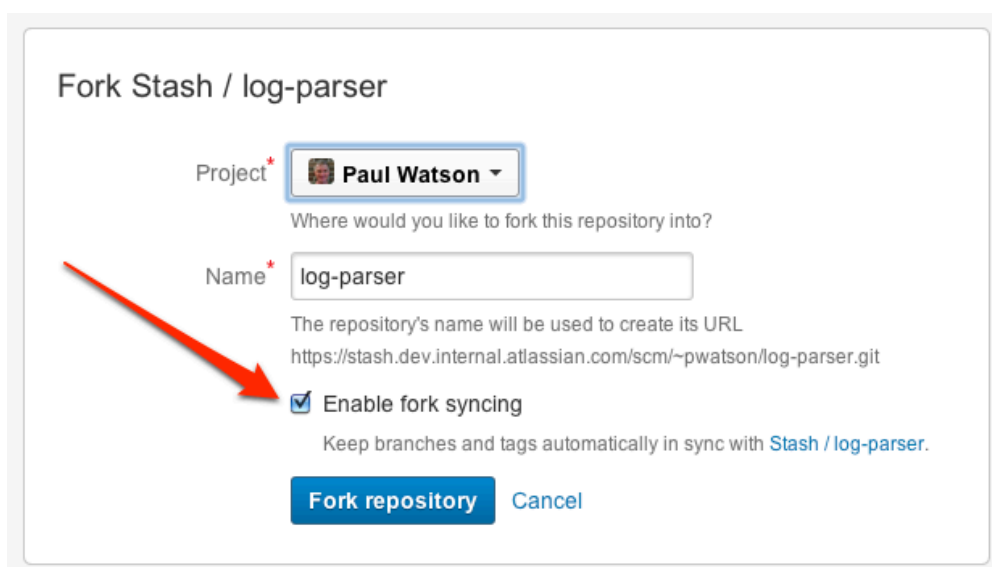
Note that syncing is about pulling recent upstream changes into your fork, whereas pull requests are about pushing your changes back to the upstream repository.

On this page:


- [Enabling automatic fork syncing](#)
- [What gets synced?](#)
- [Manual synchronization strategies](#)

Enabling automatic fork syncing

You can enable automatic fork syncing when you first fork the repository:



Fork Stash / log-parser

Project  Paul Watson

Where would you like to fork this repository into?

Name

The repository's name will be used to create its URL
<https://stash.dev.internal.atlassian.com/scm/~pwatson/log-parser.git>

☒ Enable fork syncing

Keep branches and tags automatically in sync with [Stash / log-parser](#).

Fork repository Cancel

You can also enable fork syncing at any later time by going to **Settings > Fork syncing** for the forked repository. Syncing is disabled by default.

What gets synced?

When performing automatic synchronization, Stash updates the fork as follows:

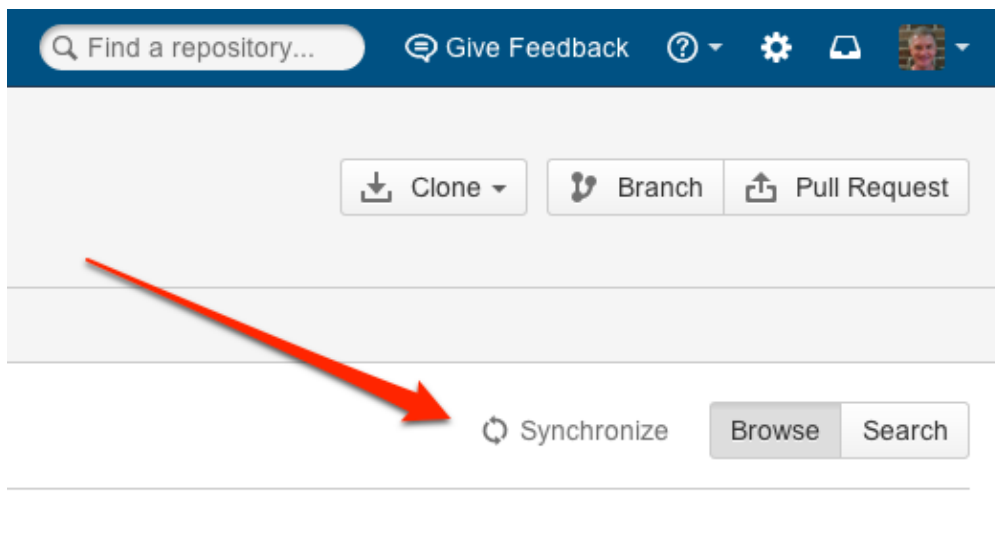
- for branches - Stash makes any fast-forward change, where there is no need to merge work and there is no risk of losing changes.
- for tags - Stash makes updates only if the current state is the same as what upstream pointed to. So, a new tag in upstream will create a new tag in the fork, unless you have a tag of the same name, when the update will fail.

Manual syncing

If upstream and your fork have diverged, so that each has changes that are not in the other, Stash will not perform a merge automatically. When you visit the branch within Stash, you have the option

to manually synchronize the branch

You can manually synchronize your branch at any time using **Synchronize** on either of the 'Files' or 'Commits' tabs for a repository:

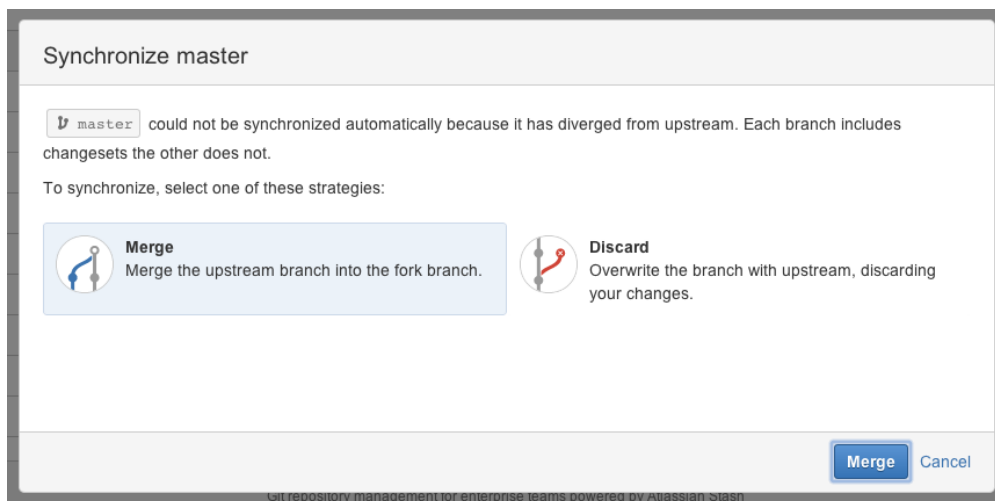


Manual synchronization strategies

When you initiate a manual synchronization, Stash ask you to choose one of the following synchronization strategies.

Merge strategy

Merge the upstream branch into the fork branch.



If Stash detects conflicts when trying to perform the merge it will offer hints on how to resolve those:

Merge conflicts

Upstream changes could not be merged in automatically due to conflicts in the following file:

- CLI feature.txt

How to perform a manual merge:

Step 1: Fetch changes from the upstream repository (saving the upstream branch as FETCH_HEAD).

```
git fetch https://stash.dev.internal.atlassian.com/scm/dox/toolbox.git master
```

Step 2: Checkout the fork branch and merge in the changes from the upstream branch. Resolve conflicts.

```
git checkout master
git merge FETCH_HEAD
```

Step 3: After the merge conflicts are resolved, stage the changes accordingly, commit the changes and push.

```
git commit
git push https://stash.dev.internal.atlassian.com/scm/~pwatson/toolbox.git HEAD
```

Close

Once the merge is complete, your branch will have incorporated all the commits on the branch in the parent repository, but your branch will still be ahead of the parent (it has your changes on it). This means automatic synchronization for this branch will not occur until your changes are pushed to the parent repository.


Discard strategy

Overwrite your changes in your fork with the upstream branch. Your changes will be lost.


Synchronize master

master could not be synchronized automatically because it has diverged from upstream. Each branch includes changesets the other does not.


To synchronize, select one of these strategies:



Merge
Merge the upstream branch into the fork branch.



Discard
Overwrite the branch with upstream, discarding your changes.

 Discarding means your changes will be lost

Discard Cancel

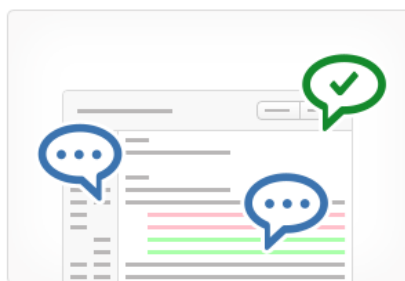
Git repository management for enterprise teams powered by Atlassian Stash

Using pull requests in Stash



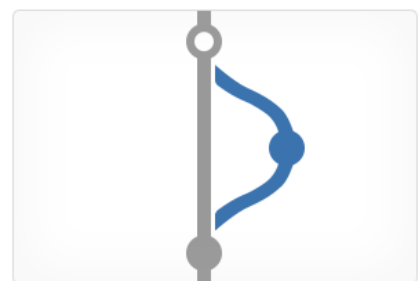
Branch

Develop features on a branch and create a pull request to get changes reviewed.



Discuss

Discuss and approve code changes related to the pull request.



Merge

Merge the branch with the click of a button.

Pull requests in Stash provide the team with a quick and easy way to review changes made on a branch, discuss those changes, and make further modifications before the branch is merged back to master or your main development branch.

On this page:

- [Creating a pull request](#)
- [Editing a pull request](#)
- [Discussing a pull request](#)
- [Merging a pull request](#)
- [Watching and notifications](#)

Related pages:

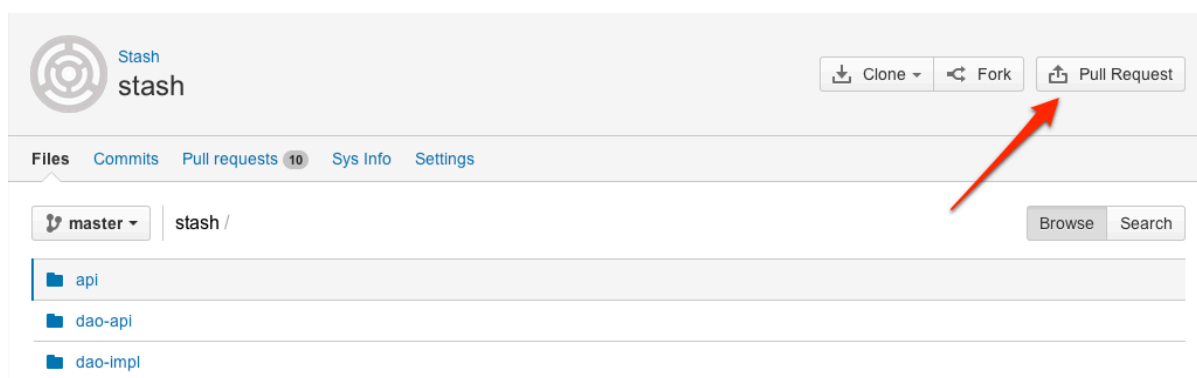
- [Using branch permissions](#)
- [Using project permissions](#)

Creating a pull request

When you are ready to start a discussion about your code changes, simply create a pull request.

To create a pull request:

1. Go to the repository in Stash.
2. Click **Pull Request** at the top of the page.



3. Choose the source and destination branches. The source branch is where you made your changes and the destination is the branch you want to merge to. The source and target branches may be located in different forks.

Create Pull Request

Source

Stash / stash ▾

Select branch ▾

Search for a branch

- STASHDEV-3230-README-plugin
- feature/account-mgmt/master
- BUILDENG-2607-backport-less-2.3
- STASHDEV-3430-navlinks-upgrade
- STASHDEV-3430-perf-regression-test
- STASHDEV-2981-restrict-granting-wrong-permissions
- STASHDEV-3648-delete-branch-on-merge-sends-repository-push-event
- release-2.3.0-RC4

Destination

Stash / stash ▾

master ▾

→

Adam Ahmed committed [a30331a](#) 20 mins ago

Automatic merge from 2.4 -> master # By Seb Ruiz # Via Adam Ahmed (1) and Seb Ruiz (1) * commit 'fd36431eb32c95d6d21a3ff334376b4c04eac766&#...

Pu

Reviewers

Reviewers can approve a pull request to let others know when it is good to merge.

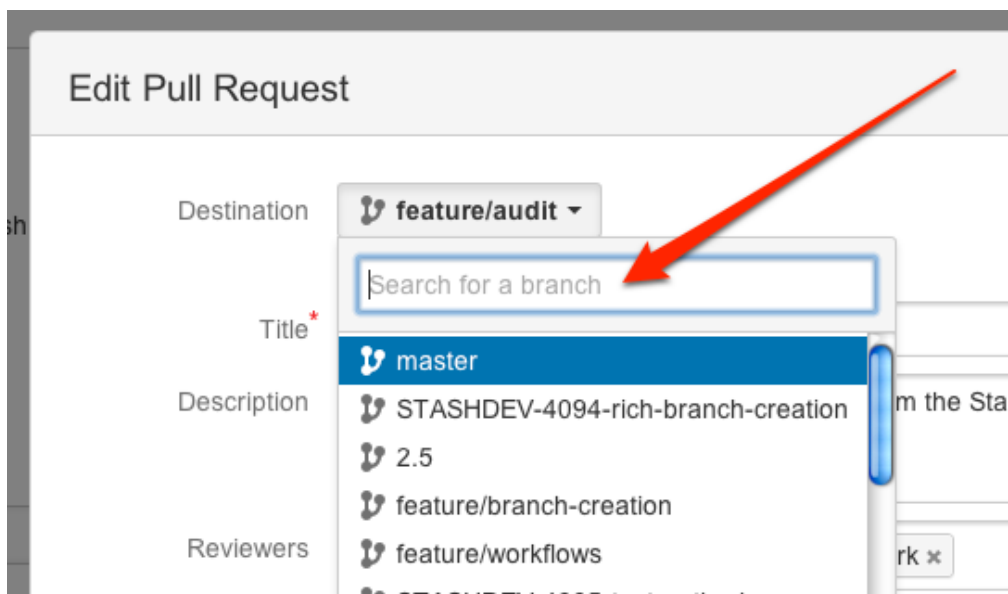
[Create pull request](#) [Cancel](#)

4. Enter a title and description that will help people understand what your pull request is about. You can use [mentions](#) (to notify another Stash user), and [markdown](#) (to add formatting) in your description.
5. Add reviewers – they will receive a notification by email. Other people who have [permissions](#) on the project can participate in the discussion if it interests them.
6. Click **Create pull request**.

You will receive email notifications when your reviewers and other participants comment on the pull request, or commit changes to it.

Editing a pull request

After creating the pull request, you can modify it later by clicking **Edit** on the pull request's page. You can edit details such as the **Title**, **Description** and the **Reviewers**. In particular, you can change the **Destination** branch for the pull request – you'll need Read [permission](#) on the branch you want to set.



Discussing a pull request

The most important thing about a pull request is the discussion that it generates. To help you contribute to the discussion, Stash organises all the information about the pull request into 3 tabs. From a project repository page, click **Pull requests**, then click on a particular pull request.

Overview

The **Overview** tab captures all of the team's activity on the pull request in one place, right from the initial creation, through to when it is finally merged (or declined), with all the comments, replies and commits that happen along the way.

You can add a comment on the **Overview** tab (above the activity), or reply to a previous comment. Use [mentions](#) to alert another Stash user to your comment, and use [markdown](#) to add formatting, for example headings or lists.

Diff

Diffs for Stash pull requests provide the following advantages:

- The diff highlights the changes that will result when the merge occurs, so you can see exactly what the effect of the merge will be.
- The diff tree on the left colour-codes files that have been added, changed or deleted, so you can quickly see the files you may need to review.
- As you'd expect, the diff for a file shows which lines of code have been added, deleted or modified.
- You can comment directly on a line of code right in the diff, by hovering over the line, clicking the icon and entering your comment. Your comment will also appear in the activity.
- Comments in the diff are threaded, to allow meaningful and contextual conversations about your code.

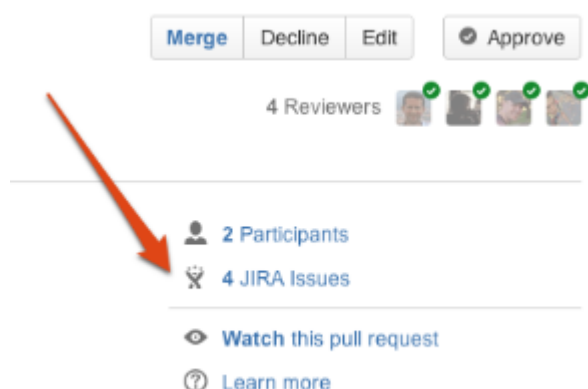
Commits

The **Commits** tab lists all the commits that will get merged.

Participants can commit new changes to the branch. Stash auto-updates the **Commits** tab of the pull request, so you can see exactly which commits will be merged. Stash is smart about comments, moving them along when lines are added or removed. If a line with a comment gets removed, you can still view the comment in the activity, but Stash marks the diff as *outdated* to let you know that this piece of code has been changed in recent commits.

JIRA issues integration

When Stash is [integrated with JIRA](#) you can see the related JIRA issues right in the pull request. You'll see either the JIRA issue key, when there is just one, or the number of related issues:



Click an issue key to see details of the issue, such as the description, status and assignee, without having to leave Stash:

STASHDEV-3099

[Back to issues](#)[View in JIRA](#)

Create BackupService and implement server-side support for backups

Details

Type Story

Priority Major

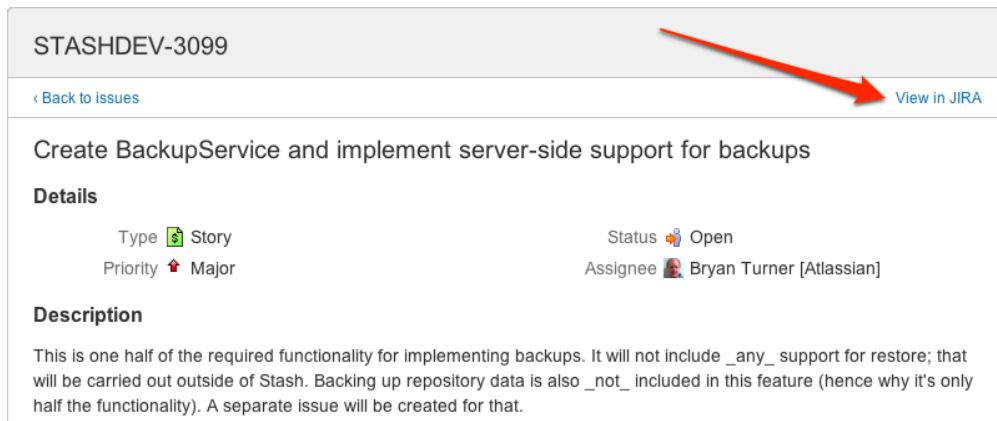
Status Open

Assignee Bryan Turner [Atlassian]

Description

This is one half of the required functionality for implementing backups. It will not include `_any_` support for restore; that will be carried out outside of Stash. Backing up repository data is also `_not_` included in this feature (hence why it's only half the functionality). A separate issue will be created for that.

Click **View in JIRA** to see the issue in JIRA. This allows reviewers to gain important insight into the task that is being worked on, by seeing the comments and attachments on the issue. This also gives access into JIRA, so you can easily keep issues updated.



STASHDEV-3099

[Back to issues](#) [View in JIRA](#)

Create BackupService and implement server-side support for backups

Details

Type Story Status Open
 Priority Major Assignee Bryan Turner [Atlassian]

Description

This is one half of the required functionality for implementing backups. It will not include `_any_` support for restore; that will be carried out outside of Stash. Backing up repository data is also `_not_` included in this feature (hence why it's only half the functionality). A separate issue will be created for that.

Merging a pull request

Once you are ready to merge a pull request, and when the reviewers have approved it, simply click **Merge** at the top right of the pull request view. You can only merge a pull request if you have Contributor [permission](#) on the project.

Note that the workflow for your team may include merge checks, which must also be met before you can merge. See [Checks for merging pull requests](#).



Files Commits Pull requests 10 Sys Info Settings

#1554 OPEN STASHDEV-39... → 2.4

STASHDEV-3933 pr activity paging 3 Reviewers

Overview Diff Commits

Details

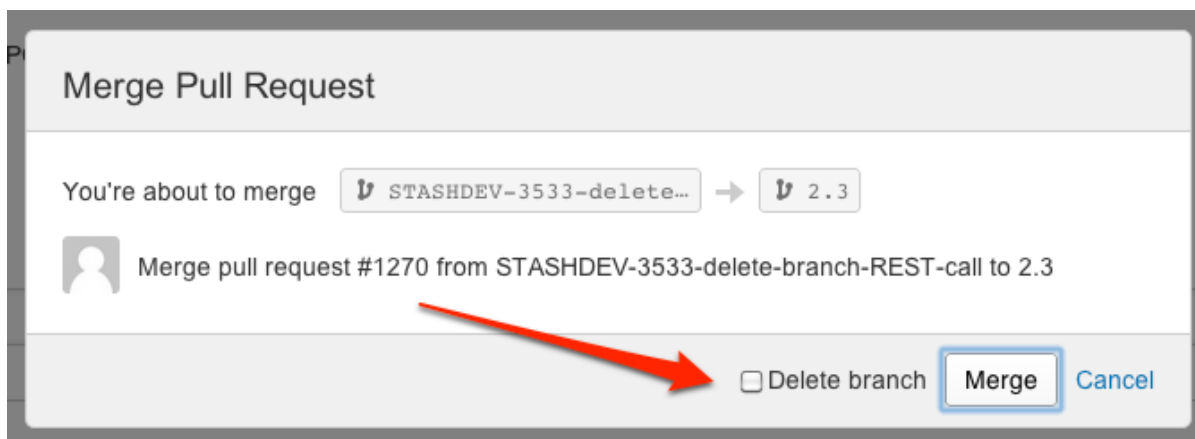
Xu-Heng Tjhin created a pull request 20 hours ago
 STASHDEV-3933 - Fix paging of activity when you deeplink to a comment

[Watch this pull request](#)
[Learn more](#)

Merge Decline Edit Approve

In the 'Merge Pull Request' dialog, you can check **Delete branch** if you no longer need that branch in the repository. Stash checks on a few things before allowing the deletion – the branch being merged will not be deleted if:

- The branch is the default repository branch.
- The user does not have permission to delete the branch.
- The branch is subject to an open pull request.



Merge Pull Request

You're about to merge STASHDEV-3533-delete... → 2.3

Merge pull request #1270 from STASHDEV-3533-delete-branch-REST-call to 2.3

☐ Delete branch **Merge** Cancel

Once accepted, the pull request is marked as merged on the **Pull requests** tab.

If Stash detects a conflict that prevents the merge, notifications are displayed on the **Overview** and **Diff** tabs of the pull request. Click **More information** to see instructions for how to resolve the conflict in your local

repository.

Watching and notifications

You automatically get added as a watcher of a pull request when you perform an action related to the pull request, such as adding a comment. You can manually add yourself as a watcher by clicking the **Watch** button on the pull request screen.

You can always stop watching a pull request by clicking the link in the email notification, or the **Unwatch** button on the pull request screen. If you stop watching a pull request you will not automatically be added as a watcher again if you subsequently perform an action that would otherwise have added you.

Stash sends email notifications to watchers when certain events occur. See [Notifications](#) for details.

Action	You are added as a watcher
You are added as a reviewer	✓
You comment on a pull request	✓
You reply to a comment	✓
You push to the source branch	✓
You approve the pull request	✓

Checks for merging pull requests

To help customise your workflow, you can set checks to control when a pull request can be merged. Pull requests cannot be merged if the required checks have not been met. These checks are set separately on each repository in a Stash project.

You'll need either project admin, admin or sys-admin [permissions](#) to set merge checks for pull requests.

So, to set merge checks for pull requests, go to a repository in a project and choose **Settings > Pull requests**. You can set the merge checks described below.

Require a number of completed builds

Select this option to stop pull requests from being merged if they have any unsuccessful builds. For a pull request, this checks builds that run against the latest commit on the source branch.

You must also specify a minimum number of builds - the pull request will not be able to be merged until at least this many builds have completed. Ideally, you should set this to the number of different builds that are configured to run against the branches in your repository.

See [Bamboo integration](#) for more information about integrating Stash with your build server.

1 Build ✓

STASHDEV-3300

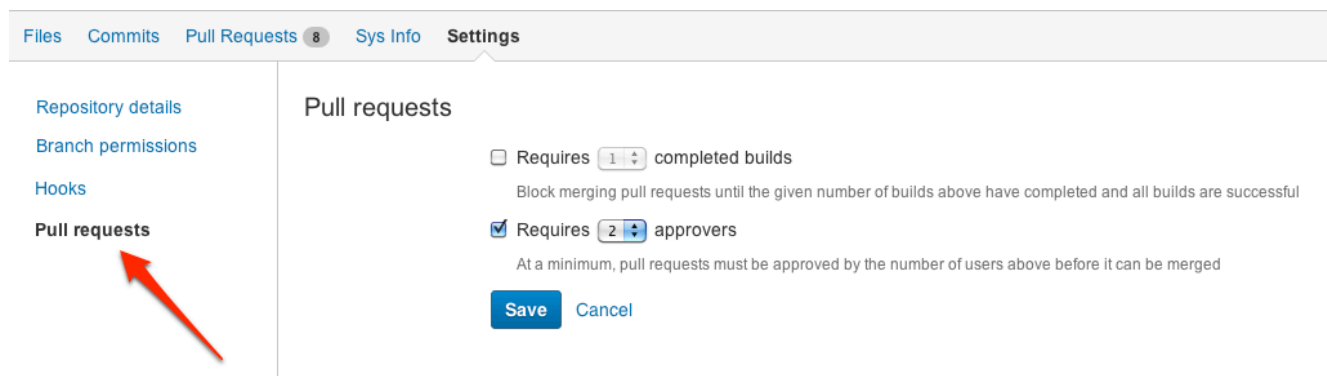
Watch this pull request

Learn more

The number of builds, and the latest result, for the head commit of a branch.

Require a minimum number of approvers

Select this option to block merging of a pull request until it has been approved by at least the selected number of participants.



Permanently authenticating with Git repositories

In addition to SSH, Stash supports HTTP or HTTPS for pushing and pulling from managed Git repositories. However, 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](#)

Related pages:

- [Getting started with Git and Stash](#)
- [Creating repositories](#)
- [Global permissions](#)
- [Git resources](#)

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.

Using SSH keys to secure Git operations

Stash provides a simple way for user accounts and other systems to connect to repositories for Git operations, using SSH keys.

- Add a [personal key](#) to a Stash user account to allow a developer to easily authenticate when performing read-write operations from his or her local machine.
- Add an [access key](#) to a Stash project or repository to allow other systems, such as build servers like Atlassian's [Bamboo](#), to authenticate for read-only operations (pull and clone), without the need to store user credentials.

On this page:

- [Personal SSH keys](#)
- [Project and repository SSH access keys](#)

Related pages:

- [Creating SSH keys](#)
- [Enabling SSH access to Git repositories in Stash](#)
- [Permanently authenticating with Git repositories](#)

Personal SSH keys

You can use SSH keys to establish a secure connection between your computer and Stash for when you are performing read-write Git operations from your local machine. Personal keys are attached to your Stash account – they are bound by that account's permissions and use the account's identity for any 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 use SSH keys to secure a connection with Stash the following must have already been done:

- your Stash administrator must have already [enabled SSH access](#) in Stash.
- you need an SSH key! See [Creating SSH keys](#). Alternatively, you can use an existing key, if it isn't already being used as a repository or project access key.
- you need to have added your personal SSH key to your Stash account – see the following section.

Once you have an SSH key associated with your Stash account, using it is easy! See [Use SSH keys to connect to Stash repositories](#) below.

Add an SSH key to your Stash account

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

Windows

```
clip < id_rsa.pub
```

On Mac OS X or Linux simply run the following in a terminal:

Mac OS X

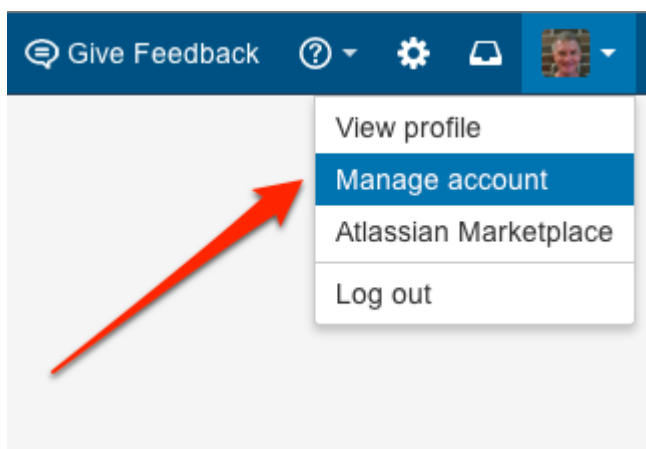
```
pbcopy < ~/.ssh/id_rsa.pub
```

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 account:



3. Click on **SSH keys** and then **Add key**.
4. Paste the key into the text box:

Add public key

Key*

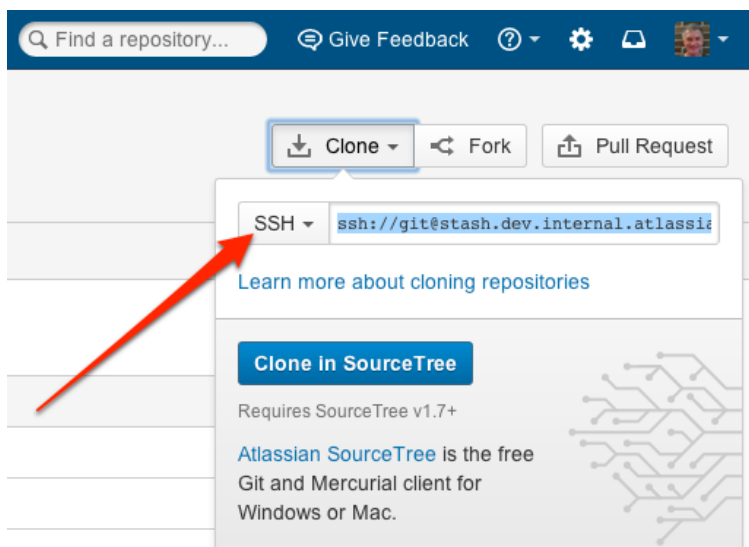
```
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQ
6ea78e5nBd8IKGo5FZ0H+3ZMlw>
UilW53Som5j8XOnhLdjrNP6EIgl/
eXsvCzoKJRmL7Q4o8EH2Hgs7Ml
y00kRV/bnm2HzlxtqhWSUHxbA63
Ywkj7P0zSRzoodq0NP8FqTb2aU
N7ZlhgbIn
/AV/j79zVEcplX1bn5DmyUU87Gui
yJzJRIryu7txrH9VSOntQ== pwats
```

5. Click **Add key**. You're done!

Use SSH keys to connect to Stash repositories

SSH access needs to have been set up, as [described above](#). Once this is done, you can use SSH keys as follows:

1. Go to **Projects**, click a project, and choose a repository from the list.
2. Click **Clone** at the top right to see the clone URLs for the repository.
3. Choose the clone URL you want to use. SSH is available if you have already added an SSH key to your account. If you haven't done that yet, see [Add an SSH key to your Stash account](#), above.



Project and repository SSH access keys

Stash administrators can set up SSH access keys to allow other systems to perform read-only Git operations on repositories managed in Stash. This allows systems, such as your build and deploy server, to authenticate with Stash to checkout and test source code, without having to store user credentials on another system, and without requiring it to link to a specific user account.

- [Project admins](#) can add and manage SSH access keys for a project. The keys apply to every repository in the project.
- [Repository admins](#) can add and manage SSH access keys for a particular repository.

[Repositories](#)
[Settings](#)

[Project details](#)
[Audit log](#)
[Permissions](#)
[Access keys](#)

Access keys

[Add key](#)

Access keys provide a simple way for other systems to access repositories using SSH without storing user credentials.

Label	Key
mobileapp@team-game.com	ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA04Mhb5KFkuHbZYvZFNvpi0yum8O1D...
twitterapp@ridinghigh.com	ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA6CYOy4zGRuXf3wHsgITnoVJLWYFq...

Stash's RSA fingerprint: 5c:0d:48:5c:02:22:16:76:77:f6:3b:ab:0f:95:5a:05

Stash supports DSA and RSA2 key types – RSA1 is not supported.

Before you can use SSH keys to secure a connection with Stash the following must have already been done:

- Your Stash administrator must have already [enabled SSH access](#), on Stash.
- You must have already created an SSL key. See [Creating SSH keys](#). Alternatively, you can use an existing key, if it isn't already being used for a personal account.

Using SSH keys to allow access to Stash repositories

To get the SSH key to work with your build, or other, system, you need to:

- Add the private key to that system. For Bamboo, see this page: [Sharing repository credentials](#).
- Add the public key to Stash as described here:

Add an SSH access key to either a Stash project or repository

You simply copy the public key, from the system that you want to have access, and paste it into Stash.

1. Copy the public key. One approach is to display the key on-screen using `cat`, and copy it from there:

```
cat < ~/.ssh/id_rsa.pub
```

2. Now, in Stash, go to the **Settings** tab for the project or repository.
3. Click **Access keys** and then **Add key**.
4. Paste the key into the text box and click **Add key**.

Permissions

- Access keys allow read-only access to a project or repository.

Stash license implications

- Access keys do not require an additional Stash user license.

Reusing access keys

- You can use the same SSH access key for multiple repositories or projects.
- Keys used for user accounts can't be re-used as a project or repository access key, and keys used as a project or repository access key can't be re-used for user accounts.

Creating SSH keys

This page describes how to create SSH keys.

SSH keys can be used to establish a secure connection with Stash for:

- when you are performing Git operations from your local machine
- when another system or process needs access to repositories in Stash (for example your build server)

The SSH key needs to be added to Stash, and your Stash administrator must have [enabled SSH access](#) to Git repositories, before you can make use of the key.

Supported key types are DSA and RSA2 – RSA1 is not supported.

On this page:

Related pages:

- [Using SSH keys to secure Git operations](#)
- [Enabling SSH access to Git repositories in Stash](#)
- [Permanently authenticating with Git repositories](#)

Creating an SSH key on Windows

1. Check for existing SSH keys

You should check for existing SSH keys on your local computer. *You can use an existing SSH key with Stash if you want, in which case you can go straight to [Using SSH keys to secure Git operations](#).*

Open a command prompt, and run:

```
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 [Using SSH keys to secure Git operations](#).

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.

In a command prompt on your local computer, run:

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

3. Generate a new SSH key

If you don't have an existing SSH key that you wish to use, generate one as follows:

1. Log in to your local computer as an administrator.
2. In a command prompt, run:

```
ssh-keygen -t rsa -C "your_email@example.com"
```

Associating the key with your email address helps you to identify the key later on.

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:

```
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):
```

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):
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:
e6:99:c3:3c:52:fb:9c:e4:3f:df:4d:b2:80:11:a5:1e ASUS@ASUS-PC
C:\Users\ASUS>
```

5. You're done! Now [add the new key to Stash](#).

Linux & Mac OS X

1. Check for existing SSH keys

You should check for existing SSH keys on your local computer. You can use an existing SSH key with Stash if you want, in which case you can go straight to [Using SSH keys to secure Git operations](#).

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 [Using SSH keys to secure Git operations](#).

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
```

3. Generate a new key

If you don't have an existing SSH key that you wish to use, generate one as follows:

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

```
ssh-keygen -t rsa -C "your_email@example.com"
```

Associating the key with your email address helps you to identify the key later on.

You'll see a response similar to this:

```
pwatsons-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:

```
pwatsons-Mac-Pro:~ pwatson$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/pwatson/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/pwatson/.ssh/id_rsa.
Your public key has been saved in /Users/pwatson/.ssh/id_rsa.pub.
The key fingerprint is:
47:68:04:f3:93:53:a3:af:bd:fc:86:60:30:47:cd:ea pwatson@pwatsons-Mac-Pro.local
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      o . oo      |
|      +. =o.      |
|      . O .       |
|      o. o=       |
|      =S o        |
|      E+         |
|      .....      |
|      .....      |
|      oo.         |
+-----+
pwatsons-Mac-Pro:~ pwatson$
```

4. You're done! Now [add the new key to Stash](#).

Notifications

An email server must be configured in Stash for email notifications to be sent. See [Setting up your mail server](#). Note that if the mail server fails, notifications will be dropped. See also [HipChat notifications](#).

Pull request notifications

Stash sends HTML email notifications to the watchers of a pull request when the following events occur:

Pull request event	Notification sent
A reviewer is added	✓
A comment is added	✓
A comment is edited	✓
A comment is replied to	✓
A commit is made to the source branch	✓
The pull request is approved	✓
The pull request is merged	✓
The pull request is declined	✓
The pull request is reopened	✓

Note that you don't receive notifications for events you initiate yourself.

Using 'mentions' to notify someone

From Stash 2.0 you can use 'mentions' to notify another Stash user about the pull request description or comment you are writing. Stash sends an email to that user.

To use mentions, simply start typing '@' and then the users display name, username or email address, and choose from the list that Stash offers. You can use quotes for unusual names, for example if it has spaces. Use Control-Shift-P or Command-Shift-P to preview the mention.

HipChat notifications

Stash can send an IM notification to a HipChat room whenever someone pushes to a repository. HipChat notifications are implemented as a [post receive hook](#) in Stash, and are available to all repos in Stash.

You can enable and disable HipChat notifications for a particular repo by going to **Settings > Hooks** for the repo:

The screenshot shows the Stash web interface for a repository's settings. The 'Hooks' tab is selected. Under 'Pre receive' (reject commits that don't match your policies), the 'Reject Force Push' hook is shown as disabled. Under 'Post receive' (perform actions after commits are processed), the 'HipChat Push Notification' hook is shown as enabled. The interface includes a sidebar with navigation links like 'Repository details', 'Hooks', 'Pull requests', and 'PERMISSIONS'.

You need to specify the API token and the name of the HipChat room:

HipChat Push Notification

Room Name*
Name of the HipChat room (or ID)

API Token*
Your HipChat API token

Specify the HipChat room to receive notifications about pushes and merges.
 You can get a list of available rooms from your HipChat client or through the [rooms page](#).
 You must specify an [API token](#) (only a user token is required).

Talk to your HipChat administrator to get the API token for your HipChat server.

Here's an example of what you might see, from our own HipChat room:

Stash	Tim Pettersen committed to 1 branch at Stash/stash On branch "STASHDEV-3418-hook-tx" - Merge branch 'STASHDEV-3418-hook-tx' of ssh://stash-dev.atlassian.com... (48327864b3f4) - STASHDEV-3418: use Operation instead of TransactionCallback (2367aab05eb0)	3:10 PM
Bamboo	Stash > Pull Request Build > STASHDEV-3418-hook-tx > #2 passed. 2893 passed. Changes by Charles O'Farrell	3:12 PM
Stash	Charles O'Farrell committed to 1 branch at Stash/stash On branch "2.2" - Merge pull request #1160 from feature/repo-hooks/wrm-reject-error-handling to ... (96368e3ae3c0) - NONE: Use html instead of empty/append (64bb2fa9817a) - NONE: Handle dynamic soy errors when showing dialog (84824e46981f)	3:13 PM
Stash	Charles O'Farrell committed to 1 branch at Stash/stash On branch "master" - Automatic merge from 2.2 -> master * commit '96368e3ae3c0c014a8590552... (9cc20cbd552d) - Merge pull request #1160 from feature/repo-hooks/wrm-reject-error-handling to ... (96368e3ae3c0) - NONE: Use html instead of empty/append (64bb2fa9817a) - NONE: Handle dynamic soy errors when showing dialog (84824e46981f)	3:13 PM

Markdown syntax guide

By default, Stash uses [Markdown](#) as its markup language. You can use markdown in the following places:

- any pull request's descriptions or comments, or
- in [README](#) files (if they have the .md file extension).

Use *Control-Shift-P* or *Command-Shift-P* to preview your markdown.

Markdown syntax

The table below contains examples of Markdown syntax. For a full list of all the Markdown syntax, consult the official documentation on John Gruber's [Daring Fireball](#) site.

Headings

```
# This is an H1
## This is an H2
...
##### This is an H6
```

Paragraphs

Each paragraph begins on a new line. Simply press <return> for a new line.

For example,
like this.

You'll need an empty line between a paragraph and any following markdown construct, such as an ordered or unordered list, for that to be rendered. Like this:

- * Item 1
- * Item 2

Character styles

```
*Italic characters*  
_Italic characters_  
**bold characters**  
__bold characters__
```

Unordered list

- * Item 1
- * Item 2
- * Item 3
 - * Item 3a
 - * Item 3b
 - * Item 3c

Ordered list

1. Step 1
2. Step 2
3. Step 3
 - a. Step 3a
 - b. Step 3b
 - c. Step 3c

List in list

1. Step 1
2. Step 2
3. Step 3
 - * Item 3a
 - * Item 3b
 - * Item 3c

Quotes or citations

Introducing my quote:

```
> Neque porro quisquam est qui  
> dolore ipsum quia dolor sit amet,  
> consectetur, adipisci velit...
```

Inline code characters

Use the backtick to refer to a `function()`.

There is a literal ``backtick (`)`` here.

Code blocks

Indent every line of the block by at least 4 spaces or 1 tab. Alternatively, you can also use 3 backtick quote marks before and after the block, like this:

```
```
```

Text to appear as a code block.

```
```
```

Within a code block, ampersands (&) and angle brackets (< and >) are automatically converted into HTML entities.

This is a normal paragraph:

 This is a code block.

 With multiple lines.

Links to external websites

This is [an example](http://www.slate.com/ "Title") inline link.

[This link](http://example.net/) has no title attribute.

Linking issue keys to JIRA

When you use JIRA issue keys (of the default format) in comments and pull request descriptions Stash automatically links them to the JIRA instance.

The default JIRA issue key format is two or more uppercase letters ([A-Z][A-Z]+), followed by a hyphen and the issue number, for example STASH-123.

Images

Inline image syntax looks like this:

```
![Alt text](/path/to/image.jpg)  
![Alt text](/path/to/image.png "Optional title attribute")  
![Alt text](url/to/image.jpg)
```

For example:

```
...
![Mockup for feature A](http://monosnap.com/image/b0cxxxxLGF.png)
...
```

Reference image links look like this:

```
![Alt text][id]
```

where 'id' is the name of a previously defined image reference, using syntax similar to link references:

```
[id]: url/to/image.jpg "Optional title attribute"
```

For example:

```
...
<!--Collected image definitions-->
[MockupA]: http://monosnap.com/image/b0cxxxxLGF.png "Screenshot of Feature A
mockup"
...
<!--Using an image reference-->
![Mockup for feature A][MockupA]
...
```

Inline HTML

An example, to add a table:

```
This is a regular paragraph.

<table>
  <tr>
    <td>Foo</td>
  </tr>
</table>

This is another regular paragraph.
```

Note that Markdown formatting syntax is not processed within block-level HTML tags. That is, you can't use Markdown-style **emphasis** inside an HTML block.

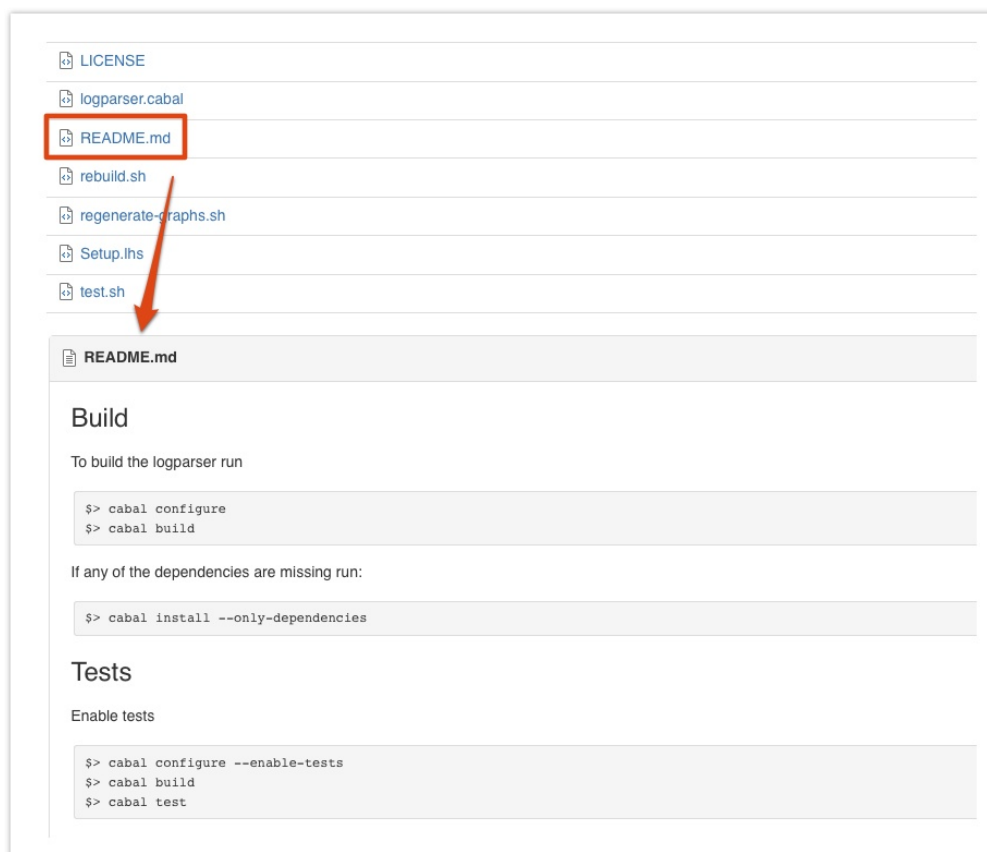
See <http://daringfireball.net/projects/markdown/syntax#html> for more details.

On this page:

- [Markdown syntax](#)
 - [Headings](#)
 - [Paragraphs](#)
 - [Character styles](#)
 - [Unordered list](#)
 - [Ordered list](#)
 - [List in list](#)
 - [Quotes or citations](#)
 - [Inline code characters](#)
 - [Code blocks](#)
 - [Links to external websites](#)
 - [Linking issue keys to JIRA](#)
 - [Images](#)
- [README files](#)

README files

From Stash 1.3, you can document a project right in the repository by creating .md or .txt files. If the ReadMe has the .md extension, any [Markdown](#) it contains gets rendered straight to the screen when viewed from the file list of the repository.



Requesting add-ons

The [Atlassian Marketplace](#) website offers hundreds of add-ons that the administrator of your Atlassian application can install to enhance and extend Atlassian products, including Stash. If the add-on request feature is enabled for your Stash instance, you can submit requests for add-ons from the Marketplace to your Stash administrator.

The 'Atlassian Marketplace for Stash' page provides an integrated view of the Atlassian Marketplace from within your Stash instance. The page offers the same features as the Marketplace website, such as searching and

category filtering, but tailors the browsing experience to Stash.

Atlassian Marketplace for Stash

Find and request powerful add-ons compatible with your Stash version on this streamlined Atlassian Marketplace. [Manage add-ons.](#)

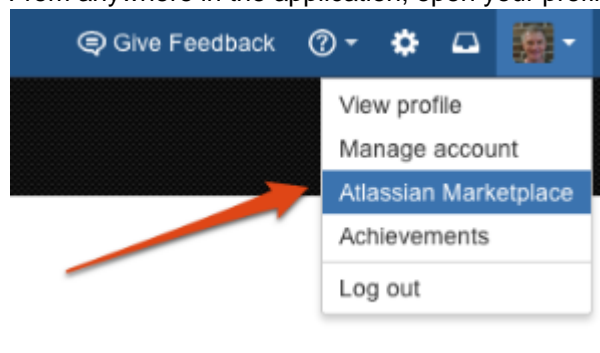
The screenshot displays the Atlassian Marketplace for Stash. At the top, there's a featured add-on 'Awesome Graphs for Stash Repos Visualisation' by StiltSoft, which is a 'Codegists Winner'. Below this is a search bar and filters for 'Staff Picked', 'All Categories', and 'Paid or Free'. The main list shows two add-ons: 'Stash Protect Unmerged Branch Hook' by Atlassian (260 Downloads, Free) and 'Notifyr' by StefanKohler (139 Downloads, Paid via Atlassian). Each add-on has a description and a 'Request' button.

This in-product view of the Marketplace gives day-to-day users of Atlassian applications, not just administrators, an easy way to discover add-ons that can help them get work done. When you find an add-on of interest, you can submit a request to your administrator for the add-on with just a few clicks.

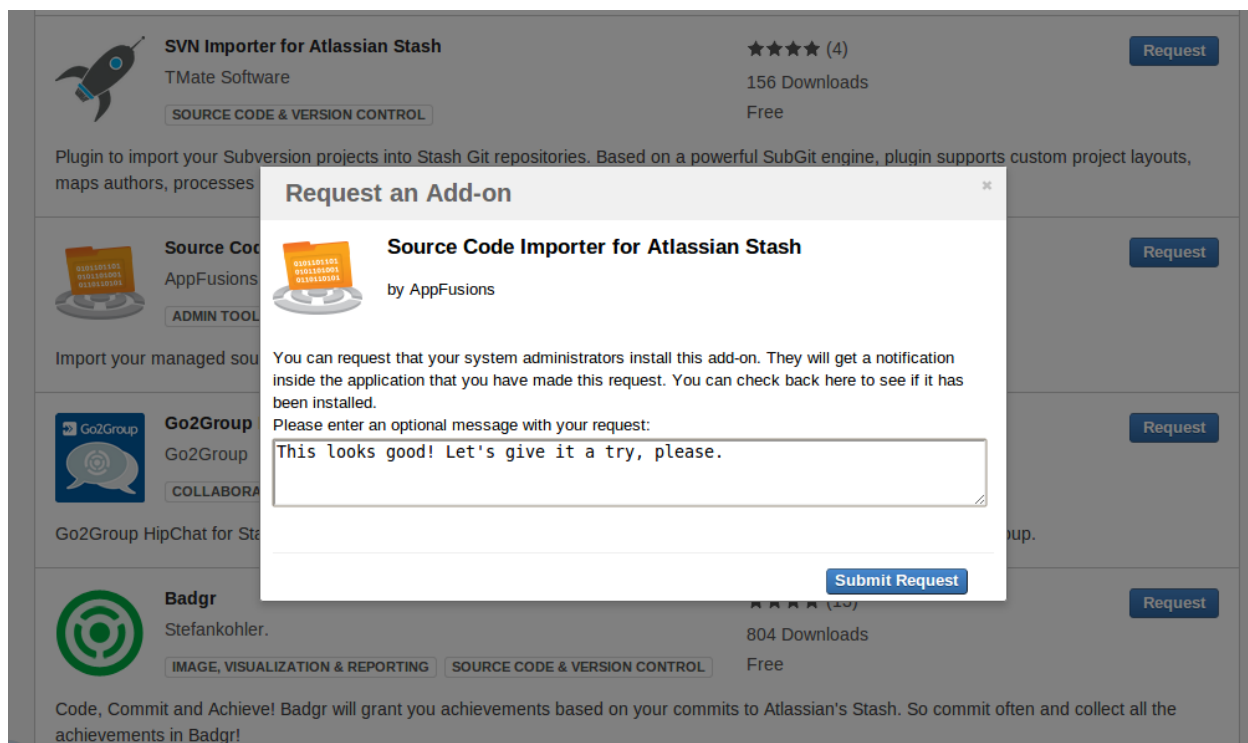
Submitting an add-on request

To browse for add-ons in the Atlassian Marketplace, follow these steps:

1. From anywhere in the application, open your profile menu and choose **Atlassian Marketplace**:



2. In the Atlassian Marketplace page, use the search box to find add-ons or use the category menus to browse or filter by add-ons by type, popularity, price or other criteria. You can see what your fellow users have requested by choosing the **Most Requested** filter.
3. When you find an add-on that interests you, click **Request** to generate a request for your administrator.
4. Optionally, type a personal message to your administrators in the text box. This message is visible to administrators in the details view for the add-on.



5. Click **Submit Request** when done.
6. Click **Close** to dismiss the 'Success!' message dialog box.

At this point, a notification appears in the interface your administrators use to administer add-ons. Also your request message will appear in the add-on details view, visible from the administrator's 'Find New Add-ons' page. From there, your administrator can purchase the add-on, try it out or dismiss requests.



Updating an add-on request





After submitting the request, you can update your message at any time. Click the **Update Request** button next to the listing in the Atlassian Marketplace page to modify the message to your administrator.

The administrator is not notified of the update. However, your updated message will appear as you have modified it in the details view for the add-on immediately.

Integrating Stash with Atlassian applications

When you integrate Stash with Atlassian applications you get the following benefits:

Application	Integration feature	Compatibility	
 	Create Git branches from within JIRA and JIRA Agile.	JIRA 6.1+	Stash 2.8+
	Transition JIRA issues from within Stash.	JIRA 5.0+	Stash 2.7+
	See the details of JIRA issues in Stash.	JIRA 5.0+	Stash 2.1+
	See the JIRA issues related to Stash commits and pull requests.	JIRA 5.0+	Stash 2.1+
	See all the files committed for the issue (on the JIRA Source tab). Click through to see a changed file , or the full commit, in Stash.	JIRA 5.0.4+	Plugin version bundled in JIRA

		JIRA 5.0–5.0.3	JIRA FishEye/Stash Plugin 5.0.4.1
		JIRA 4.4.x	JIRA FishEye/Stash Plugin 3.4.12
		JIRA 4.3.x	JIRA FishEye/Stash Plugin 3.1.8
	<p>When you have SourceTree installed, you can:</p> <ul style="list-style-type: none"> clone a Stash repository using SourceTree. check out a branch in SourceTree, when viewing files, commits or branches in a Stash repository. 	SourceTree 1.7+	Stash 2.7+
	<p>When Stash is integrated with Bamboo, you can:</p> <ul style="list-style-type: none"> see the latest build status for a commit when viewing Stash commits and pull requests. 	Bamboo 4.4+	Stash 2.1+
	<p>When Stash is integrated with HipChat, you:</p> <ul style="list-style-type: none"> get notifications to a HipChat room whenever someone pushes to a repository in Stash. 		
	<p>When Stash is integrated with Crowd, you can:</p> <ul style="list-style-type: none"> use Crowd for user and group management, and for authentication. 		

JIRA integration

When Stash is integrated with Atlassian [JIRA](#), you and your team get all these benefits:

- [Create Git branches](#) from within JIRA and JIRA Agile.
- [Transition JIRA issues](#) from within Stash.
- [Use JIRA issue keys](#) in Stash markdown.
- [See the details for JIRA issues](#) in Stash.
- [See the JIRA issues related to Stash commits and pull requests](#).
- [See all the files committed for the issue](#) (on the JIRA Source tab).
- From JIRA, click through to [see a changed file](#), or the [full commit](#), in Stash.

You can also use JIRA for delegated management of your Stash users. See [External user directories](#).

Your Stash administrator needs to set up [linking with JIRA](#) before you'll see these work.

Related pages:

- [Linking Stash with JIRA](#)
- [JIRA FishEye-Stash Plugin compatibility](#)
- [Connecting to JIRA for user management](#)

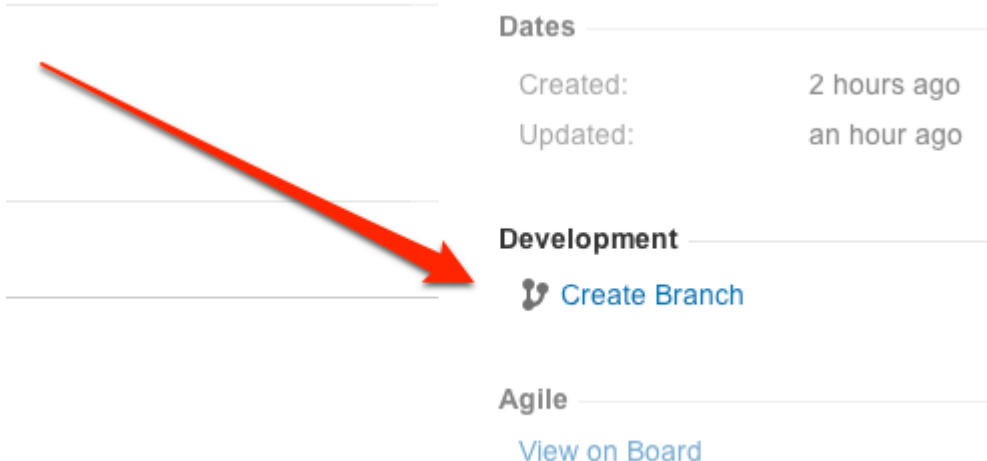
[Page feedback \(1 min\) »](#)

Create Git branches from within JIRA and JIRA Agile

STASH 2.8+

JIRA 6.1+

You can start creating a branch from a JIRA issue. This gives you a faster workflow from picking an issue to starting coding.



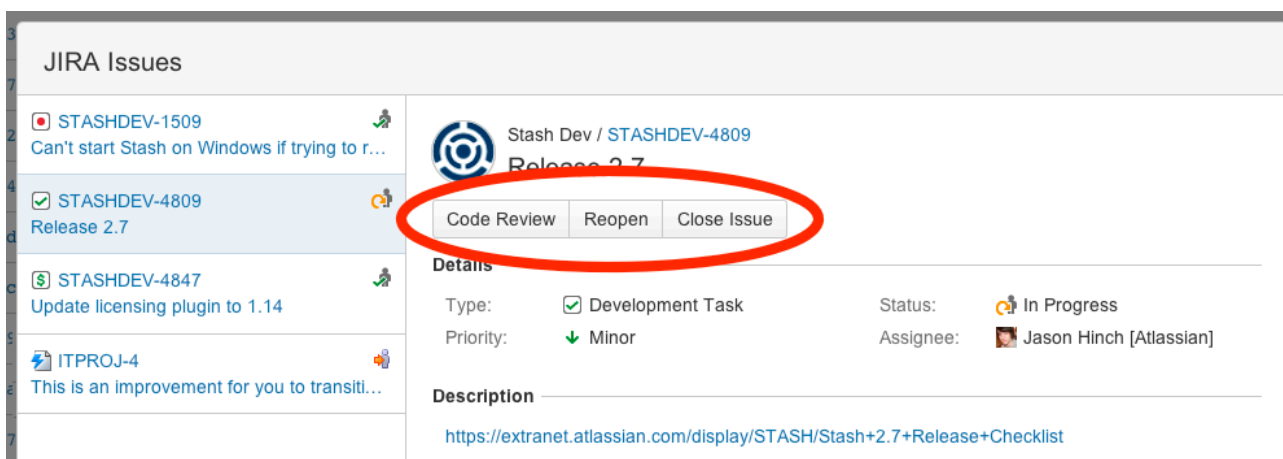
Stash will suggest the branch type and branch name, based on the JIRA issue type and summary – you can change these, of course.

Transition JIRA issues from within Stash

STASH 2.8+

JIRA 5.0+

You can easily [transition](#) a JIRA issue from within Stash. For example, when creating a pull request you may want to transition the issue into review. Click on a linked JIRA issue anywhere in Stash to see a dialog with the available workflow steps:



Click on a step and complete the fields as required. If there are custom fields that are unsupported by Stash, just click **Edit this field in JIRA** to transition the issue directly in JIRA.

See issues from multiple instances of JIRA

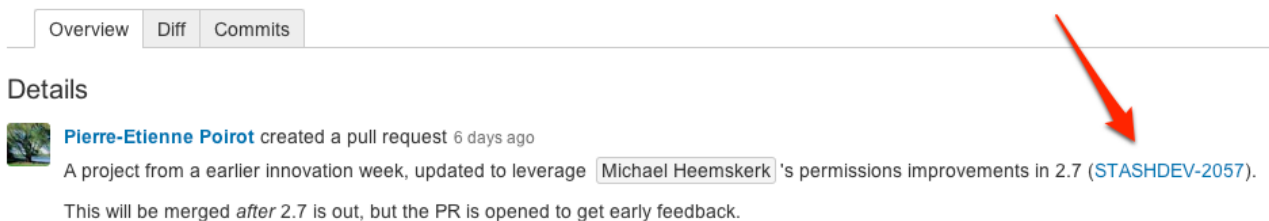
STASH 2.7+

Stash can link to more than one JIRA server at a time, so different teams can work with their own projects in different JIRA instances, or a single team can link to issues across multiple JIRA servers.

Use JIRA issue keys in markdown

STASH 2.7+

When you mention a JIRA issue key in Stash, for example in a pull request description or a comment, the key gets automatically linked:



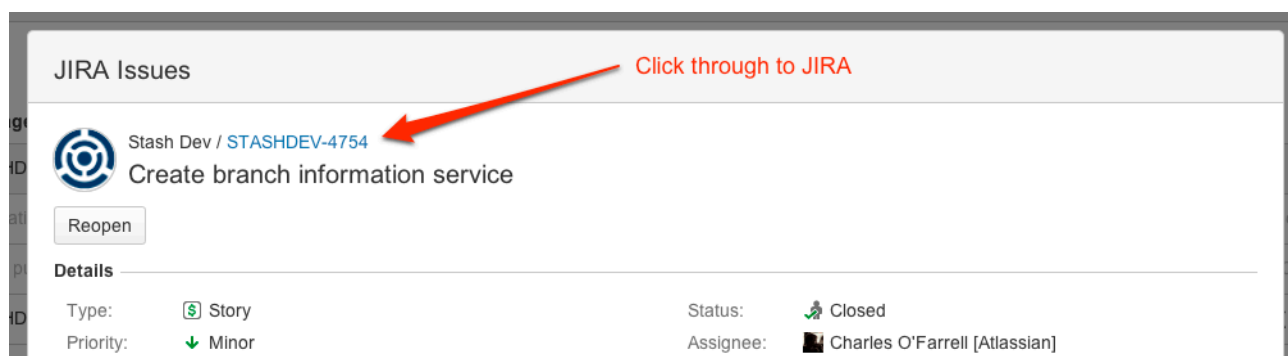
Click on the linked key to see [details](#) for the issue.

See the details for JIRA issues

STASH 2.1+

JIRA 5.0+

Click a linked issue key anywhere in Stash to see the details of that issue in a dialog. And you can just click the issue key at the top of the dialog to go straight to the issue in JIRA:



See the JIRA issues related to commits and pull requests

STASH 2.1+

JIRA 5.0+

Stash recognises JIRA issue keys in commit messages, and displays the keys as links on the Commits tabs for both the repository and [pull requests](#):

Message	Commit Date	Issues
STASHDEV-4785 - Upgrade stash inbox plugin to 1.3.2 with resources loaded asynchronously	37 mins ago	STASHDEV-4785
Automatic merge from 2.7 -> master * commit '664812515e74698423873e6a1fe6f84cd31db478': STASHD.	1 hour ago	STASHDEV-4836
Merge pull request #2161 in STASH/stash from ~MSTUDMAN/stash:STASHDEV-4836-disable-cancel-buttc.	1 hour ago	STASHDEV-4836
STASHDEV-4754 Switch to default-repository git zip for branch info The source build is restricted to public	2 days ago	STASHDEV-4754
STASHDEV-4816 Fixed missing space - breaking checks	2 days ago	STASHDEV-4816
STASHDEV-4836: jsHint fix	2 days ago	STASHDEV-4836

Click on the linked key to see [details](#) for the issue.

See details of Stash commits in JIRA

STASH 1.0+

JIRA 4.3+

Plugin version conditions
apply

In JIRA, you can see details for a commit made in Stash. And you can click through to go straight to the commit in Stash:

Activity

All Comments Work Log History Activity Commits **Source** Reviews Builds

Jason Hinch Yesterday 2:28 AM **Click through to Stash** → [View full commit](#)

Merge remote-tracking branch 'origin/master' into STASHDEV-1801-authentication-cleanup

Stash / stash **MERGE**

- MODIFIED** api/src/main/java/com/atlassian/stash/nav/NavBuilder.java
- MODIFIED** api/src/main/java/com/atlassian/stash/user/UserAdminService.java
- MOVED** distribution/licenses/src/main/resources/licenses/com.atlassian.utils-processutils--1.5.10.txt
- ADDED** etc/release/check_help_prefix.sh
- ADDED** etc/release/check_help_prefix_test.sh

... see more changes in Stash

Bamboo integration

When Stash is integrated with a build server such as Atlassian's [Bamboo](#), build results are published to Stash. You see the build results status for a commit when viewing any commit or pull request. This lets you easily check the build status of a branch when deciding whether to merge the changes.

Stash displays the overall status of the build results. The status is 'passed' if all the different builds (for example, unit tests, functional tests, deploy to staging) have succeeded, and 'failed' if at least one run failed for any of those.

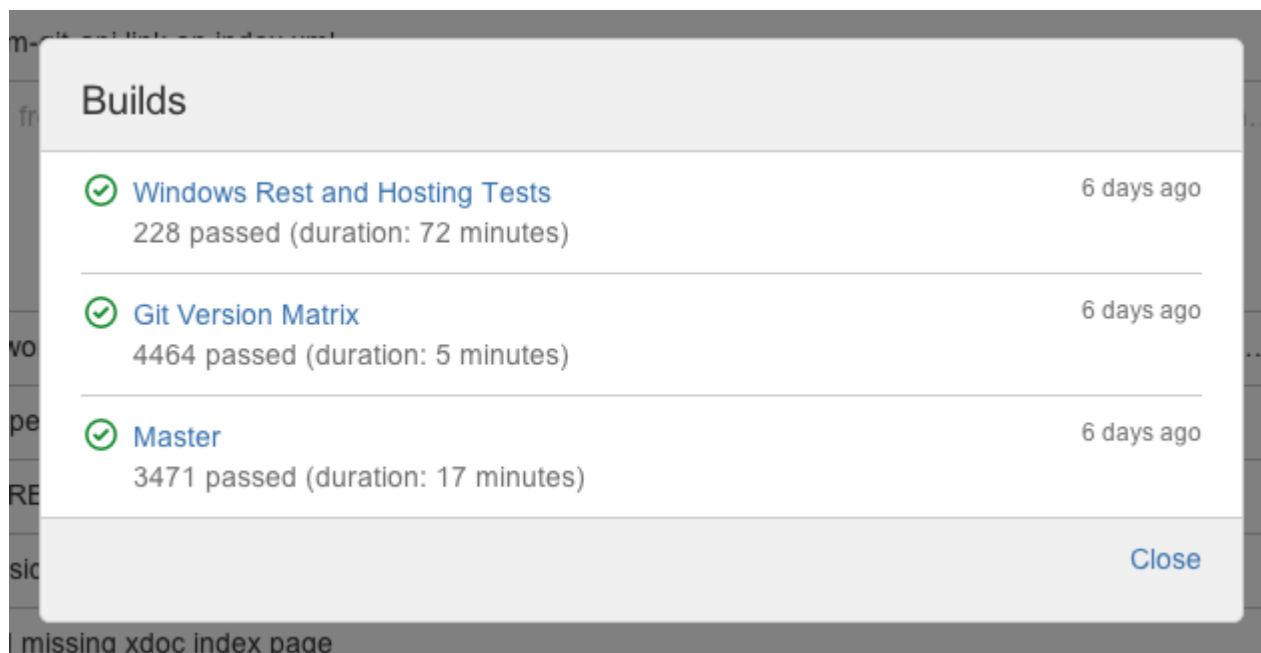
Related pages:

- [JIRA integration](#)
- [Using pull requests in Stash](#)

For example, when viewing the Commits tab for a Stash project, you will see icons that indicate the status of the latest build results. The red fail icon is displayed if there is at least one failed build run for the commit:

	Commit Date	Issues	Builds
f-file-modes to ...	22 Jan 2013	STASH-2798	✓
	21 Jan 2013	STASHDEV-3018	✓ 3 builds passed
	21 Jan 2013	STASHDEV-3021	
		STASHDEV-3020	
		STASHDEV-3018	!
		STASHDEV-3016	
		STASHDEV-3010	

Click a build status icon to see further details:



The information that you see in this dialog is specific to the build server that provides it, in this example Bamboo.

Integration of Bamboo build results in Stash is entirely configured on the Bamboo server. See [Integrating Bamboo with Stash](#).

You can also use the Stash Rest API to automatically publish build status from Bamboo, Jenkins or any other build tool to Stash. See the Stash developer documentation to do with [updating build status](#).

Administering Stash

This section describes some of the administrative actions that can be performed from the Stash Administration user interface (get there by clicking the 'cog' icon in the Stash header).





In this section:









- [Users and groups](#)
- [Global permissions](#)
- [JIRA integration](#)
 - [JIRA FishEye-Stash Plugin compatibility](#)
- [External user directories](#)
 - [Configuring an LDAP directory](#)
 - [Configuring delegated LDAP authentication](#)
 - [Connecting to Crowd](#)
 - [Connecting Stash 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:

-  [Setting up SSH port forwarding](#)
-  [Integrating Stash with Apache HTTP Server](#)
-  [Moving Stash to a different context path](#)
-  [Running Stash with a dedicated user](#)

-  [Data recovery and backups](#)
-  [Securing Stash with Tomcat using SSL](#)
-  [Stash debug logging](#)
-  [Changing the port that Stash listens on](#)
-  [Securing Stash with Apache using SSL](#)
-  [Stash config properties](#)
-  [Enabling SSH access to Git repositories in Stash](#)
-  [Scaling Stash](#)

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:

- Even after users have been added to the Stash user directory, they will not be able to log in to Stash until they have been given [global access permissions](#).
- Permissions can also be applied separately at the level of [projects](#), [repositories](#) and [branches](#).

On this page:

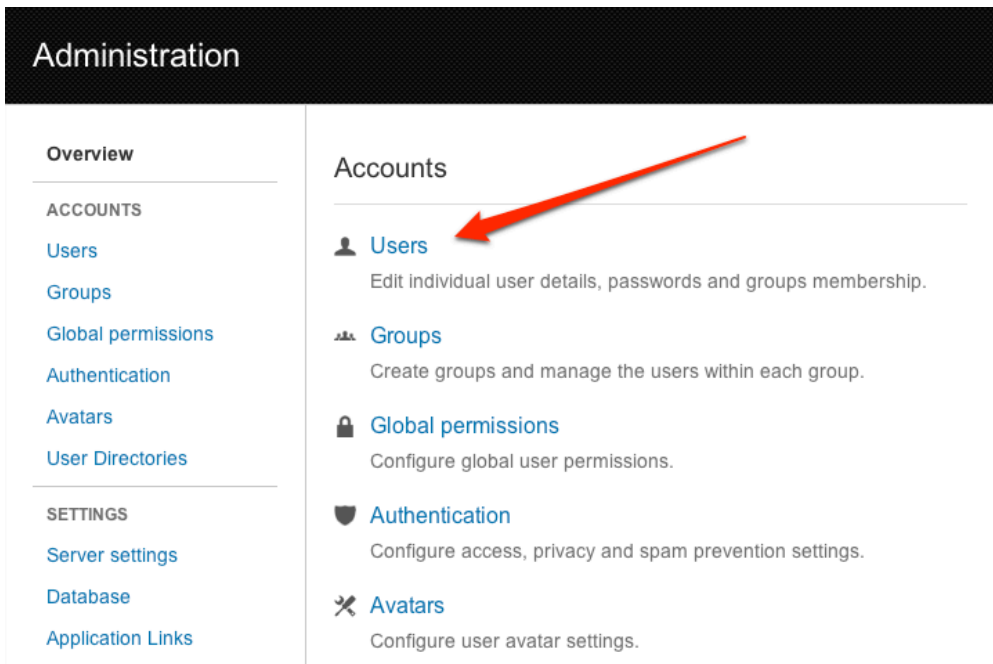
- [Creating a user](#)
- [Creating a group](#)
- [Adding users to groups](#)
 - [From the user account](#)
 - [From the group page](#)
- [Changing usernames](#)
- [Deleting users and groups](#)

Related pages:

- [Getting started with Stash](#)
- [External user directories](#)

Creating a user

In the administration area, click **Users** (under 'Accounts') and then **Create user** (on the 'Users' screen)



The 'Create User' form contains three required text fields: 'Username' (filled with 'jdoes'), 'Full name' (filled with 'John Does'), and 'Email address' (filled with 'john@does.com'). Below these fields is a checked checkbox labeled 'Email a link to the user to set their password'. At the bottom are two buttons: 'Create user' and 'Cancel'.

Create User

Username*

Full name*

Email address*

☒ Email a link to the user to set their password

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

[← Back to Users](#)

Edit

Change password

Rename

Delete



Change your avatar
with [Gravatar](#)

John Does

jdoes

john@does.com

STASH USER

[Change permissions](#)

Groups

SSH keys

Groups

Add to Group



stash-users

See [Global permissions](#) for more information.

Creating a group

In the administration area, click **Groups** (under 'Accounts') and then **Create group**. Enter the name for the new group, and click **Create group** (again):

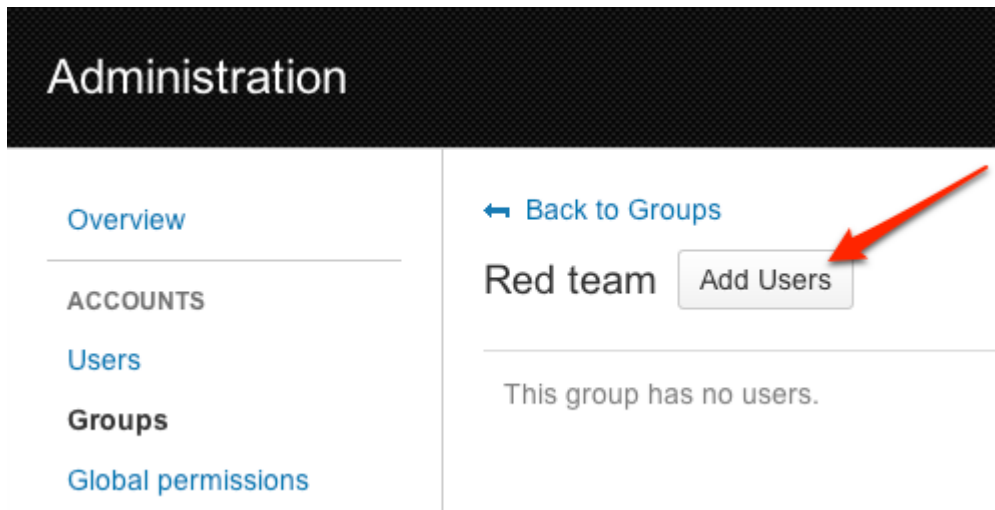
Create group

Group name*

Create group

Cancel

Now you can add users to your new group:



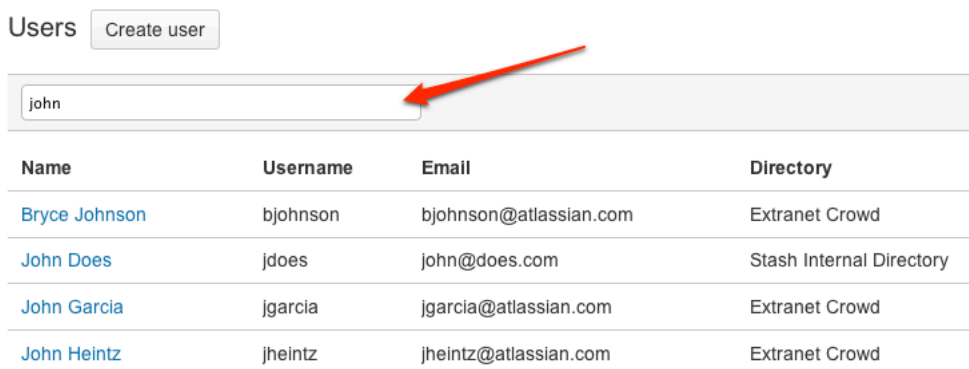
Adding users to groups

You can add users to groups in two ways:

- add a particular user to multiple groups, from the user account page in the admin area.
- add multiple users to a particular group, from the group's page.

From the user account

To add a user to a group from the user account page, go to **Users** in the Administration section, and use the filter to find the user:



On the account page for the user, click **Add to Group** to go to the list of available groups:

[← Back to Users](#)

Edit

Change password

Rename

Delete



Change your avatar
with [Gravatar](#)

John Does

jdoes

john@does.com

STASH USER

[Change permissions](#)

Groups

SSH keys

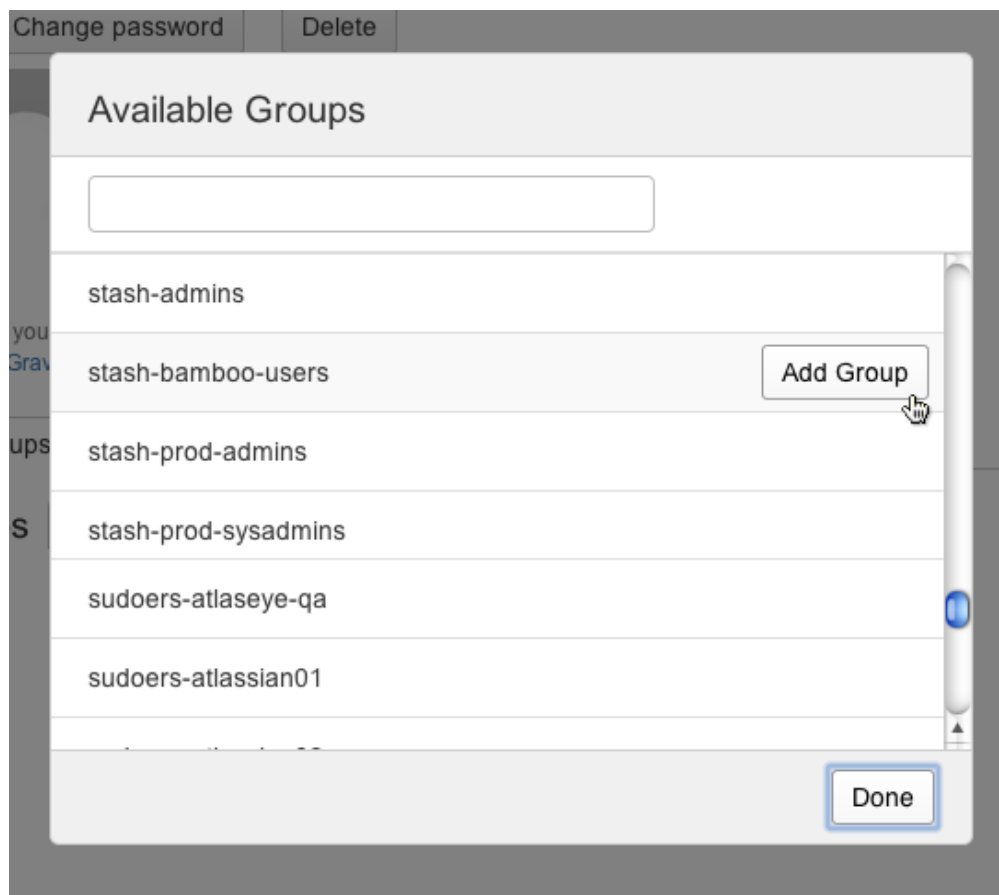
Groups

Add to Group



stash-users

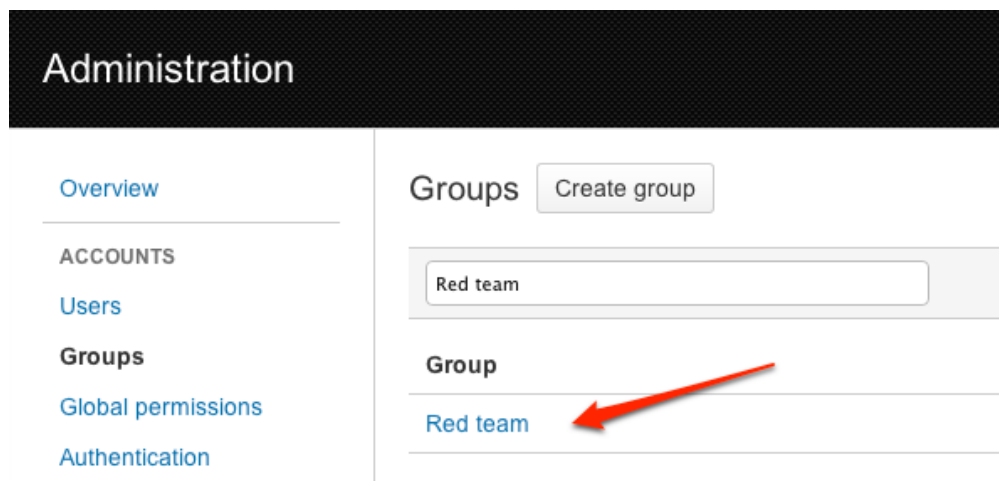
You can use the filter to find the group you want to add the user to. Hover to the right of the group name and click **Add Group** to make the user a member of the group.



Click **Done** when you have finished.

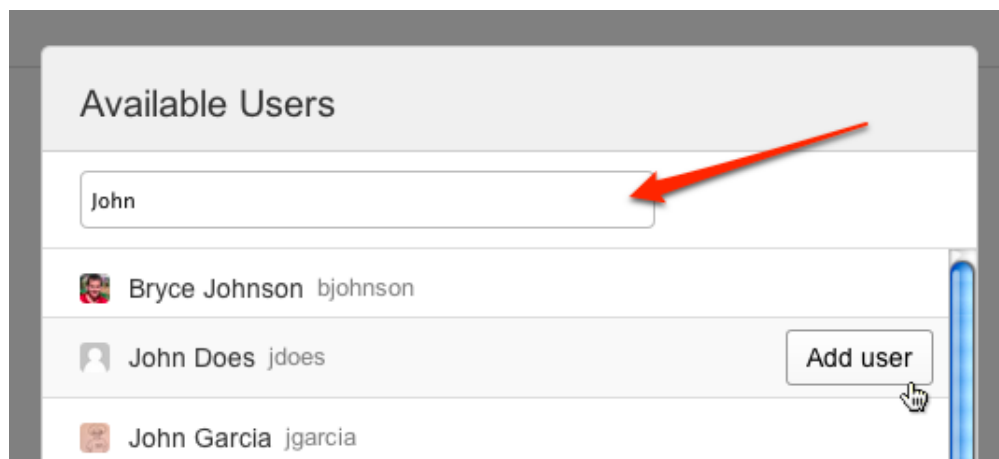
From the group page

To add a user to a group from the group's page, go to **Groups** (under "Accounts") in the administration area, 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 users list, use the filter to find a user. Hover to the right of the user's name and click **Add user** to make them a member of the group:



Click **Done** when you have finished.

Changing usernames

You can change the username for a user account that is hosted in Stash's internal user directory. Go to **Users** in the Administration section, and use the filter to find the user. On the account page for the user, click **Rename**.

Deleting users and groups

You can delete a user or group from Stash's internal user directory, or the external directory from which Stash sources users, such as an LDAP, Crowd or JIRA server.

When a user or group is deleted from such a directory, Stash checks to see if that user still exists in another directory:

- If the user or group **does** exist in another directory, Stash assumes the administrator intended to *migrate* the user or group between directories and we leave their data intact.
- If the user or group **does not** exist in another directory, Stash assumes the intent was to permanently delete them, and we delete the users permissions, SSH keys and 'rememberme' tokens.

Notes

- If an entire directory is deleted Stash *always* assumes it is a migration and does nothing to clean up after users and groups.
- Content which might be of historical interest (comments, pull requests, etc.) is not deleted when a user or group is. Only authentication, authorisation and data which serves no purpose to a user who can no longer log in is removed.
- In some situations, reordering the directories will change the directory that the current user comes from, if a user with the same username happens to exist in both. This behaviour can be used in some cases to create a copy of the existing configuration, move it to the top, then remove the old one. Note, however, that duplicate usernames are not a supported configuration.
- You can enable or disable a directory at any time. If you disable a directory, your configuration details will remain but Stash will not recognise the users and groups in that directory.

Limitations

- You cannot edit, disable or delete the directory that your own user account belongs to. This prevents administrators from locking themselves out of Stash, and applies to internal as well as external directories.
- You cannot remove the internal directory. This limitation aligns with the recommendation that you always keep an administrator or sysadmin account active in the Stash internal directory, so that you can troubleshoot problems with your user directories.
- You have to disable a directory before you can remove it. Removing a directory will remove the details from the database.

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.

See also this [information about deleting users and groups](#) in Stash.



Connecting Atlassian Stash to your external directory is not sufficient to allow your users to log in to Stash. You must explicitly grant them access to Stash in the [global permission screen](#).

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

There are several approaches to consider when using external user directories with Stash, described briefly below:

- [LDAP](#)
- [JIRA](#)
- [Crowd](#)

LDAP

You should consider connecting to an LDAP directory server if your users and groups are stored in an enterprise directory.

There are two common ways of using an external LDAP directory with Stash:

- For full user and group management, and for user authentication — see [Connecting Stash to an existing LDAP directory](#) for instructions.
- For delegated user authentication only, while using Stash's internal directory for user and group management — see [Delegating Stash authentication to an LDAP directory](#) for instructions.

Related pages:

- [Connecting Stash to an existing LDAP directory](#)
- [Delegating Stash authentication to an LDAP directory](#)
- [Connecting Stash to Crowd](#)
- [Connecting Stash to JIRA for user management](#)
- [Users and groups](#)

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

JIRA

You can delegate Stash user and group management, as well as user authentication, to an [Atlassian JIRA](#) instance. This is a good option if you already use JIRA in your organization. Note that Stash can only connect to a JIRA server running JIRA 4.3 or later.

You should consider using [Atlassian Crowd](#) for more complex configurations with a large number of users.

See [Connecting Stash to JIRA for user management](#) for configuration instructions.

Crowd

You can connect Stash to [Atlassian Crowd](#) for user and group management, as well as for user 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 Stash to Crowd](#) for configuration instructions.

Connecting Stash to an existing LDAP directory

You can connect Stash to an existing LDAP user directory, so that your existing users and groups stored in an enterprise directory, can be used in Stash.

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

See also this [information about deleting users and groups](#) in Stash.



Connecting Atlassian Stash to your external directory is not sufficient to allow your users to log in to Stash. You must explicitly grant them access to Stash in the [global permission screen](#).

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

On this page:

- [Server settings](#)
- [LDAP schema](#)
- [LDAP permission](#)
- [Advanced settings](#)
- [User schema settings](#)
- [Group schema settings](#)
- [Membership schema settings](#)

To connect Stash to an LDAP directory:

1. Log in as a user with 'Admin' permission.
2. In the Stash administration area, click **User Directories** (under 'Accounts').
3. Click **Add Directory** and select either **Microsoft Active Directory** or **LDAP** as the directory type.
4. Configure the directory settings, as described in the tables below.
5. Save the directory settings.
6. Define the directory order by clicking the 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

Setting	Description
Name	Enter a meaningful name to help you identify the LDAP directory server. Examples: <ul style="list-style-type: none"> • Example Company Staff Directory • Example Company Corporate LDAP
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 many of the options on the rest of screen. Examples: <ul style="list-style-type: none"> • Microsoft Active Directory • OpenDS • And more.
Hostname	The host name of your directory server. Examples: <ul style="list-style-type: none"> • ad.example.com • ldap.example.com • opens.example.com
Port	The port on which your directory server is listening. Examples: <ul style="list-style-type: none"> • 389 • 10389 • 636 (for example, for SSL)
Use SSL	Check this 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: <ul style="list-style-type: none"> • 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.

LDAP schema

Setting	Description
Base DN	The root distinguished name (DN) to use when running queries against the directory server. Examples: <ul style="list-style-type: none"> • 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 the domain1 and local for your specific configuration. Microsoft Server provides a tool called <code>ldp.exe</code> which is useful for finding out and configuring the 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: <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> • 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

Setting	Description
Enable Nested Groups	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.
Use Paged Results	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.
Follow Referrals	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.

Naive DN Matching	<p>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.</p> <p>This setting determines how your application will compare DN's to determine if they are equal.</p> <ul style="list-style-type: none"> • If this check box is selected, 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 DN's. • If this check box is not selected, the application will parse the DN and then check the parsed version.
Enable Incremental Synchronisation	<p>Enable incremental synchronisation if you only want changes since the last synchronisation to be queried when synchronising a directory.</p> <p> Please be aware that when using this option, the user account configured for synchronisation must have read access to:</p> <ul style="list-style-type: none"> • The <code>uSNChanged</code> 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). <p>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.</p> <p>This setting is only available if the directory type is set to "Microsoft Active Directory".</p>
Synchronisation Interval (minutes)	<p>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.</p>
Read Timeout (seconds)	<p>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.</p>
Search Timeout (seconds)	<p>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.</p>

Connection Timeout (seconds)	<p>This setting affects two actions. The default value is 0.</p> <ul style="list-style-type: none"> • 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

Setting	Description
User Object Class	<p>This is the name of the class used for the LDAP user object. Example:</p> <ul style="list-style-type: none"> • <code>user</code>
User Object Filter	<p>The filter to use when searching user objects. Example:</p> <ul style="list-style-type: none"> • <code>(&(objectCategory=Person)(sAMAccountName=*))</code> <p>More examples can be found here and here.</p>
User Name Attribute	<p>The attribute field to use when loading the username. Examples:</p> <ul style="list-style-type: none"> • <code>cn</code> • <code>sAMAccountName</code> <p>NB: In Active Directory, the 'sAMAccountName' is the 'User Logon Name (pre-Windows 2000)' field. The User Logon Name field is referenced by 'cn'.</p>
User Name RDN Attribute	<p>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:</p> <ul style="list-style-type: none"> • <code>cn</code>
User First Name Attribute	<p>The attribute field to use when loading the user's first name. Example:</p> <ul style="list-style-type: none"> • <code>givenName</code>
User Last Name Attribute	<p>The attribute field to use when loading the user's last name. Example:</p> <ul style="list-style-type: none"> • <code>sn</code>
User Display Name Attribute	<p>The attribute field to use when loading the user's full name. Example:</p> <ul style="list-style-type: none"> • <code>displayName</code>
User Email Attribute	<p>The attribute field to use when loading the user's email address. Example:</p> <ul style="list-style-type: none"> • <code>mail</code>

User Password Attribute	<p>The attribute field to use when loading a user's password. Example:</p> <ul style="list-style-type: none"> • <code>unicodePwd</code>
User Unique ID Attribute	<p>The attribute used as a unique immutable identifier for user objects. This is used to track username changes and is optional. If this attribute is not set (or is set to an invalid value), user renames will not be detected — they will be interpreted as a user deletion then a new user addition.</p> <p>This should normally point to a UUID value. Standards-compliant LDAP servers will implement this as 'entryUUID' according to RFC 4530. This setting exists because it is known under different names on some servers, e.g. 'objectGUID' in Microsoft Active Directory.</p>

Group schema settings

Setting	Description
Group Object Class	<p>This is the name of the class used for the LDAP group object. Examples:</p> <ul style="list-style-type: none"> • <code>groupOfUniqueNames</code> • <code>group</code>
Group Object Filter	<p>The filter to use when searching group objects. Example:</p> <ul style="list-style-type: none"> • <code>(&(objectClass=group)(cn=*))</code>
Group Name Attribute	<p>The attribute field to use when loading the group's name. Example:</p> <ul style="list-style-type: none"> • <code>cn</code>
Group Description Attribute	<p>The attribute field to use when loading the group's description. Example:</p> <ul style="list-style-type: none"> • <code>description</code>

Membership schema settings

Setting	Description
Group Members Attribute	<p>The attribute field to use when loading the group's members. Example:</p> <ul style="list-style-type: none"> • <code>member</code>
User Membership Attribute	<p>The attribute field to use when loading the user's groups. Example:</p> <ul style="list-style-type: none"> • <code>memberOf</code>

<p>Use the User Membership Attribute, when finding the user's group membership</p>	<p>Check this if your directory server supports the group membership attribute on the user. (By default, this is the 'memberOf' attribute.)</p> <ul style="list-style-type: none"> • If this checkbox is selected, 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 selected, your application will use the members attribute on the group ('member' by default) for the search. • If the Enable Nested Groups checkbox is selected, your application will ignore the Use the User Membership Attribute option and will use the members attribute on the group for the search.
<p>Use the User Membership Attribute, when finding the members of a group</p>	<p>Check this if your directory server supports the user membership attribute on the group. (By default, this is the 'member' attribute.)</p> <ul style="list-style-type: none"> • If this checkbox 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 search. • If this checkbox is not selected, your application will use the members attribute on the group ('member' by default) for the search.

Connecting Stash to JIRA for user management

You can connect Stash to an existing Atlassian JIRA instance, to delegate Stash user and group management, and authentication. Stash provides a "read-only" connection to JIRA for user management. This means that users and groups, fetched from JIRA, can only be modified or updated in that JIRA server, rather than in Stash.

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.

See also this [information about deleting users and groups](#) in Stash.

On this page:

- [Connecting Stash to JIRA](#)
- [Server settings](#)
- [JIRA server permissions](#)
- [Advanced settings](#)

Related pages:

- [External user directories](#)
- [Configuring JIRA integration in the Setup Wizard](#)
- [JIRA integration](#)

Connecting Stash 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 to Stash as a user with 'Admin' permission.
2. In the Stash administration area click **User Directories** (under 'Accounts').
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 arrows for 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 global permissions. 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 those groups as members of one or both of the above Stash groups.
 - Option 2: Log in to Stash using your JIRA account and go to the administration area. Click **Global permissions** (under 'Accounts'). Assign the appropriate permissions to the relevant JIRA groups. See [Global permissions](#).



Connecting Atlassian Stash to JIRA for user management is not sufficient, by itself, to allow your users to log in to Stash. You must also grant them access to Stash by using one of the above 2 options.

Server settings

Setting	Description
---------	-------------

Name	A meaningful name that will help you to identify this JIRA server amongst your list of directory servers. Examples: <ul style="list-style-type: none"> • JIRA Server • My Company JIRA
Server URL	The web address of your JIRA server. Examples: <ul style="list-style-type: none"> • <code>http://www.example.com:8080</code> • <code>http://jira.example.com</code>
Application Name	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 & Roles' section of the 'Administration' menu.
Application Password	The password used by your application when accessing the JIRA server that acts as user manager.

JIRA server permissions

Setting	Description
Read Only	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.

Advanced settings

Setting	Description
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.

Delegating Stash authentication to an LDAP directory

You can configure Stash to use an LDAP directory for delegated user authentication while still using the internal Stash directory for user and group management.

There is an option to automatically create a user account in Stash's internal directory when the user attempts to log in, as described in the [Copy users on login](#) section below.

See also this [information about deleting users and groups](#) in Stash.

To connect Stash to an LDAP directory for delegated authentication:

1. Log in to Stash as a user with 'Admin' permission.
2. Go to the Stash administration area and click **User Directories** (under 'Accounts').
3. Click **Add Directory** and select **Internal with LDAP Authentication** as the directory type.
4. Configure the directory settings, as described in the tables below.
5. Save the directory settings.
6. Define the directory order by clicking the arrows for 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.



Connecting Atlassian Stash to your external directory is not sufficient to allow your users to log in to Stash. You must explicitly grant them access to Stash in the [global permission screen](#).

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

On this page:

- [Server settings](#)
- [Copying users on login](#)
- [LDAP schema](#)
- [Advanced settings](#)
- [User schema settings](#)
- [Group schema settings](#)
- [Membership schema settings](#)

Server settings

Setting	Description
Name	<p>A descriptive name that will help you to identify the directory. Examples:</p> <ul style="list-style-type: none"> • Internal directory with LDAP Authentication • Corporate LDAP for Authentication Only
Directory Type	<p>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:</p> <ul style="list-style-type: none"> • Microsoft Active Directory • OpenDS • And more.
Hostname	<p>The host name of your directory server. Examples:</p> <ul style="list-style-type: none"> • ad.example.com • ldap.example.com • opens.example.com

Port	The port on which your directory server is listening. Examples: <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> • <code>cn=administrator,cn=users,dc=ad,dc=example,dc=com</code> • <code>cn=user,dc=domain,dc=name</code> • <code>user@domain.name</code>
Password	The password of the user specified above.

Copying users on login



The settings described in the table below relate to when a user attempts to authenticate with Stash. This authentication attempt can occur either:

- when using the Stash login screen.
- when issuing a Git clone or push command at the command line, for a repository managed by Stash.

Setting	Description
Copy User on Login	<p>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.</p> <p>If you select this check box the following additional fields will appear on the screen, which are described in more detail below:</p> <ul style="list-style-type: none"> • Default Group Memberships • Synchronise Group Memberships • User Schema Settings (described in a separate section below)

Default Group Memberships	<p>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.</p> <p>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.</p> <p>Examples:</p> <ul style="list-style-type: none"> • confluence-users • bamboo-users, jira-users, jira-developers
Synchronise Group Memberships	<p>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 the internal directory each time the user logs in.</p> <p>If you select this check box the following additional fields will appear on the screen, both described in more detail below:</p> <ul style="list-style-type: none"> • Group Schema Settings (described in a separate section below) • Membership Schema Settings (described in a separate section below)

Alternatively, you can move the delegated authentication directory to the top of the User Directories list and create the user manually (go to **Administration > Users > Create user**). Using this manual method you must currently create a junk password when creating users: there is an improvement request to address this. Please log in and vote for

 **STASH-3424** - Disable "Change password" field from admin and user page when delegated authentication is used ( **Open**)

LDAP schema

Setting	Description
---------	-------------

Base DN	<p>The root distinguished name (DN) to use when running queries against the directory server. Examples:</p> <ul style="list-style-type: none"> • <code>o=example,c=com</code> • <code>cn=users,dc=ad,dc=example,dc=com</code> • 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> for your specific configuration. Microsoft Server provides a tool called <code>ldp.exe</code> which is useful for finding out and configuring the the LDAP structure of your server.
User Name Attribute	<p>The attribute field to use when loading the username. Examples:</p> <ul style="list-style-type: none"> • <code>cn</code> • <code>sAMAccountName</code>

Advanced settings

Setting	Description
Enable Nested Groups	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.
Use Paged Results	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.
Follow Referrals	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.

User schema settings

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

Setting	Description
Additional User DN	<p>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:</p> <ul style="list-style-type: none"> • <code>ou=Users</code>

User Object Class	This is the name of the class used for the LDAP user object. Example: <ul style="list-style-type: none"> • user
User Object Filter	The filter to use when searching user objects. Example: <ul style="list-style-type: none"> • (&(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: <ul style="list-style-type: none"> • cn
User First Name Attribute	The attribute field to use when loading the user's first name. Example: <ul style="list-style-type: none"> • givenName
User Last Name Attribute	The attribute field to use when loading the user's last name. Example: <ul style="list-style-type: none"> • sn
User Display Name Attribute	The attribute field to use when loading the user's full name. Example: <ul style="list-style-type: none"> • displayName
User Email Attribute	The attribute field to use when loading the user's email address. Example: <ul style="list-style-type: none"> • mail

Group schema settings

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

Setting	Description
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: <ul style="list-style-type: none"> • ou=Groups
Group Object Class	This is the name of the class used for the LDAP group object. Examples: <ul style="list-style-type: none"> • groupOfUniqueNames • group
Group Object Filter	The filter to use when searching group objects. Example: <ul style="list-style-type: none"> • (objectCategory=Group)

Group Name Attribute	The attribute field to use when loading the group's name. Example: <ul style="list-style-type: none"> cn
Group Description Attribute	The attribute field to use when loading the group's description. Example: <ul style="list-style-type: none"> description

Membership schema settings

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

Setting	Description
Group Members Attribute	The attribute field to use when loading the group's members. Example: <ul style="list-style-type: none"> member
User Membership Attribute	The attribute field to use when loading the user's groups. Example: <ul style="list-style-type: none"> 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.) <ul style="list-style-type: none"> 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 Stash 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.

See also this [information about deleting users and groups](#) in Stash.



Connecting Atlassian Stash to your external directory is not sufficient to allow your users to log in to Stash. You must explicitly grant them access to Stash in the [global permission screen](#).

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

On this page:

- [Server settings](#)
- [Crowd permissions](#)
- [Advanced settings](#)
- [Single sign-on \(SSO\) with Crowd](#)

To connect Stash to Crowd:

1. Log in as a user with 'Admin' permission.
2. In the Stash administration area, click **User Directories** (under 'Accounts').
3. Click **Add Directory** and select **Atlassian Crowd**.
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.

Server settings

Setting	Description
Name	A meaningful name that will help you to identify this Crowd server amongst your list of directory servers. Examples: <ul style="list-style-type: none"> • Crowd Server • Example Company Crowd
Server URL	The web address of your Crowd console server. Examples: <ul style="list-style-type: none"> • http://www.example.com:8095/crowd/ • http://crowd.example.com
Application Name	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 .
Application Password	The password which the application will use when it authenticates against the Crowd framework as a client. This must be the same as the password you have registered in Crowd for this application. See the Crowd documentation on adding an application .

Crowd permissions

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

Setting	Description
Enable Nested Groups	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.

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.
------------------------------------	--

Single sign-on (SSO) with Crowd

Once the Crowd Directory has been set up, you can enable Crowd SSO integration by adding the following setting to `<STASH_HOME>/stash-config.properties`:

stash-config.properties

```
# Whether SSO support should be enabled or not. Regardless of this setting SSO
authentication
# will only be activated when a Crowd directory is configured in Stash that is
configured
# for SSO.
plugin.auth-crowd.sso.enabled=true
```

Please note that you will need to correctly set up the domains of the applications involved in SSO. See [Crowd SSO Domain examples](#)

In addition to this property, Crowd SSO integration can be tuned through the following properties, all set in `stash-config.properties`. The configuration properties and their default values are displayed below:

Property	Description	Default
<code>plugin.auth-crowd.sso.session.validationinterval</code>	The number of minutes to cache authentication validation in the session. If this value is set to 0, the SSO session will be validated with the Crowd server for every HTTP request.	3
<code>plugin.auth-crowd.sso.http.max.connections</code>	The maximum number of HTTP connections in the connection pool for communication with the Crowd server.	20
<code>plugin.auth-crowd.sso.http.proxy.host</code>	The name of the proxy server used to transport SOAP traffic to the Crowd server.	(none)
<code>plugin.auth-crowd.sso.http.proxy.port</code>	The connection port of the proxy server (must be specified if a proxy host is specified).	(none)
<code>plugin.auth-crowd.sso.http.proxy.username</code>	The username used to authenticate with the proxy server (if the proxy server requires authentication).	(none)
<code>plugin.auth-crowd.sso.http.proxy.password</code>	The password used to authenticate with the proxy server (if the proxy server requires authentication).	(none)

<code>plugin.auth-crowd.sso.http.timeout</code>	The HTTP connection timeout (milliseconds) used for communication with the Crowd server. A value of zero indicates that there is no connection timeout.	5000
<code>plugin.auth-crowd.sso.socket.timeout</code>	The socket timeout in milliseconds. You may wish to override the default value if the latency to the Crowd server is high.	20000

Global permissions

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


	Login / Browse	Create projects	Manage users / groups	Manage global permissions	Edit application settings	Edit server config
Stash User	✓	✗	✗	✗	✗	✗
Project Creator	✓	✓	✗	✗	✗	✗
Administrator	✓	✓	✓	✓	✓	✗
System Administrator	✓	✓	✓	✓	✓	✓



User accounts that have not been assigned "Stash User" permission or higher, either directly or through group membership, will not be able to log in to Stash. These users are considered unlicensed and do not count towards your Stash license limit.

A user's permission level is displayed on the user's page seen from the admin area.

[← Back to Users](#)

[Edit](#)
[Change password](#)
[Delete](#)



Kostya Marchenko
 marchenko_kostya
 kostya@atlassian.com
UNLICENSED [Change permissions](#)

Change your avatar with [Gravatar](#)

Note that you can also [apply access permissions to projects](#).

Related pages:

- [Getting started with Stash](#)
- [Users and groups](#)
- [Using project permissions](#)

To edit the account permissions for an existing Stash user or group:

1. Click the 'cog' menu in the header, to go to the admin area.
2. Click **Global permissions** (under 'Accounts').
3. Select, or clear, the permission checkboxes as required.
4. Click in the **Add Users** or **Add Groups** field to set permissions for additional users or groups.

You can remove all permissions for a user or group by clicking the X at the right-hand end of the row. This will remove the row.

Global Permissions

Individual Users

Name	System Admin...	Administrator	Project Creator	Stash User
Add Users			Stash User ▾	Add
Adam Ahmed	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Amber Buchan	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Aundray Cheam	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Administrator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Anton Mazkovoï	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Extranet Bamboo User	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Bryan Turner	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Groups

Name	System Admin...	Administrator	Project Creator	Stash User
Add Groups			Stash User ▾	Add
atlassian-dev	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
stash-admins	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
stash-users	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Setting up your mail server


Setting up Stash to use your SMTP mail server:

- allows Stash to send notifications about events to do with pull requests. See [Using pull requests in Stash](#). Note that if the mail server fails, notifications will be dropped.
- allows Stash to email a link to a newly created user, which the user can use to generate their own password.
- allows a user to reset his or her password if they forget it.

To configure a mail server for Stash, go to the administration area and click **Mail server** (under 'Settings'). See [Supported platforms](#) for the mail clients supported by Stash.

Fill in the form and click **Save**.

Hostname	The hostname of the mail server (for example "localhost" or "192.168.1.15").
-----------------	--

 **Anonymous User**
If you are looking to setup the outgoing mail server as an anonymous user, simply leave the username and password fields empty. In Chrome, these fields could be auto-populated, leading to an error. Try with another browser to help work around it.

Configuring the Mail Server to Use GMail

Linking Stash with JIRA

- [create Git branches](#) from a JIRA issue.
- see, in Stash, the JIRA issues related to particular commits.
- see the JIRA issues related to particular [pull requests](#).
- click through to see those issues in JIRA.
- [transition JIRA issues](#) directly from within Stash.
- see issues from multiple linked instances of JIRA.
- use JIRA issue keys in Markdown, and have them automatically link to JIRA.
- use JIRA for delegated user management. See [External user directories](#).

On this page:

- [Linking Stash with JIRA instances](#)
- [Restrictions for JIRA integration](#)
- [Known issues with JIRA integration](#)
- [Troubleshooting integration with JIRA](#)


Related pages:

- [JIRA integration](#)
- [JIRA FishEye-Stash Plugin compatibility](#)
- [Connecting to JIRA for user management](#)

Linking Stash with JIRA instances

You can integrate Stash with one or more instances of JIRA by means of 'application links'. You set up application links either:

- during the Stash install process, using the [Setup Wizard](#), or
- at any time after installation, as described below.

 **Note** that integrating Stash with JIRA may require an upgraded version of the FishEye/Stash plugin in JIRA. See [JIRA FishEye-Stash Plugin compatibility](#) for details about upgrading the JIRA FishEye/Stash plugin, and for download links to the upgraded plugin versions.

To link Stash to a JIRA server:

1. Click **Application Links** (under 'Settings') in the Stash admin area.
2. Enter the URL for the JIRA instance you want to link to and click **Create new link**.
3. Complete the application link wizard to connect Stash to your JIRA server. You *must* make use of the automatic link-back from JIRA to Stash to get full integration (you'll need JIRA system administrator global permission for that).

You're finished! 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 now appear in the issue's Source tab. Note that Stash only begins scanning commit messages for JIRA issue keys on the first push after you created the application link to JIRA – the scan may take a short time.

More detailed information about application links can be found on [Configuring Application Links](#).



Restrictions for JIRA integration

- The display of [details for JIRA issues](#), for example when viewing a pull request, relies on the JIRA 5.0 REST API. Issue details are not displayed when Stash is integrated with JIRA versions earlier than 5.0.
- Transitioning JIRA issues requires Trusted Apps or OAuth AppLinks. If only a Basic Auth applink is set up, users will be able to view issue details, but will not be able to transition issues.
- JIRA permissions are respected, so a user who is not permitted to transition an issue in JIRA will not see the transition buttons in Stash.

Known issues with JIRA integration

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

- There is no checking for project or issue-key validity; Stash may link to issues that do not actually exist:

 **STASH-2470** - JIRA Integration: Check for issue validity before linking issues ( [Open](#))

We apologise for the inconvenience. Please watch the issue to keep track of our progress.

Troubleshooting integration with JIRA

There are a few scenarios where the integration of Stash with JIRA can produce an error:

The application link is misconfigured

This can result if authentication for the application link has not been set up. See [Troubleshooting JIRA Integration](#).

You don't have permission to access the JIRA project

If you don't have permission to access the JIRA project then Stash is unable to display issues.

The JIRA server is of an unsupported version

Stash can integrate with JIRA 4.3.x, or later. Some features require higher versions of JIRA to function properly. See [Integrating Stash with Atlassian applications](#) for details.

The JIRA issue key is invalid

Stash doesn't check for invalid issue keys, such as UTF-8. An error will result if Stash tries to connect to an issue that doesn't exist.

The JIRA issue keys are of a custom format

Stash assumes that JIRA issue keys are of the default format (that is, two or more uppercase letters ([A-Z][A-Z]+), followed by a hyphen and the issue number, for example STASH-123). By default, Stash will not recognise custom JIRA issue key formats. See [Using custom JIRA issue keys with Stash](#) for details.

The Application Link is created with OAuth only without the option to create a link using Trusted Applications

Stash allows a user with global permissions of "Administrator" to create an OAuth only application link. Log in with a user having "System Administrator" privileges to create an application link using Trusted Applications.

JIRA FishEye-Stash Plugin compatibility

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

The information on this page applies to Stash versions 2.1 and later.

On this page:

- [Supported JIRA versions](#)
- [Plugin upgrade guide](#)

Related pages:

- [JIRA integration](#)
- [Linking Stash with JIRA](#)
- [Stash releases](#)

Supported JIRA versions

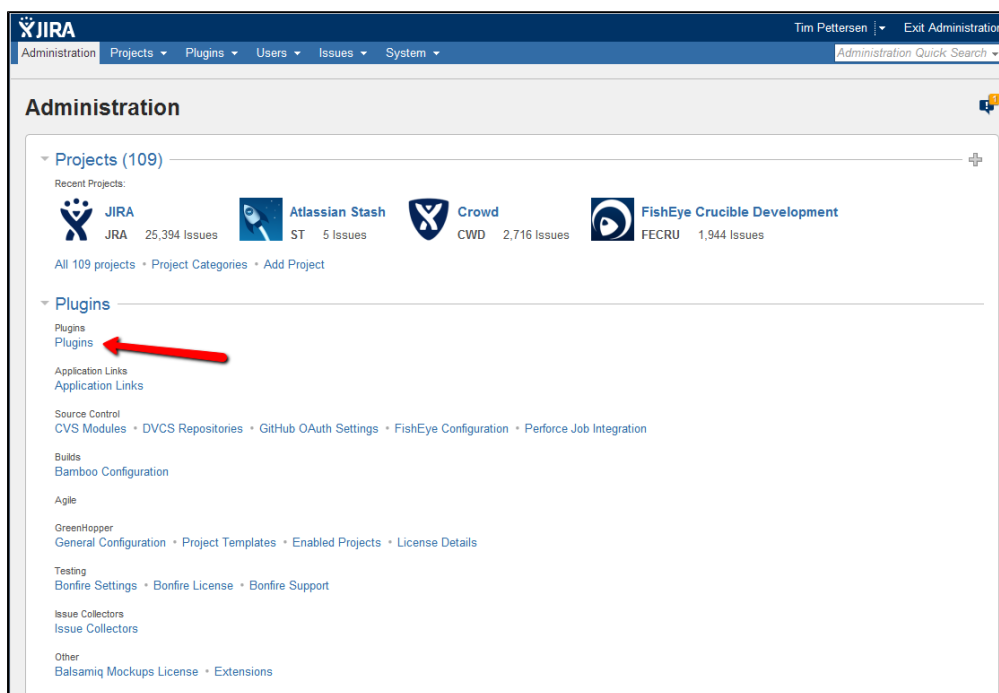
If you're using a version of JIRA earlier than **5.0.3** you may need to upgrade the FishEye/Stash plugin in JIRA to get support for Atlassian Stash.

JIRA Version	Compatibility	FishEye/Stash Plugin URL
5.0.4+	Works straight out of the box!	NA
5.0-5.0.3	Requires JIRA FishEye/Stash Plugin 5.0.4.1	https://maven.atlassian.com/content/repositories/atlassian-contrib/com/atlassian/jira/plugins/jira-fisheye-plugin/5.0.4.1/jira-fisheye-plugin-5.0.4.1.jar
Limited support for JIRA 4.4.x and earlier		
JIRA 4.3+ allows for showing commits associated with issues in JIRA. However, viewing issues within Stash is not supported for JIRA 4.4.x and earlier.		

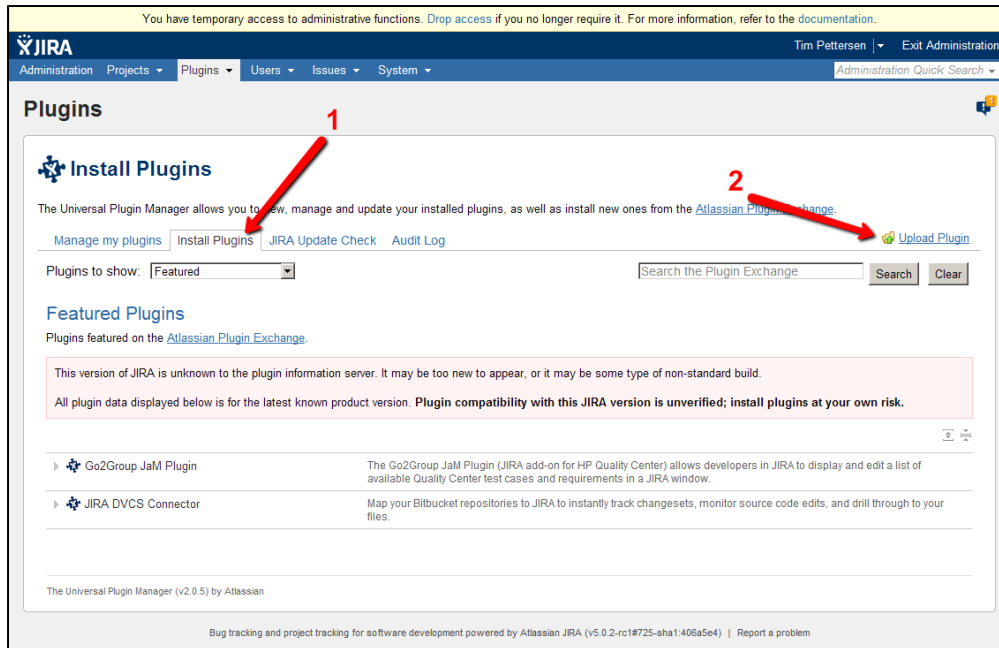
4.4.x	Requires JIRA FishEye/Stash Plugin 3.4.12	https://maven.atlassian.com/content/repositories/atlassian-contrib/com/atlassian/jira/plugins/jira-fisheye-plugin/3.4.12/jira-fisheye-plugin-3.4.12.jar
4.3.x	Requires JIRA FishEye/Stash Plugin 3.1.8	https://maven.atlassian.com/contrib/com/atlassian/jira/plugins/jira-fisheye-plugin/3.1.8/jira-fisheye-plugin-3.1.8.jar
Earlier versions	JIRA-to-Stash integration is unsupported	NA

Plugin upgrade guide

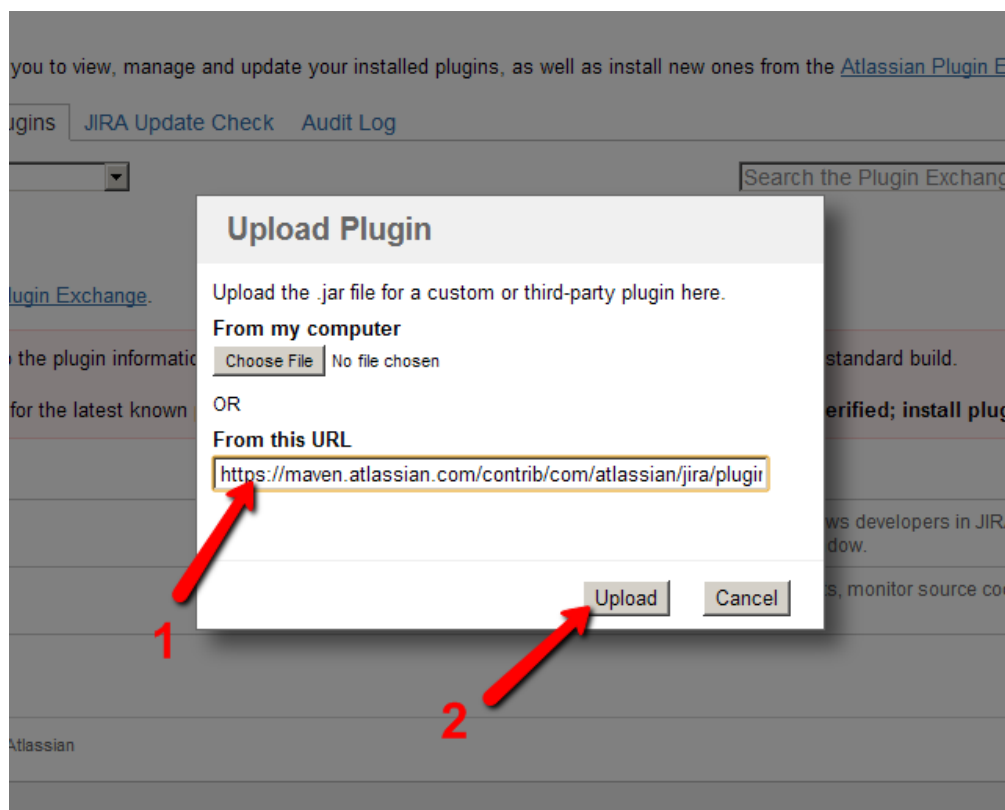
To upgrade the FishEye/Stash Plugin, copy the link from the table above that matches your JIRA version. Then navigate in JIRA to **Administration > Plugins**.



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.



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/Stash Plugin in JIRA 4.3](#) for more details.

Using custom JIRA issue keys with Stash

Stash assumes that JIRA issue keys are of the default format (that is, two or more uppercase letters (`[A-Z][A-Z]+`), followed by a hyphen and the issue number, for example `STASH-123`). By default, Stash will not recognise custom JIRA issue key formats.

You can use custom JIRA issue key formats with Stash by editing either the `<Stash installation directory>/bin/setenv.sh` or `<Stash installation directory>/bin/setenv.bat` file, depending upon the OS platform.

To override the default issue key format, pass the `-Dstash.jira.key.pattern=<some different regex>` parameter to the `JVM_SUPPORT_RECOMMENDED_ARGS` property, like this:

```
JVM_SUPPORT_RECOMMENDED_ARGS="-Dstash.jira.key.pattern=(<Some different regex>)"
```

For example, to use lowercase letters in issue keys, use a regex as in:

```
-Dstash.jira.key.pattern="((?!([a-z]{1,10})-?)[a-z]+-\d+)"
```

You will need to restart Stash.

Stash will index JIRA issues from this point onwards. If you want to re-index pre-existing commits in Stash you will have to delete the cached index in the Stash [home directory](#), located at `<Stash home directory>/cache/idx-snapshots`.

As always, please back up your home directory (and perhaps the database) before performing any manual operation on Stash. Consider testing this change on another copy of Stash before using it in production.

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 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. You need to have created the DBMS (such as MySQL) that you wish to migrate the Stash data to before running the Migration Wizard.

To run the Database Migration Wizard:

1. Log in to Stash.
2. In the administration area, click **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 will be 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 than 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.

MySQL 5.6.x Compatibility

Stash isn't compatible with MySQL 5.6.x because of bugs in its query optimizer ([#68424](#), [#69005](#)). We recommend using MySQL 5.5.x or 5.1.x. for the time being. Please watch [STASH-3164](#) for further updates on this.

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.

Related pages:

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

Prerequisites

Download and install the JDBC driver

The JDBC drivers for MySQL 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 your `<Stash home directory> /lib` directory (for Stash 2.1 or later).
4. Stop Stash, on [Windows](#), or on [Linux and Mac](#).
5. Restart Stash, on [Windows](#), or on [Linux and Mac](#).

If you configure your JDBC driver to use loadbalancing or failover with the Stash server, you may need to set `db`

`.pool.cache.statements=0` in the `stash-config.properties` file (in your [Stash home directory](#)). Please note that Stash does not yet (as of version 2.8) support DB redundancy, and that this configuration change is not supported by Atlassian.

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:

- configure the database to use the InnoDB storage engine
- 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 (to ensure case sensitivity)
- If MySQL is using binary logging, configure the database to use a binary logging format of either `MIXED` or `ROW`. Refer to the [MySQL documentation](#). Note that Stash sets the MySQL transaction isolation level to `READ-COMMITTED` when it connects to the database.

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 values):

```
mysql> SET GLOBAL storage_engine = 'InnoDB';
mysql> CREATE DATABASE stash CHARACTER SET utf8 COLLATE utf8_bin;
mysql> GRANT ALL PRIVILEGES ON stash.* TO 'stashuser'@'localhost' IDENTIFIED BY
'password';
mysql> FLUSH PRIVILEGES;
mysql> QUIT
```

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>.

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

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. In the administration area, click **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**.

Hostname	The host name or IP address of the computer running the database server.
Port	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.
Database name	The name of the database that Stash should connect to.
Database username	The username that Stash should use to access the database.
Database password	The password that Stash should use to access the database.

Migrate Database

Stash will be unavailable to users while a migration is in progress. See our [documentation](#) for more information.

Database Type:

Hostname*:

Port*:

Database name*:

Database username*:

Database password:

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:

[Prerequisites](#)
[Connect Stash to the Oracle database](#)
[Install the JDBC driver](#)

Related pages:

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

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.
 - i 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 `connect` and `resource` roles only. The `connect` role is required to set up a connection, while `resource` role is required to allow the user to create objects in its own schema.
- Create a local `all_objects` view to the user's schema, so that there is no possibility that a table with the same name as one of the Stash tables in another schema will cause any conflicts.

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

```
CREATE USER <user>
  IDENTIFIED BY <password>;
GRANT CONNECT, RESOURCE to <user>;
CREATE VIEW <user>.all_objects AS
  SELECT *
  FROM sys.all_objects
  WHERE owner = upper('<user>');
```

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;

GRANT CONNECT, RESOURCE to stash;
CREATE VIEW stash.all_objects AS
  SELECT *
  FROM sys.all_objects
  WHERE owner = upper('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.

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. In the Stash administration area, click **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**.

Hostname	The host name or IP address of the computer running the Oracle server.
Port	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.

Database name	The system identifier of the Oracle instance that Stash should connect to. Stash does not support connecting to Oracle servers which are using SID s or TNS Alias to identify themselves; it requires the fully qualified Service Name instead.
Database username	The username that Stash should use to access the database.
Database password	The password that Stash should use to access the database.

Migrate Database

Stash will be unavailable to users while a migration is in progress. See our [documentation](#) for more information.

Database Type

Hostname*

Hostname or IP address of the database server

Port*

TCP port number for the database server

Database name*

Database username*

Database password

Install the JDBC driver

This section is only relevant to some distributions of Stash, for example if you are running Stash via the Atlassian Plugin SDK, or have built Stash from source.

If the Oracle JDBC driver is *not* bundled with Stash, you will need to download and install the driver yourself.

1. Download the appropriate JDBC driver from the Oracle [download site](#).
2. Copy the downloaded jar file to your <Stash home directory>/lib directory (for Stash 2.1 or later).
3. Stop, then restart, Stash on [Windows](#), or on [Linux and Mac](#).

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:

- Install PostgreSQL where it is accessible to Stash.
- Create a database and user on the PostgreSQL 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 PostgreSQL database, or
 - at a later time, migrate Stash to the PostgreSQL database.

It is assumed here that you already have PostgreSQL installed and running. For more information about PostgreSQL installation and operation, refer to the [PostgreSQL documentation](#).

PostgreSQL has the idea of schemas. When you create a PostgreSQL database, a 'public' schema is created and set as the default for that database. It is possible to create a different schema (e.g. 'stash') and set that as the default schema. Stash will use whatever schema is set as the default for the logged-in user. Stash does not provide a way for a user to nominate the schema to use; it uses schema that is set as the PostgreSQL default.

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

On this page:

[Prerequisites](#)
[Connect Stash to the PostgreSQL database](#)
[Install the JDBC driver](#)

Related pages:

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

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.

```
CREATE ROLE stashuser WITH LOGIN PASSWORD 'jellyfish' VALID UNTIL 'infinity';  
  
CREATE DATABASE stash WITH ENCODING='UTF8' OWNER=stashuser CONNECTION LIMIT=-1;
```

Note that you must also add Stash to the PostgreSQL `pg_hba.conf` file, otherwise PostgreSQL will not allow the connection. See [Password authentication failed with PostgreSQL 9](#) for details.

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. In the Stash administration area, click **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**.

Hostname	The host name or IP address of the computer running the database server.
Port	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.
Database name	The name of the database that Stash should connect to.
Database username	The username that Stash should use to access the database.
Database password	The password that Stash should use to access the database.

Migrate Database

Stash will be unavailable to users while a migration is in progress. See our [documentation](#) for more information.

Database Type

Hostname*

Hostname or IP address of the database server

Port*

TCP port number for the database server

Database name*

Database username*

Database password

Install the JDBC driver

This section is only relevant to some distributions of Stash, for example if you are running Stash via the Atlassian Plugin SDK, or have built Stash from source.

If the PostgreSQL JDBC driver is *not* bundled with Stash, you will need to download and install the driver

yourself.

1. Download the appropriate JDBC driver from the Postgres [download site](#).
2. Copy the downloaded jar file to your <Stash home directory>/lib directory (for Stash 2.1 or later).
3. Stop, then restart, Stash on [Windows](#), or on [Linux and Mac](#).

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.

It is assumed here that you already have SQL Server installed and running.

- SQL Server documentation is available at <http://msdn.microsoft.com/en-us/library/bb545450.aspx>.
- JDBC documentation is available at <http://msdn.microsoft.com/en-us/library/ms378672.aspx>.

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

On this page:

[Prerequisites](#)
[Connect Stash to the SQL Server database](#)
[Use Integrated Authentication or 'Windows Authentication Mode' \(Optional\)](#)
[Install the JDBC driver](#)

Related pages:

- [Transitioning from jTDS to Microsoft's JDBC driver](#)
- [Connecting Stash to an external database](#)
- [Connecting Stash to MySQL](#)
- [Connecting Stash to Oracle](#)
- [Connecting Stash to PostgreSQL](#)

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:

Step	Notes
Create a database	e.g. <code>stash</code> . Remember this database name for the connection step below.
Set the collation type	This should be case-sensitive, for example, 'SQL_Latin1_General_CP1_CS_AS' (CS = Case Sensitive).

Set the isolation level	Configure the database to use the isolation level, Read Committed with Row Versioning.
Create a database user	e.g. <code>stashuser</code> . This database user should not be the database owner, but should be in the <code>db_owner</code> role. See SQL Server Startup Errors . Remember this database user name for the connection step below.
Set database user permissions	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.
Enable TCP/IP	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.
Check the authentication mode	<p>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'.</p> <p><i><u>Stash instances running on Windows are also able to support SQL Server databases running in 'Windows Authentication Mode'. This is described at the bottom of this page and it has to be manually configured: Connecting Stash to SQL Server - Use Integrated Authentication (Optional)</u></i></p>
Check that SET NOCOUNT is off	<p>Ensure that the SET NOCOUNT option is turned off. You can do that in SQL Server Management Studio as follows:</p> <ol style="list-style-type: none"> 1. Navigate to Tools > Options > Query Execution > SQL Server > Advanced. Ensure that the SET NOCOUNT option is cleared. 2. Now, go to the Server > Properties > Connections > Default Connections properties box and clear the no count option.

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):


```

SQL Server> CREATE DATABASE stash
SQL Server> GO
SQL Server> USE stash
SQL Server> GO
SQL Server> ALTER DATABASE stash SET ALLOW_SNAPSHOT_ISOLATION ON
SQL Server> GO
SQL Server> ALTER DATABASE stash SET READ_COMMITTED_SNAPSHOT ON
SQL Server> GO
SQL Server> ALTER DATABASE stash COLLATE SQL_Latin1_General_CP1_CS_AS
SQL Server> GO
SQL Server> SET NOCOUNT OFF
SQL Server> GO
SQL Server> USE master
SQL Server> GO
SQL Server> CREATE LOGIN stashuser WITH PASSWORD=N'password',
DEFAULT_DATABASE=stash, CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
SQL Server> GO
SQL Server> ALTER AUTHORIZATION ON DATABASE::stash TO stashuser
SQL Server> GO

```

This creates an empty 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.

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. In the Stash administration area, click **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**.

Hostname	The host name or IP address of the computer running the database server.
-----------------	--

Port	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.
Database name	The name of the database that Stash should connect to.
Database username	The username that Stash should use to access the database.
Database password	The password that Stash should use to access the database.

Migrate Database

Stash will be unavailable to users while a migration is in progress. See our [documentation](#) for more information.

Database Type

Hostname*

Hostname or IP address of the database server

Port*

TCP port number for the database server

Database name*

Database username*



Database password

Use Integrated Authentication or 'Windows Authentication Mode' (Optional)

Windows authentication is only available for Stash instances running on Windows. It cannot be used on Linux because Microsoft does not provide shared objects for it. You will either need to run Stash on Windows, allowing you to use Windows security, or you will need to enable mixed-mode authentication for SQL Server if you are running Stash on Linux. Unfortunately, there are no other options at this time.

Integrated authentication uses a native DLL to access the credentials of the logged-in user to authenticate with SQL Server. The native DLLs for both 32- and 64-bit systems are included in the distribution; there is no need to download the entire package from Microsoft.

Stash does not currently support configuring the system to use integrated authentication from the UI (Vote for it!

 **STASH-3035** - Add support for integrated authentication for Microsoft SQL Server ( **Open**)). This

means you can't currently migrate to SQL Server with integrated authentication, nor can you configure Stash to use SQL Server with integrated authentication during initial setup. However, if Stash has already been configured to use SQL Server (for example, when the Setup Wizard was run at first use), you can enable integrated authentication by directly modifying Stash's configuration, as follows:

1. Based on the JVM being used to run Stash, rename either the x64 or x86 DLL to `sqljdbc_auth.dll` in `lib/native`. Note that running on Windows x64 does *not* require the use of the x64 DLL; you should only use the x64 DLL if you are also using a 64-bit JVM.
2. In `setenv.bat`, a `JVM_LIBRARY_PATH` variable has already been defined. Simply remove the leading `%` from `em`. Note that if you are putting the native DLL in an alternative location, you may need to change the value to point to your own path. The value of the `JVM_LIBRARY_PATH` variable will automatically be

included in the command line when Tomcat is run using `start-stash.bat`.

3. Edit the `%STASH_HOME%\stash-config.properties` file to include `;integratedSecurity=true` in the `jdbc.url` line. Note that `jdbc.user` and `jdbc.password` will no longer be used to supply credentials *but they must still be defined* – Stash will fail to start if these properties are removed.
4. Ensure the Stash process or service is running as the correct user to access SQL Server.



It is also possible to configure integrated authentication over Kerberos, rather than using the native DLLs. Details for that are included in the [JDBC documentation](#).

Install the JDBC driver

This section is only relevant to some distributions of Stash, for example if you are running Stash via the Atlassian Plugin SDK, or have built Stash from source.

If the SQL Server JDBC driver is *not* bundled with Stash, you will need to download and install the driver yourself.

1. Download the appropriate JDBC driver from the Microsoft [download site](#).
2. Install the driver file to your `<Stash home directory>/lib` directory (for Stash 2.1 or later).
3. Stop, then restart, Stash on [Windows](#), or on [Linux and Mac](#).

If Stash was configured to use Microsoft SQL Server by manually entering a JDBC URL, please refer to [this guide](#).

Transitioning from jTDS to Microsoft's JDBC driver

This page describes how to change from using jTDS to using the Microsoft SQL Server JDBC driver to access Microsoft SQL Server.

What do I have to do?

If Stash was configured to use Microsoft SQL Server by following the steps outlined in [Connecting Stash to SQL Server](#), *no change is necessary*. However, If Stash was configured to use Microsoft SQL Server by **manually entering a JDBC URL**, the system will lock on startup if the driver class and URL are not manually updated.

How to proceed

In the [Stash home directory](#), `stash-config.properties` must be edited to change the JDBC driver and URL. The existing configuration should look similar to this:

```
jdbc.driver=net.sourceforge.jtds.jdbc.Driver
jdbc.url=jdbc:jtds:sqlserver://localhost:1433;databaseName=stash;
jdbc.user=stashuser
jdbc.password=secretpassword
```

The JDBC URL above is in the format constructed by Stash when Connecting Stash to SQL Server and will automatically be updated to a URL compatible with Microsoft's driver, with no change required on the administrator's part. If the URL contains additional properties, such as `domain=`, it will need to be manually updated.

To use Microsoft's SQL Server driver, the settings above would be updated to this:

```
jdbc.driver=com.microsoft.sqlserver.jdbc.SQLServerDriver
jdbc.url=jdbc:sqlserver://localhost:1433;databaseName=stash;
jdbc.user=stashuser
jdbc.password=secretpassword
```

The exact values to use in the new URL are beyond the scope of this documentation; they must be chosen based on the jTDS settings they are replacing.

Additional Information for the curious

The new JDBC driver class is: `com.microsoft.sqlserver.jdbc.SQLServerDriver`

The JDBC URL format for the jTDS driver is documented on SourceForge at <http://jtds.sourceforge.net/faq.html#urlFormat>.

The JDBC URL format for Microsoft's SQL Server driver is documented on MSDN at <http://msdn.microsoft.com/en-us/library/ms378428.aspx>, with documentation for additional properties at <http://msdn.microsoft.com/en-us/library/ms378988.aspx>.

Why change drivers?

▼ [Click here to find all the technical details...](#)

Recent releases of Hibernate, which Stash uses to simplify its persistence layer, have introduced a requirement that the JDBC drivers and connection pools used be JDBC4-compliant. JDBC4 was introduced with Java 6.

The jTDS driver used by releases prior to Stash 2.1 is a JDBC3 driver, compatible with Java 1.3, and therefore cannot be used with newer versions of Hibernate. While jTDS 1.3.0 implements JDBC4, and JDBC4.1 which is provided by Java 7, it *requires* a Java 7 runtime environment. Upgrading Stash to that version was a non-starter, as it would require raising the minimum Java version for the product to Java 7.

Instead, the decision was made to replace jTDS with Microsoft's own SQL Server driver. Microsoft's driver is actively maintained, where jTDS is only recently seeing its first updates in over 3 years, and supports all the features of SQL Server, including SQL Server 2012.

Stash attempts to automatically update jTDS JDBC URLs to values compatible with Microsoft's JDBC driver. However, for installations using custom JDBC URLs—for example, to use domain authentication—such automatic updating is not possible; the URL, which was manually entered, must be manually updated.

Migrating Stash to another server

This page describes how to move your Stash installation from one physical machine to a different machine. For most scenarios, the overall procedure involves the following 3 steps, although your situation may not require all of these:

1. Move the Stash data.
2. Move the Stash installation to the new location, and update `STASH_HOME`.
3. Update the Stash `stash-config.properties` file. This will be necessary if you were unable to use the Migration Wizard in Step 1.

See also the [Stash upgrade guide](#). You can upgrade Stash either before or after you migrate Stash. This page does *not* describe any aspect of the upgrade procedure.

On this page:

First steps

1. [Move the Stash data to a different machine](#)
2. [Move Stash to a different machine](#)
3. [Update the Stash configuration](#)

Related pages:

- [Supported platforms](#)
- [Connecting Stash to an external database](#)
- [Installing Stash on Windows](#)
- [Installing Stash on Linux and Mac](#)
- [Stash upgrade guide](#)

First steps

In preparation for migrating Stash to another server, check that you have done the following:

- Confirm that the operating system, database, other applicable platforms and hardware on the new machine will comply with the [requirements](#) for Stash.
- Check for any known migration issues in the [Stash Knowledge Base](#).
- Alert users to the forthcoming Stash service outage.

- Ensure that users will not be able to update existing Stash data during the migration. You can do this by temporarily changing the access permissions for Stash.
- Make sure you have [created a user in Stash](#) (not in your external user directory) that has System Administrator [global permissions](#) so as to avoid being locked out of Stash in case the new server does not have access to your external user directory.

1. Move the Stash data to a different machine

This section gives a brief overview of how to move the Stash data to a different machine. You do not need to do anything in this section if you will continue to use the embedded database - the Stash data is moved when you move the Stash installation.

The Stash data includes the data directories (including the Git repositories), log files, installed plugins, temporary files and caches.

You can move the Stash data:

- from the embedded database to a [supported](#) external DBMS.
- to another instance of the same DBMS.
- from one DBMS to another supported DBMS (for example, from MySQL to PostgreSQL).

You can also move the actual DBMS. Atlassian recommends that for large installations, Stash and the DBMS run on separate machines.

There are 2 steps:

1. Create and configure the DBMS in the new location. Please refer to [Connecting Stash to an external database](#), and the relevant child page, for more information.
2. Either:
 - If the new location is currently visible to Stash, use the Stash Database Migration Wizard. Please refer to [Connecting Stash to an external database](#), and the relevant child page, for more information.
 - If the new location is not currently visible to Stash (perhaps because you are moving to a new hosting provider), you need to perform a database export and then import the backup to the new DBMS. Please refer to the vendor documentation for your DBMS for detailed information. You will also need to update the `stash-config.properties` file in the `<Stash home directory>` as described below.

2. Move Stash to a different machine

This section describes moving the Stash installation to a different machine.

1. Stop Stash. Use `bin\stop-stash.bat` on Windows, or `bin/stop-stash.sh` on Linux and Mac.
2. Make an archive (such as a zip file) of the Stash home directory. The home directory contains data directories (including the Git repositories), log files, installed plugins, temporary files and caches. 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`.
3. Copy the archive to the new machine and unzip it to its new location there.
4. Set up an instance of Stash in the new location by doing one of the following:
 - Make an archive of the old Stash installation directory and copy it across to the new machine.
 - Install the same version of Stash from scratch on the new machine.
5. Redefine the value for `STASH_HOME`, mentioned in Step 2. above, in the new `<Stash installation directory>`, using the new location for your copied home directory. See either [Installing Stash on Windows](#) or [Installing Stash on Linux and Mac](#) for more information.
6. If you are continuing to use the Stash embedded database, or you used the Migration Wizard to move the Stash data, you should now be able to start Stash on the new machine and have all your data available. Use `bin\start-stash.bat` on Windows, or `bin/start-stash.sh` on Linux and Mac. Once you have confirmed that the new installation of Stash is working correctly, revert the access permissions for Stash to their original values.
7. If you moved the Stash data by performing a database export and import, carry on to Step 3. below to update the `stash-config.properties` file in the `<Stash home directory>`.

3. Update the Stash configuration

If you moved the Stash data by performing a database export, you must update the `stash-config.properties` file in the `<Stash home directory>` with the changed configuration parameters for the database connection.

The configuration parameters are described in [Stash config properties](#).

Once the configuration parameters are updated, you should be able to start Stash on the new machine and have all your data available. Use `bin\start-stash.bat` on Windows, or `bin/start-stash.sh` on Linux and Mac. Once you have confirmed that the new installation of Stash is working correctly, revert the access permissions for Stash to their original values.

Specifying the base URL for Stash

This is the base URL for this installation of Stash. All links *which are not from a web request* (for example in Stash email notifications), will be prefixed by this URL. If you are experiencing trouble with setting an `https` base URL, please ensure you have configured [Tomcat with SSL](#) correctly.

To specify Stash's base URL:

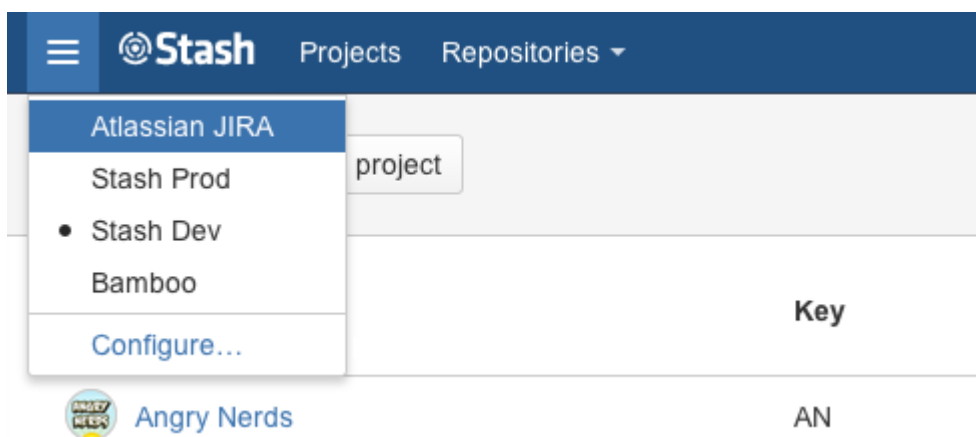
1. In the Stash administration area, click **Server settings** (under 'Settings').
2. In the **Base URL** field, type the URL address of your Stash server (for example, "`https://stash.mycompany.com`").
3. Click **Save**.

Related pages:

- [Administering Stash](#)

Configuring the application navigator

The application navigator, on the left of the Stash header, allows you to switch to your other applications, such as Atlassian JIRA and Bamboo – or any other web application – all from the Stash header:



Users only see the application navigator when links are set up – if there are no links, only administrators can see it.

Stash administrators can configure which apps appear in the navigator – just click **Configure** in the application navigator, or go to the Stash admin area and click **Application navigator**:

- [Linked applications](#) are automatically configured in the application navigator, and can't be deleted. Click **Manage** to configure those in the source application.
- Specify new links, as required by your users, by entering a **Name** and **URL**.
- Restrict the visibility of links to particular user groups, or hide the link completely. Click in a row, under the Groups column header, to edit those properties for existing rows.
- Use the 'handles' at the left to change the link order when seen in Stash.

Name	URL	Hide	Restricted to Groups	
<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>	<input type="button" value="Add"/>
Atlassian JIRA	https://jira.atlassian.com/	Manage		
Stash Prod	https://stash.atlassian.com			<input type="button" value="Delete"/>
Stash Dev	https://stash.dev.internal.atlassian.com/			
extranet-bamboo	https://extranet-bamboo.internal.atlassian.com/	Manage		

Managing add-ons

An add-on is an installable component that supplements or enhances the functionality of Stash in some way. For example, the [Custom Navigation Plugin](#) enables you to configure custom navigation tabs specific to a repository. Other add-ons are available for adding graphs to Stash, importing SVN source control projects into Stash, and accessing Atlassian support from Stash.

Stash comes with many pre-installed add-ons (called system add-ons). You can install more add-ons, either by acquiring the add-on from the [Atlassian Marketplace](#) or by uploading it from your file system. This means that you can install add-ons that you have developed yourself. For information about developing your own add-ons for Stash, see the [Stash Developer Documentation](#).

On this page:

- [About the Universal Plugin Manager \(UPM\)](#)
- [Administering add-ons in Stash](#)

About the Universal Plugin Manager (UPM)

You administer add-ons for Stash using the Universal Plugin Manager (UPM). The UPM is itself an add-on that exposes add-on administration pages in the Stash Administration Console. UPM works across Atlassian applications, providing a consistent interface for administering add-ons in Stash, Crucible, Confluence, Fisheye, JIRA and Bamboo.

UPM comes pre-installed in recent versions of all Atlassian applications, so you do not normally need to install it yourself. However, like other add-ons, the UPM software is subject to regular software updates. Before administering add-ons in Stash, therefore, you should [verify your version](#) of the UPM and update it if needed.

Administering add-ons in Stash

You can update UPM, or any add-on, from the UPM's own add-on administration pages. Additionally, you can perform these tasks from the UPM administration pages:

- Install or remove add-ons
- Configure add-on settings
- Discover and install new add-ons from the [Atlassian Marketplace](#)
- Enable or disable add-ons and their component modules

It shows only those plugins that are supported in your version of the product, so that you do not install incompatible plugins.

If the add-on request feature is enabled in your Atlassian application, non-administrative users can also discover add-ons on the Atlassian Marketplace. Instead of installing the add-ons, however, these users have the option of requesting the add-ons from you, the administrator of the Atlassian application.

For more information on administering the add-on request feature or performing other common add-on administration tasks, see the [Universal Plugin Manager documentation](#). For an end-user's view of requesting add-ons in Stash, see [Requesting add-ons](#).

POST service webhook for Stash

Repository administrators can add a POST service to a repository. Stash POSTs to the service URL you specify.

You can use an URL with the following format:

```
https://server:port/path/
```

The service receives a POST whenever the user pushes to the repository.

The content type header of the POST has an `application/json` type. The content is a JSON payload that represents the repository push.

Setting up the POST service

You can either set up the POST service manually or you can write a service to automate this. You would write a service if you are integrating an application with Stash.

Manually from Repository Administration

1. Go to the repository's settings.
2. Click **Hooks** in the left-hand navigation.
3. Select the **Post-Receive Webhooks** item from the listing.
4. Click **Enable**. A new dialog appears for the **POST** hook. You can add up to 5 hooks:

Post-Receive WebHooks

Description

Stash will send a POST request to the URLs provided after code has been pushed to your Git repository. The body of the POST request contains information about the repository where the change originated, a list of certain commits, and the user that made the push

You can find the details in [our documentation](#).

URL *

[Add another URL](#)

5. Enter the URL where Stash should send its update messages.
6. Press **Enable**.

POST data

When a user pushes to a repository, Stash POSTs to the URL you provided. The body of the POST request contains information about the repository where the change originated, a list of recent commits, and the user that made the push.

Example of payload

This is an example of a push that contains one commit that changes 2 files (*pom.xml*) in folders *iridium-common* and *iridium-magma*.

JSON Payload

```
{
  "repository":{
    "slug":"iridium-parent",
    "id":11,
    "name":"iridium-parent",
    "scmId":"git",
    "state":"AVAILABLE",
    "statusMessage":"Available",
    "forkable":true,
    "project":{
      "key":"IR",
      "id":21,
      "name":"Iridium",
      "public":false,
      "type":"NORMAL",
      "isPersonal":false
    },
    "public":false
  },
  "refChanges":[
    {
      "refId":"refs/heads/master",
      "fromHash":"2c847c4e9c2421d038fff26ba82bc859ae6ebe20",
      "toHash":"f259e9032cdeb1e28d073e8a79a1fd6f9587f233",
      "type":"UPDATE"
    }
  ],
  "changesets":{
    "size":1,
    "limit":100,
    "isLastPage":true,
    "values":[
      {
        "fromCommit":{
          "id":"2c847c4e9c2421d038fff26ba82bc859ae6ebe20",
          "displayId":"2c847c4"
        },
        "toCommit":{
          "id":"f259e9032cdeb1e28d073e8a79a1fd6f9587f233",
          "displayId":"f259e90",
          "author":{
            "name":"jhocman",
            "emailAddress":"jhocman@atlassian.com"
          },
          "authorTimestamp":1374663446000,
          "message":"Updating poms ...",
          "parents":[
            {
              "id":"2c847c4e9c2421d038fff26ba82bc859ae6ebe20",
              "displayId":"2c847c4"
            }
          ]
        },
        "changes":{
          "size":2,
          "limit":500,
          "isLastPage":true,
          "values":[
            {
              "contentId":"2f259b79aa7e263f5829bb6e98096e7ec976d998",

```

```

        "path":{
            "components":[
                "iridium-common",
                "pom.xml"
            ],
            "parent":"iridium-common",
            "name":"pom.xml",
            "extension":"xml",
            "toString":"iridium-common/pom.xml"
        },
        "executable":false,
        "percentUnchanged":-1,
        "type":"MODIFY",
        "nodeType":"FILE",
        "srcExecutable":false,
        "link":{
            "url":"/projects/IR/repos/iridium-parent/commits/f259e9032cdeble28d073e8a79a1fd6f95
            87f233#iridium-common/pom.xml",
            "rel":"self"
        }
    },
    {
        "contentId":"2f259b79aa7e263f5829bb6e98096e7ec976d998",
        "path":{
            "components":[
                "iridium-magma",
                "pom.xml"
            ],
            "parent":"iridium-magma",
            "name":"pom.xml",
            "extension":"xml",
            "toString":"iridium-magma/pom.xml"
        },
        "executable":false,
        "percentUnchanged":-1,
        "type":"MODIFY",
        "nodeType":"FILE",
        "srcExecutable":false,
        "link":{
            "url":"/projects/IR/repos/iridium-parent/commits/f259e9032cdeble28d073e8a79a1fd6f95
            87f233#iridium-magma/pom.xml",
            "rel":"self"
        }
    }
],
"start":0,
"filter":null
},
"link":{
    "url":"/projects/IR/repos/iridium-parent/commits/f259e9032cdeble28d073e8a79a1fd6f95
    87f233#iridium-magma/pom.xml",
    "rel":"self"
}
},
"start":0,

```

```

    "filter":null
  }
}

```

Properties

Some of the system-wide properties for the Webhook Plugin can be overridden in the [Stash configuration file](#).

Property	Description
<code>plugin.com.atlassian.stash.plugin.hook.threadPoolCoreSize=2</code>	Core size of thread pool – the default number of concurrent hooks notifications.
<code>plugin.com.atlassian.stash.plugin.hook.threadPoolMaxSize=3</code>	Maximal size of thread pool – the maximum number of concurrent hooks notifications.
<code>plugin.com.atlassian.stash.plugin.hook.queueSize=1024</code>	The maximum size of the queue which holds queued requests that are yet to be sent. When this size is exceeded the oldest unsent message will be dropped and a warning message logged.
<code>plugin.com.atlassian.stash.plugin.hook.connectionTimeout=10000</code>	Connection timeout for hook request in MILLISECONDS . When the connection times out a warning message will be logged.
<code>plugin.com.atlassian.stash.plugin.hook.changesetsLimit=500</code>	Limit of maximum count of changesets that will be sent in the POST data for a single ref change.
<code>plugin.com.atlassian.stash.plugin.hook.changesLimit=100</code>	Limit of maximum count of changes for a single changeset in the POST data.

Audit logging in Stash

Stash comes with an internal audit system enabled by default at installation. The audit system is intended to give administrators an insight into the way Stash is being used. The audit system could be used to identify authorized and unauthorized changes, or suspicious activity over a period of time.

Viewing recent events

Stash administrators and system administrators can see a list of recent events for each project and repository in the 'Audit log' view. This is found under the **Settings** tab for a project or repository, and shows only the most important audit events.

User	Action	Details	Time
Administrator	RestrictedRefRemovedEvent	{"id":9,"value":"master"}	A moment ago
Administrator	RestrictedRefAddedEvent	{"id":9,"value":"master","users":["admin"]}	A moment ago
Administrator	RepositoryModifiedEvent	{"old.isForkable":true,"new.isForkable":false}	A moment ago
Administrator	RepositoryPermissionGrantedEvent	{"permission":"REPO_READ","group":"stash-users"}	A moment ago
Administrator	RepositoryPermissionGrantedEvent	{"permission":"REPO_ADMIN","user":"user"}	A moment ago
Administrator	RepositoryHookEnabledEvent	{"project":"AP","repository":"core","hook":"com.atlassian.stash.stash-bundled-hooks.force-push-hook"}	A moment ago
Administrator	RepositoryCreatedEvent	{"project":"AP","repository":"core"}	1 min ago

The audit log displays a subset of the events recorded in the log file and is kept to a configurable maximum size (the default is 500 events). See [Audit events in Stash](#) for more details.

Accessing the audit log file

The full audit log file records a wide range of events in Stash. See [Audit events in Stash](#) for a list of these. The volume of events that are logged is coarsely configurable by changing a Stash server setting. See [Stash config properties](#) for more details.

You can find the log file in the `<Stash home directory>/log/audit` directory.

The log file will roll daily and also when it grows past a maximum size of 25 MB. There is a limit (currently 100) to the number of rolled files that Stash will keep. When the limit is reached, the oldest file is deleted each day.

⚠️ Note that you will need to back up the log files before they are removed, if your organisation needs to keep copies of those.

Configuring audit logging

There are [various system properties](#) that can be used to configure audit logging in Stash.

Audit events in Stash

The auditing component of Stash will log many different events that occur when Stash is being used. The events have been assigned priorities based on how important they are – these priorities can be used to control how much information is added to the audit log file. For example, if you have a server under high load and no need for auditing, you may wish to turn audit logging off by setting it to `NONE` – see the [audit log config properties](#).

Server level events


Event	Description	Priority
ApplicationConfigurationChangedEvent	The server configuration has changed e.g. the display name or the base url.	HIGH
BackupEvent	Audited at the beginning and the end of a system backup .	HIGH
LicenseChangedEvent	The server license has changed.	HIGH
MailHostConfigurationChangedEvent	The servers mail host has changed (used to send email notifications).	HIGH
MigrationEvent	Audited at the beginning and the end of a database migration .	HIGH
ServerEmailAddressChangeEvent	The server email address has changed (used in email notifications).	HIGH

TicketRejectedEvent	Certain resources (e.g. the Git processes) are throttled, when tickets are rejected (e.g. too many Git processes are in use) this event is fired.	LOW
---------------------	---	-----





User management events

Event	Description	Priority
DirectoryCreatedEvent	Occurs when a new directory is created.	HIGH
DirectoryDeletedEvent	Occurs when a new directory is deleted.	HIGH
GroupCreatedEvent	Occurs when a new group is created in the internal directory.	HIGH
GroupUpdatedEvent	Occurs when a new group is updated (not when membership changes) in the internal directory.	HIGH
GroupDeletedEvent	Occurs when a new group is deleted from the internal directory.	HIGH
GroupMembershipCreatedEvent	Occurs when a user is added to a group in the internal directory.	HIGH
GroupMembershipDeletedEvent	Occurs when a user is removed from a group in the internal directory.	HIGH
UserAuthenticatedEvent	Occurs when a user is successfully authenticated (logged in).	LOW
UserAuthenticationFailedInvalidAuthenticationEvent	Occurs whenever a user fails to authenticate. Note that this can occur frequently in Stash whenever a command line CLI is used as the initial URL provided to Stash contains a username but no password, which is rejected by Crowd.	MEDIUM
UserCreatedEvent	Occurs when a user is created in the internal directory.	HIGH
UserCredentialUpdatedEvent	Occurs when a user changes password in the internal directory.	HIGH
UserDeletedEvent	Occurs when a user is deleted from the internal directory.	HIGH
UserRenamedEvent	Occurs when the username of a user is changed in the internal directory.	HIGH

Permission events



 in the table below indicates that the event is visible in the recent audit log screen for the project or repository.

Event	Description	Priority
-------	-------------	----------


GlobalPermissionGrantedEvent	Occurs when a user or group is granted a global permission (e.g. create project).	HIGH
GlobalPermissionRevokedEvent	Occurs when a user or group has a global permission revoked.	HIGH
ProjectPermissionGrantedEvent	Occurs when a user or group is granted a permission for a specific project. 	HIGH
ProjectPermissionRevokedEvent	Occurs when a user or group has a permission for a specific project revoked. 	HIGH
RepositoryPermissionEvent	Occurs when a user or group has a permission for a specific repository altered. 	HIGH
RestrictedRefEvent	Children of this event are fired when a restricted ref is altered. 	HIGH


Project events



 in the table below indicates that the event is visible in the recent audit log screen for the project.

Event	Description	Priority
ProjectAvatarUpdatedEvent	Raised when a project avatar has been successfully updated.	LOW
ProjectCreatedEvent	Raised when a project is created. 	HIGH
ProjectCreationRequestedEvent	Raised just before a project is created; can be cancelled.	LOW
ProjectModifiedEvent	Raised when a project has been successfully updated (e.g. the project name). 	HIGH
ProjectModificationRequestedEvent	Raised just before a project is updated; can be cancelled.	LOW
ProjectDeletedEvent	Raised when a project is deleted.	HIGH
ProjectDeletionRequestedEvent	Raised just before a project is deleted; can be cancelled.	LOW

Repository events

 in the table below indicates that the event is visible in the recent audit log screen for the project or repository.

Event	Description	Priority
RepositoryAccessedEvent	Raised when a repository is accessed by a user. Stash currently only fires this event selectively - when users hit a repository page.	LOW
RepositoryCreatedEvent	Raised when a repository is created. 	MEDIUM

RepositoryCreationFailedEvent	Raised when an attempt to create a repository fails.	LOW
RepositoryCreationRequestedEvent	Raised just before a repository is created; can be cancelled.	LOW
RepositoryForkedEvent	Raised when a repository is forked successfully. 	MEDIUM
RepositoryForkFailedEvent	Raised when an attempt to fork a repository fails.	LOW
RepositoryForkRequestedEvent	Raised just before a repository is forked; can be cancelled.	LOW
RepositoryDefaultBranchModifiedEvent	Raised when the default branch of a repository is reconfigured (typically through repository settings).	LOW
RepositoryDeletedEvent	Raised when a repository is deleted. 	HIGH
RepositoryDeletionRequestedEvent	Raised just before a repository is deleted; can be cancelled.	LOW
RepositoryOtherReadEvent	Raised when the server uploads a pack file to the client via HTTP.	LOW
RepositoryOtherWriteEvent	Raised when the server receives a pack file from the client via HTTP.	LOW
RepositoryPullEvent	Raised when a Git client pulls from a repository (only when new content is sent to the client).	LOW
RepositoryPushEvent	Raised when a Git client pushed to a repository.	LOW

Pull request events

Event	Description	Priority
PullRequestEvent	Fired at different points in the pull request lifecycle (declined, commented, merged, opened, reopened, rescoped [code updated], updated, approved, unapproved, participants updated).	LOW

Plugin events

See this [plugin documentation](#) for details of when these events below are triggered.

Event	Description	Priority
PluginDisabledEvent	Occurs when a plugin has been disabled, either by the system or a user.	MEDIUM
PluginEnabledEvent	Occurs when a plugin has been enabled, either by the system or a user.	MEDIUM

PluginModuleDisabledEvent	Occurs when a plugin module has been disabled, either by the system or a user.	MEDIUM
PluginModuleEnabledEvent	Occurs when a plugin module has been enabled, either by the system or a user.	MEDIUM
PluginModuleUnavailableEvent	Signifies a plugin module is now unavailable outside the usual installation process.	MEDIUM
PluginUninstalledEvent	Occurs when a plugin is explicitly uninstalled (as opposed to as part of an upgrade).	MEDIUM
PluginUpgradedEvent	Signifies that a plugin has been upgraded at runtime.	MEDIUM
PluginContainerUnavailableEvent	Occurs when the container of a plugin is being shutdown, usually as a result of the server being stopped.	LOW
PluginModuleAvailableEvent	Signifies that a plugin module is now available outside the usual installation process.	LOW
PluginFrameworkStartedEvent	Signifies that the plugin framework has been started and initialized.	LOW

Advanced actions

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

In this section:

- [Scaling Stash](#)
- [Stash debug logging](#)
- [Stash config properties](#)
- [Enabling SSH access to Git repositories in Stash](#)
- [Proxying and securing Stash](#)
- [Moving Stash to a different context path](#)
- [Changing the port that Stash listens on](#)
- [Running Stash with a dedicated user](#)
- [Data recovery and backups](#)

Related pages:

- [Administering Stash](#)
- [Supported platforms](#)
- [Stash FAQ](#)

Scaling Stash

Hardware requirements

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

- The number and 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 size of your repositories – there are many operations in Stash that require more memory and more CPUs when working with very large repositories. Furthermore, huge Git repositories (larger than a few GBs) are likely to impact the performance of the Git client – see [this discussion](#).

- The number of users.

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. Note that enabling the [SCM Cache Plugin](#) (bundled with Stash from version 2.5.0) can help to reduce the cloning load on the Stash server due to CI polling. See [Scaling Stash for Continuous Integration performance](#).
- Estimate or calculate the average repository size and allocate 1.5 x number of concurrent clone operations x min(repository size, 700MB) of memory.

On this page:

- [Hardware requirements](#)
- [Understanding Stash's resource usage](#)
 - [Memory](#)
 - [CPU](#)
 - [Clones examined](#)
- [Configuring Stash scaling options and system properties](#)
- [Database requirements](#)

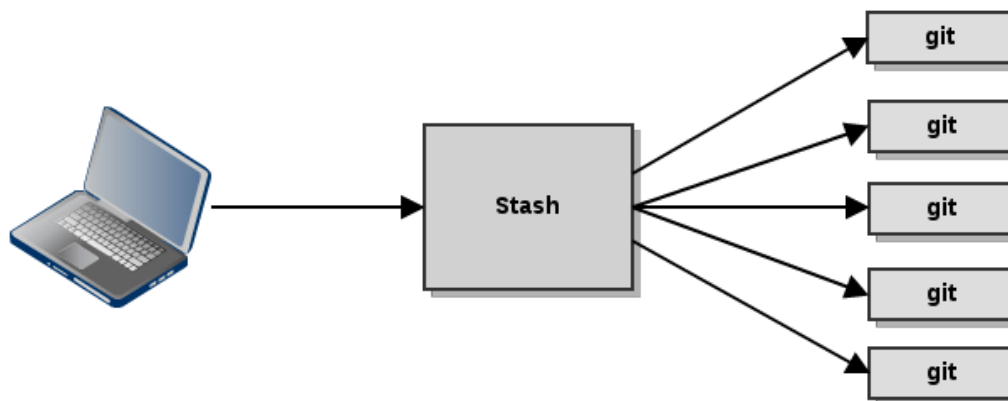
Related pages:

- [Using Stash in the enterprise](#)
- [Resources for migrating to Git](#)
- [Stash production server data](#)
- [Scaling Stash for Continuous Integration performance](#)
- [Potential performance impact of embedded Crowd directory ordering](#)
- [Stash config properties](#)

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.

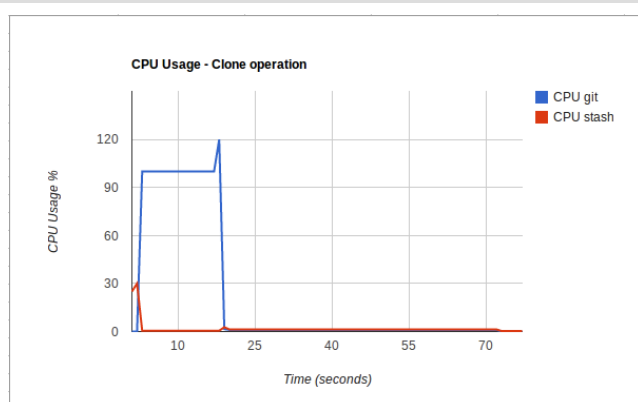
For users that connect to Stash using SSH, the encryption of data adds to overall CPU usage. For day-to-day push and pull operations the overhead will not be significant, but when cloning repositories the overhead will be noticeable.



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:

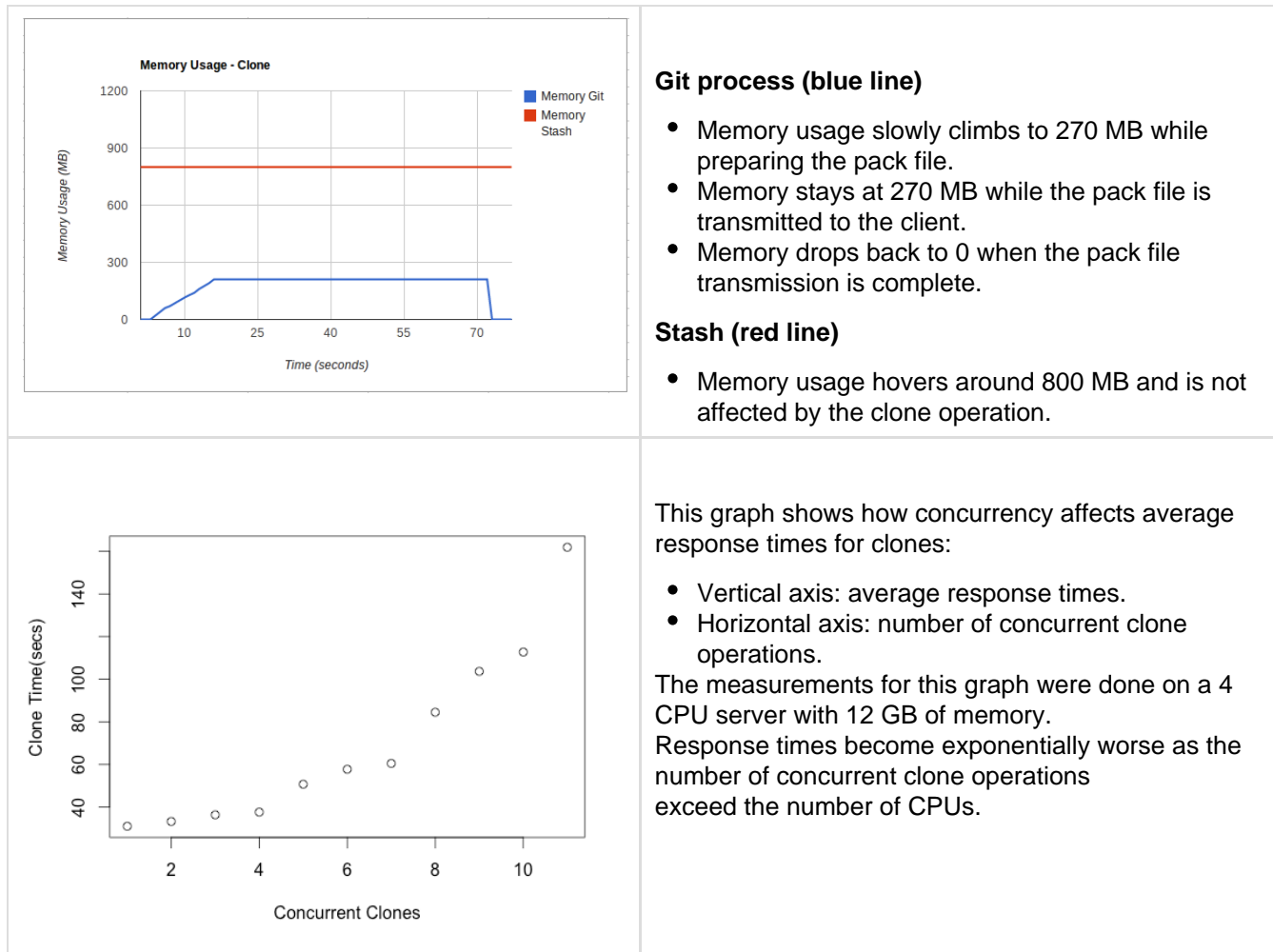


Git process (blue line)

- 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.



Configuring Stash scaling options and system properties

Stash limits the number of Git operations that can be executed concurrently, to prevent the performance for all clients dropping below acceptable levels. These limits can be adjusted – see [Stash config properties](#).

Database requirements

The size of the database required for Stash depends in large part on the number of repositories and the number of commits in those repositories.

A very rough guideline is: $100 + ((\text{total number of commits across all repos}) / 2500)$ MB.

So, for example, for 20 repositories with an average of 25,000 commits each, the database would need $100 + (20 * 25,000 / 2500) = 300$ MB.

Scaling Stash for Continuous Integration performance

If you've got CI or other automatic tooling set up to poll Stash for changes, you can end up with high load on your Stash server. Consider for instance a CI server that has a number of builds set up for a given repository. Each of those builds polls Stash for changes and when it detects a change, it starts a new build. If your CI server supports parallel and/or chained build steps, each of these builds typically results in multiple clone operations of the same repository. The result: lots of polling for changes, and bursts of clones of a repository.

Caching

CI results in highly repetitive calls to Stash: polling for changes typically results in the same response 90% of the time and a build triggers a number of identical clone calls to Stash. Both operations can benefit greatly from caching when you experience repetitive load from CI.

Stash 2.5, and later versions, ship with the SCM Cache plugin already bundled. The plugin is enabled by default, but note that ref advertisement is disabled by default – see the 'Limitations' section below. The plugin is also available from the Atlassian [Marketplace](#).

On this page:

- [Limitations](#)
- [Enabling and disabling caching](#)
- [Configuration](#)
- [REST API](#)

Related pages:

- [Scaling Stash](#)
- [Stash config properties](#)

Limitations

- Caching can be applied to 'ref advertisement' (polling for changes), clone and shallow clone requests only. Fetch/pull operations are not cached, but these operations will still benefit from the 'ref advertisement' cache.
- As a precaution, ref advertisement caching is *disabled* by default. The openness of the plugin system means that plugins (or manually installed git hooks) could be updating refs in repository without the caching plugin detecting these changes. The result would be a stale refs cache and failing clone/fetch operations. However, if you know that there are no plugins or git hooks installed that make changes to the repository directly, you can enable the ref advertisement caching using the system properties listed in the Configuration section below. Note that as an additional precaution, the ref advertisement caches are configured to automatically expire after a minute.

Enabling and disabling caching

Enable the SCM Cache Plugin from the admin area in Stash. Click **Manage Add-ons** (under 'Add-Ons') and filter for system add-ons. Click **SCM Cache Plugin for Stash** and then either **Enable** or **Disable**.

Configuration

The SCM Cache Plugin for Stash can be configured using either the REST endpoint (some settings, not all) or the system properties in `<STASH-HOME>/stash-config.properties`. Settings configured through REST are considered *before* the settings in `stash-config.properties`.

See [Stash config properties](#) for descriptions of the available system properties.

REST API

Method	Url	Description
GET	<code>/rest/scm-cache/latest/config/protocols</code>	Retrieve the protocols for which caching has been enabled.
PUT	<code>/rest/scm-cache/latest/config/protocols/{protocol}</code>	Enable caching of hosting requests for the protocol (HTTP or SSH).
DELETE	<code>/rest/scm-cache/latest/config/protocols/{protocol}</code>	Disable caching of hosting requests for the protocol (HTTP or SSH).
GET	<code>/rest/scm-cache/latest/config/refs/enabled</code>	Retrieve whether ref advertisement caching is enabled (true) or disabled (false).
PUT	<code>/rest/scm-cache/latest/config/refs/enabled/{status}</code>	Enable (status = true) or disable (status = false) ref advertisement caching.
GET	<code>/rest/scm-cache/latest/config/refs/ttl</code>	Retrieve the expiry in seconds for the ref advertisement caches.

PUT	/rest/scm-cache/latest/config/refs/ttl/{expiryInSeconds}	Set the expiry in seconds for the ref advertisement caches.
GET	/rest/scm-cache/latest/config/upload-pack/enabled	Retrieve the whether clone caching is enabled (true) or disabled (false).
PUT	/rest/scm-cache/latest/config/upload-pack/enabled/{status}	Enable (status = true) or disable (status = false) clone caching.
GET	/rest/scm-cache/latest/config/upload-pack/ttl	Retrieve the expiry in seconds for the clone caches.
PUT	/rest/scm-cache/latest/config/upload-pack/ttl/{expiryInSeconds}	Set the expiry in seconds for the clone caches.
GET	/rest/scm-cache/latest/caches	Retrieve information about the current caches: size, number of cache hits and misses, etc.
DELETE	/rest/scm-cache/latest/caches	Clear all caches.
GET	/rest/scm-cache/latest/caches/{projectKey}/{repoSlug}	Retrieve information about the caches for the repository identified by projectKey and repoSlug: size, number of cache hits and misses, etc.
DELETE	/rest/scm-cache/latest/caches/{projectKey}/{repoSlug}	Clear the caches for the repository identified by projectKey and repoSlug.

Stash production server data

This page provides some data around the Stash production instance that we run internally at Atlassian. We're providing this to give some idea of how Stash performs in a production environment. Please realise that this information is entirely specific to this particular instance – the details of your own installation may result in different performance data.

This data was collected with New Relic in February 2013, when the server was running a pre-release version of Stash 2.2.

On this page:

- [Hardware](#)
- [Load](#)
- [Server load](#)
- [Git operations](#)

Hardware

The Atlassian Stash production server runs on:

- Virtualised hardware
- 4 Hyper-threaded cores
- 12 GB RAM

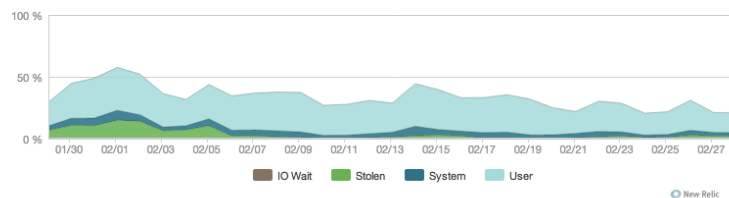
Load

Load data summary for February 2013:

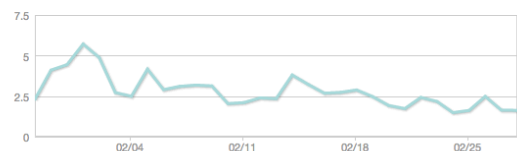
Type	Load
CPU usage	less than 30% on average
Load average	less than 3 on average
Physical Memory	peaked at 31%
Processes	Git: 17.3% CPU Java: 18.8% CPU
Clones	on average less than 300ms
Git operations/hour	peaking at 11,000 with an average of about 3,500
Concurrent connections/hour	peaking at 100 connections with an average of about 40 concurrent connections
CI running against Stash server	3 build servers with approximately 300 agents

Server load

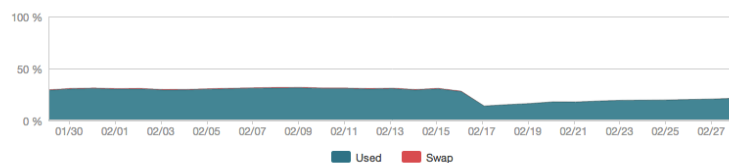
CPU usage



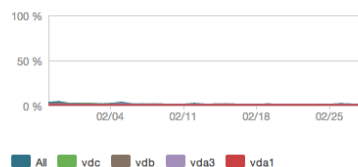
Load average



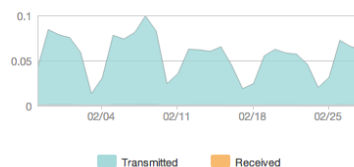
Physical memory



Disk I/O utilization



Network I/O (Gb/s)



stash-prod.private.atlassian.com

8 CPUs
12 GB RAM
Intel QEMU

Scientific Linux release 6.2
(Carbon)
Linux 2.6.32-220.7.1.el6.x86_64
x86_64
New Relic agent 1.1.2.124

Apps	Response time	Throughput	Errors
j2ee_stash-prod.private.atlassian.com	1,290 ms	570 rpm	1.46 %

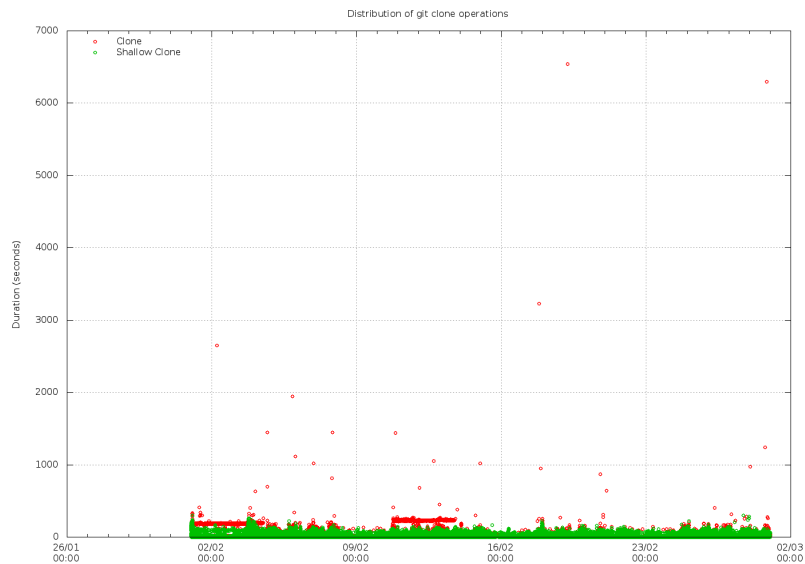
Processes >	User	Count	CPU	Memory
git	j2ee_stash-prod.private.atlassian.com-run	5	17.3%	226 MB
java	j2ee_stash-prod.private.atlassian.com-run	1	13.8%	836 MB
bzip2	root	1	7.0%	7.01 MB
portreserve	root	2	6.2%	393 MB
rsync	j2ee_stash-prod.private.atlassian.com-run	1	6.2%	90.8 MB

65 more...

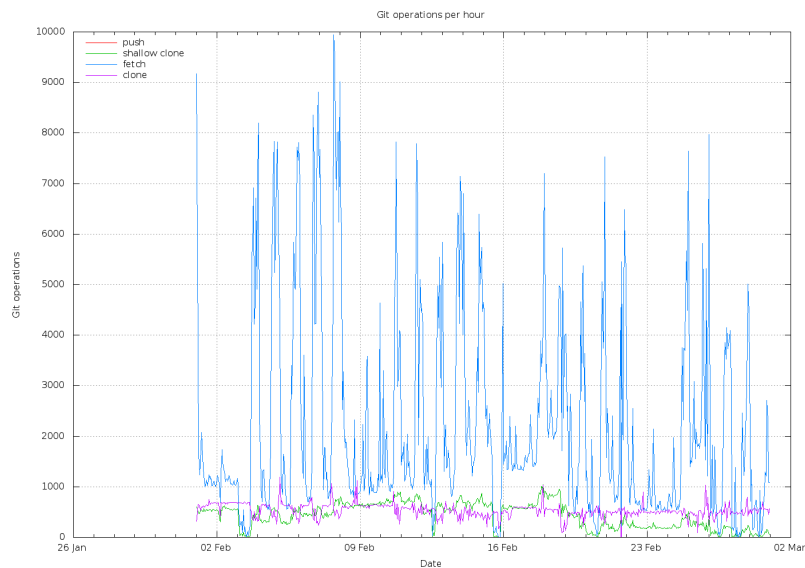
Recent stash-prod.private.atlassian.com events

Git operations

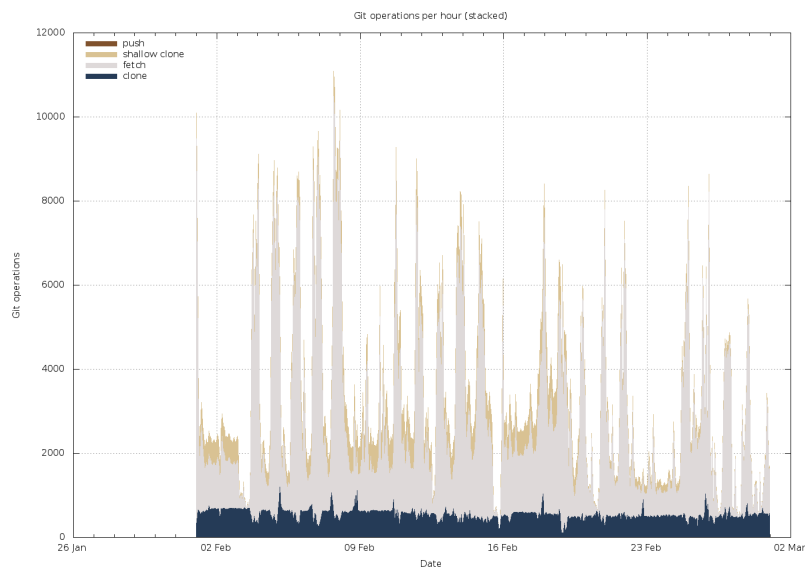
Git clone operations



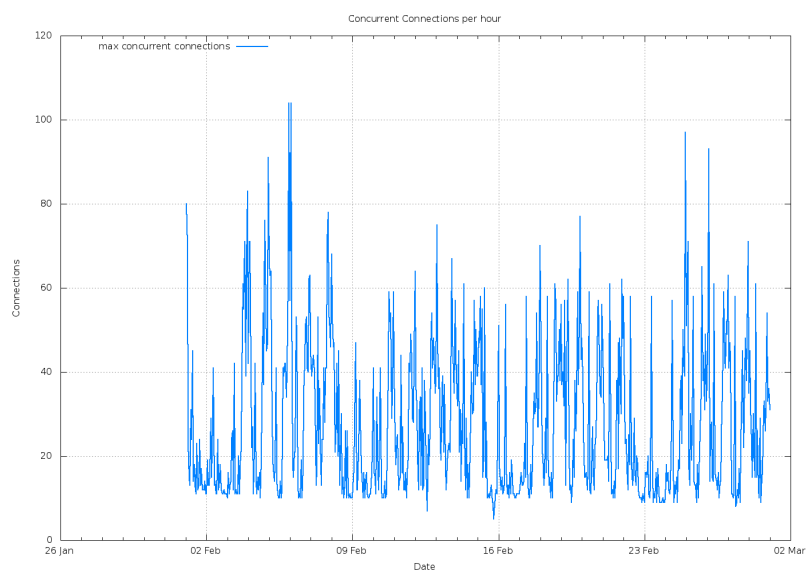
Git operations per hour



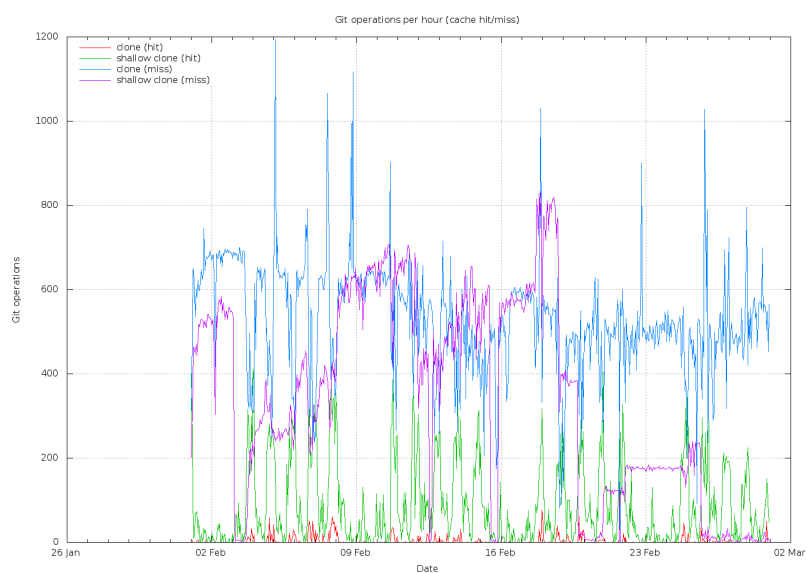
Git operations per hour (stacked)



Concurrent connections per hour



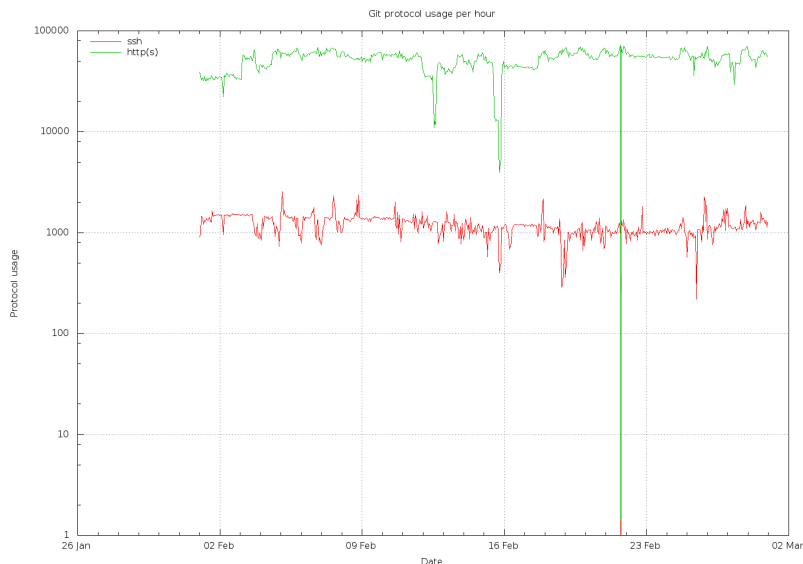
Git operations - cache hit/miss



Git operations - cache hit/miss



Git protocol usage per hour



Stash debug logging

On this page:

- [Debug logging for the Stash server](#)
 - [Enabling debug logging via the UI](#)
 - [Enabling debug logging on startup](#)
 - [Enabling debug logging at runtime](#)
- [Profiling logging for the Stash server](#)
 - [Enabling profiling logging via the UI](#)
- [Debug logging for Git operations on the client](#)
 - [On Linux](#)
 - [On Windows](#)

Debug logging for the Stash server

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

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 via the UI

To enable debug logging, go to the Stash admin area, choose **Logging and Profiling** (under 'Support') and select **Enable debug logging**.

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 two lines:

```
logging.logger.ROOT=DEBUG
logging.logger.com.atlassian.stash=DEBUG
```

Enabling debug logging at runtime

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

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

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

Profiling logging for the Stash server

This section describes how to enable profiling in Stash. This log is essential when troubleshooting performance issues. Stash logs can be found in <STASH_HOME>/log.

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

Enabling profiling logging via the UI

To turn on detailed trace information, go to the Stash admin area, choose **Logging and Profiling** (under 'Support') and select **Enable profiling**.

Debug logging for Git operations on the client

Atlassian Support might request DEBUG logs for Git operations (on the client) when troubleshooting issues. You can enable DEBUG logging on the Git client by setting the following variables.

On Linux

Execute the following in the command line before executing the Git command:

```
export GIT_TRACE_PACKET=1
export GIT_TRACE=1
export GIT_CURL_VERBOSE=1
```

On Windows

Execute the following in the command line before executing the Git command:

```
set GIT_TRACE_PACKET=1
set GIT_TRACE=1
set GIT_CURL_VERBOSE=1
```

Stash config properties

This page describes Stash system properties that can be used to control aspects of the behaviour in Stash. System properties are contained in the `stash-config.properties` file, in your [Stash home directory](#).

Stash must be restarted for changes to become effective.

Default values for system values, where applicable, are specified in the tables below.

On this page:

- [Audit](#)
- [Authentication](#)
- [Avatars](#)
- [Changesets](#)
- [Changeset indexing](#)
- [Database](#)
- [Database pool](#)
- [Downloads](#)
- [Events](#)
- [Executor](#)
- [Features](#)
- [Hibernate](#)
- [JIRA](#)
- [Liquibase](#)
- [Logging](#)
- [Notifications](#)
- [Paging](#)
- [Password reset](#)
- [Process execution](#)
- [Pull requests](#)
- [Readme parsing](#)
- [Ref metadata](#)
- [Resource throttling](#)
- [SCM – Cache](#)
- [SCM – Git](#)
- [Server busy banners](#)
- [SMTP](#)
- [SSH command execution](#)
- [Syntax highlighting](#)

Audit

Property	Description
<code>audit.highest.priority.to.log=HIGH</code>	<p>Defines the lowest priority audit events that will be logged.</p> <p>Accepted values are: HIGH, MEDIUM, LOW and NONE.</p> <p>Setting the value to HIGH will result in only HIGH level events being logged. NONE will cause no events to be logged. MEDIUM will only allow events with a priority of MEDIUM and HIGH to be logged.</p> <p>Refer to the levels for the various events.</p> <p>This does not affect events displayed in the Audit log screens for projects and repositories.</p>
<code>audit.details.max.length=1024</code>	<p>Defines the number of characters that can be stored as details for a single audit entry.</p>
<code>plugin.stash-audit.max.entity.rows=500</code>	<p>The maximum number of entries a project or repository can have in the audit tables.</p> <p>This does not affect the data stored in the logs.</p>

<code>plugin.stash-audit.cleanup.batch.size=1000</code>	<p>When trimming the audit entries table this is the maximum number of rows that will be trimmed in one transaction. Reduce this size if you are having issues with long running transactions.</p> <p>This does not affect the data stored in the logs.</p>
<code>plugin.stash-audit.cleanup.run.interval=24</code>	<p>How often the audit tables will be checked to see if they need to be trimmed (in hours).</p> <p>This does not affect the data stored in the logs.</p>

Authentication

Property	Description
<code>plugin.auth-crowd.sso.enabled=false</code>	Whether SSO support should be enabled or not. Regardless of this setting SSO authentication will only be activated when a Crowd directory is configured in Stash that is configured for SSO.
<code>plugin.auth-crowd.sso.session.validationinterval=3</code>	The number of minutes to cache authentication validation in the session. If this value is set to 0, each HTTP request will be authenticated with the Crowd server.
<code>plugin.auth-crowd.sso.session.lastvalidation=atl.crowd.sso.lastvalidation</code>	The session key to use when storing a Date value of the user's last authentication.
<code>plugin.auth-crowd.sso.session.tokenkey=atl.crowd.sso.tokenkey</code>	The session key to use when storing a String value of the user's authentication token.

Avatars

Property	Description
<code>avatar.gravatar.default=mm</code>	<p>The fallback URL for Gravatar avatars when a user does not have an acceptable avatar configured. This may be a URL resource, or a Gravatar provided default set.</p> <p>This configuration setting is DEPRECATED. It will be removed in Stash 3.0. Use <code>avatar.url.default</code> instead.</p>

<code>avatar.max.dimension=1024</code>	<p>Controls the max height <i>and</i> width for an avatar image. Even if the avatar is within the acceptable file size, if its dimensions exceed this value for height or width, it will be rejected.</p> <p>When an avatar is loaded by the server for processing, images with large dimensions may expand from as small as a few kilobytes on disk to consume a substantially larger amount of memory, depending on how well the image data was compressed. Increasing this limit can <i>substantially</i> increase the amount of heap used while processing avatars and may result in OutOfMemoryErrors.</p> <p>Value is in PIXELS.</p>
<code>avatar.max.size=1048576</code>	<p>Controls how large an avatar is allowed to be. Avatars larger than this are rejected and cannot be uploaded to the server, to prevent excessive disk usage.</p> <p>Value is in BYTES.</p>
<code>avatar temporary.cleanup.interval=1800000</code>	<p>Controls how frequently temporary avatars are cleaned up. Any temporary avatars that have been uploaded are checked against their configured max age and removed from the file system if they are "too old".</p> <p>Value is in MILLISECONDS.</p>
<code>avatar temporary.max.age=30</code>	<p>Controls how long a temporary avatar that has been uploaded is retained before it is automatically deleted.</p> <p>Value is in MINUTES.</p>
<code>avatar.url.default=\${avatar.gravatar.default}</code>	<p>Defines the fallback URL to be formatted into the <code>avatar.url.format.http</code> or <code>avatar.url.format.https</code> URL format for use when a user does not have an acceptable avatar configured. This value may be a URL or, if using Gravatar, it may be the identifier for one of Gravatar's default avatars.</p> <p>The default here falls back on the now-deprecated <code>avatar.gravatar.default</code> setting, which should ensure that value, if set, continues to work until it is removed in Stash 3.0. At that time, this default will become "mm".</p>
<code>avatar.url.format.http=http://www.gravatar.com/ avatar/%1\$s.jpg?s=%2\$d&d=%3\$s</code>	<p>Defines the default URL format for retrieving user avatars over HTTP. This default uses any G-rated avatar provided by the Gravatar service [http://www.gravatar.com]</p> <p>The following format parameters are available:</p> <ul style="list-style-type: none"> <code>%1\$s</code> – The user's e-mail address, MD5 hashed, or "00000000000000000000000000000000" if the user has no e-mail <code>%2\$d</code> – The requested avatar size <code>%3\$s</code> – The fallback URL, URL-encoded, which may be defined using "avatar.url.default" <code>%4\$s</code> – The user's e-mail address, not hashed, or an empty string if the user has no e-mail.

```
avatar.url.format.https=https://secure.
gravatar.com/
avatar/%1$s.jpg?s=%2$d&d=%3$s
```

Defines the default URL format for retrieving user avatars over HTTPS. This default uses any G-rated avatar provided by the Gravatar service [<http://www.gravatar.com>]

The following format parameters are available:

- %1\$s – The user's e-mail address, MD5 hashed, or "00000000000000000000000000000000" if the user has no e-mail
- %2\$d – The requested avatar size
- %3\$s – The fallback URL, URL-encoded, which may be defined using "avatar.url.default"
- %4\$s – The user's e-mail address, not hashed, or an empty string if the user has no e-mail.

Changesets

Property	Description
<code>changeset.diff.context=10</code>	Defines the number of context lines to include around diff segments in changeset diffs.

Changeset indexing

These properties control how changesets are indexed when new commits are pushed to Stash.

Property	Description
<code>indexing.max.threads=2</code>	Controls the maximum number of threads which are used to perform indexing. The resource limits configured below are not applied to these threads, so using a high number may negatively impact server performance.
<code>indexing.job.batch.size=250</code>	Defines the number of changesets which will be indexed in a single database transaction.
<code>indexing.job.queue.size=150</code>	Defines the maximum number of pending indexing requests. When this limit is reached, attempts to queue another indexing operation will be rejected.
<code>indexing.process.timeout.execution=3600</code>	Controls how long indexing processes are allowed to execute before they are interrupted, even if they are producing output or consuming input. The value is in SECONDS

Database

Database properties allow very specific configuration for your database connection parameters, which are set by Stash during database setup and migration, and allow you to configure a database of your own. We don't expect that you will edit these, except in collaboration with Atlassian Support.

Any other driver must be placed in `WEB-INF/lib` in order to use the associated database.

Warning: `jdbc.driver` and `jdbc.url` are available to plugins via the `ApplicationPropertiesService`. Some JDBC drivers allow the username and password to be defined in the URL. Because that property is available throughout the system (and will be included in STP support requests), that approach should not be used. The `jdbc.username` and `jdbc.password` properties should be used for these values instead.

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

Property	Description
<code>jdbc.driver=org.hsqldb.jdbcDriver</code>	<p>The JDBC driver class that should be used by Stash to connect to the database.</p> <p>The default Stash database uses the file-based HSQL database, storing its data in the <code><Stash home directory></code>. Stash currently bundles the following JDBC drivers:</p> <ul style="list-style-type: none"> <code>com.mysql.jdbc.Driver</code> <code>org.hsqldb.jdbcDriver</code> <code>org.postgresql.Driver</code>
<code>jdbc.url=jdbc:hsqldb:\${stash.home}/data/db;shutdown=true</code>	This is the JDBC url that Stash will use to connect to the database. This should include the driver subprotocol (e.g., <code>postgresql:</code>), 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.
<code>jdbc.user=SA</code>	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 <code>jdbc.url</code> .
<code>jdbc.password=</code>	The password that the user defined by <code>jdbc.user</code> will connect with.
<code>jdbc.ignoreunsupported=false</code>	Allows using a given database, even though it is marked as <code>UNSUPPORTED</code> . This is not intended to be broadly documented, nor to be used generally. It is here as a support mechanism to override the supported database check in the event that it incorrectly blocks access to a database.


Database pool

These properties control the database pool. The pool implementation used is BoneCP. Documentation for these settings can be found at: <http://jolbox.com/configuration.html>

To get a feel for how these settings really work in practice, the most relevant classes in BoneCP are:

- `com.jolbox.bonecp.BoneCP` - Creates the partitions, opens initial connections, starts threads
- `com.jolbox.bonecp.BoneCPDataSource` - Manages the pool, bridge to the `DataSource` interface
- `com.jolbox.bonecp.PoolWatchThread` - Handles the connection threshold

Property	Description
<code>db.pool.acquireIncrement=2</code>	Defines the number of connections to open in a batch when open connections are almost exhausted for a given partition.

<code>db.pool.cache.statements=100</code>	<p>Defines the number of statements to cache.</p> <p>If you configure your JDBC driver to use loadbalancing or failover with the Stash server, you may need to set <code>db.pool.cache.statements=0</code>. Please note that Stash does not yet (as of version 2.8) support DB redundancy, and that this configuration change is not supported by Atlassian.</p>
<code>db.pool.connection.timeout=15</code>	<p>Defines the amount of time the system will wait when attempting to open a new connection before throwing an exception. The system may hang, during startup, for the configured number of seconds if the database is unavailable. As a result, the timeout configured here should not be generous.</p> <p>Value is in SECONDS.</p>
<code>db.pool.idle.maxAge=30</code>	<p>Defines the maximum period of time a connection may be idle before it is closed. Generous values should be used here to prevent creating and destroying many short-lived database connections (which defeats the purpose of pooling).</p> <p>Value is in MINUTES.</p>
<code>db.pool.idle.testInterval=10</code>	<p>Defines the amount of time a connection may be idle before a test query is executed by the pool. This helps prevent connections from being closed by the database server due to inactivity.</p> <p>Value is in MINUTES.</p>
<code>db.pool.partition.connection.maximum=20</code>	<p>Defines the maximum number of connections that may be open in a given partition.</p>
<code>db.pool.partition.connection.minimum=4</code>	<p>Defines the minimum number of connections open for a given partition. Each partition will open this many connections on startup. That means <code>db.pool.partition.connection.minimum x db.pool.partition.count = initial connection count</code>.</p>
<code>db.pool.partition.connection.threshold=10</code>	<p>Defines the threshold as a <i>percentage of the maximum connections</i> that each partition will attempt to keep available at all times. If the number of available connections drops to or below the threshold, <code>acquireIncrement</code> connections will be opened until the partition is above it again.</p> <p> Warning: Be careful to take this number into account when setting the minimum and maximum counts. For example, if the maximum is 30 and the minimum is 5 and the threshold is 20 (20%), 5 is not 20% of 30, so immediately after it is created the partition will open additional connections to get above the threshold. Effectively, that would mean that the "minimum" per partition is 10 (5 + 5 <code>acquireIncrement</code>), or 40 connections at all times.</p>

<code>db.pool.partition.count=4</code>	Defines the number of different connection partitions to use. This value is used to decrease lock contention, because each partition locks individually. The recommended setting is 3 or 4, but in servers with heavy load and many short-lived requests, performance may be improved by using a higher value.
<code>db.pool.threads=4</code>	Defines the number of helper threads which will be used by the pool to cleanup and release connections back into the pool. Setting a value of 0 disables this feature, which means the executing thread will perform cleanup and release itself. A non-zero value results in a pool of helpers which process connections out of a holding queue. When a thread "closes" a connection, that thread is allowed to continue executing and the connection is placed in the queue. One of the helper threads then performs final cleanup to prepare the connection to be returned to the pool.

Downloads

Property	Description
<code>http.download.raw.policy=Smart</code>	<p>Controls the download policy for raw content.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> <code>Insecure</code> – Allows all file types to be viewed in the browser <code>Secure</code> – Requires all file types to be downloaded rather than viewed in the browser <code>Smart</code> – Forces "dangerous" file types to be downloaded, rather than allowing them to be viewed in the browser <p>These options are case-sensitive and defined in <code>com.atlassian.http.mime.DownloadPolicy</code></p>

Events

These properties control the number of threads that are used for dispatching asynchronous events. Setting this number too high can decrease overall throughput when the system is under high load because of the additional overhead of context switching. Configuring too few threads for event dispatching can lead to events being queued up, thereby reducing throughput. These defaults scale the number of dispatcher threads with the number of available cpu cores.

Property	Description
<code>event.dispatcher.core.threads=0.8*cpu</code>	The minimum number of threads that is available to the event dispatcher. The <code>cpu</code> variable is resolved to the number of cpus that are available.

<code>event.dispatcher.max.threads=cpu</code>	The maximum number of event dispatcher threads. The number of dispatcher threads will only be increased when the event queue is full and <code>max.threads</code> has not been reached yet.
<code>event.dispatcher.queue.size=4096</code>	The number of events that can be queued. When the queue is full and no more threads can be created to handle the events, events will be discarded.
<code>event.dispatcher.keepAlive=60</code>	The time a dispatcher thread will be kept alive when the queue is empty and more than <code>core.threads</code> threads are running. Value is in SECONDS.

Executor


Controls the thread pool that is made available to plugins for asynchronous processing.

Property	Description
<code>executor.max.threads=100</code>	Specifies the maximum number of threads in the thread pool. When more threads are required than the configured maximum, the thread attempting to schedule an asynchronous task to be executed will block until a thread in the pool becomes available.
<code>executor.keepAliveTime=60</code>	Controls how long idle threads are kept alive. Threads idle for more than this time will be terminated. Value is in SECONDS and must be ≥ 1 . If 0 or a negative value is used, a default value of 1 will be configured.

Features

Feature properties control high-level system features, allowing them to be disabled for the entire instance. Features that are disabled at this level are disabled *completely*. This means that instance-level configuration for a feature is overridden. It also means that a user's permissions are irrelevant; a feature is still disabled even if the user has the `SYS_ADMIN` permission.

Property	Description
----------	-------------

<code>feature.auth.captcha=true</code>	<p>Controls whether to require CAPTCHA verification when the number of failed logins is exceeded. If enabled, any client who has exceeded the number of failed logins allowed using either the Stash web interface or the Git hosting interface will be required to authenticate in the Stash web interface and successfully submit a CAPTCHA before continuing. Setting this to false will remove this restriction and allow users to incorrectly authenticate as many times as they like without penalty.</p> <p> Warning: It is STRONGLY recommended you keep this setting enabled. Disabling it will have the following ramifications:</p> <ul style="list-style-type: none"> • Your users may lock themselves out of any underlying user directory service (LDAP, Active Directory etc) because Stash will pass through all authentication requests (regardless of the number of previous failures) to the underlying directory service. • For Stash installations where you use Stash for user management or where you use a directory service with no limit on the number of failed logins before locking out users, you will open Stash or the directory service up to brute-force password attacks.
<code>feature.forks=true</code>	<p>Controls whether repositories can be forked. This setting <i>supersedes and overrides</i> instance-level configuration.</p> <p>If this is set to false, even repositories which are marked as forkable cannot be forked.</p>
<code>feature.public.access=true</code>	<p>Public access to Stash allows unauthenticated users to be granted access to projects and repositories for specific read operations including cloning and browsing repositories. This is normally controlled by project and repository administrators but can be switched off system wide by setting this property to false. This can be useful in highly sensitive environments.</p>

Hibernate

Property	Description
<code>hibernate.format_sql=false</code>	When <code>hibernate.show_sql</code> is enabled, this flag controls whether Hibernate will format the output SQL to make it easier to read.
<code>hibernate.jdbc.batch_size=20</code>	Controls Hibernate's JDBC batching limit, which is used to make bulk processing more efficient (both for processing and for memory usage).

<code>hibernate.show_sql=false</code>	Used to enable Hibernate SQL logging, which may be useful in debugging database issues. This value should generally only be set by developers, not by customers.
---------------------------------------	--

JIRA

Property	Description
<code>plugin.jira-integration.pullrequest.attribute.changesets.max=100</code>	Controls the maximum number of changesets to retrieve when retrieving attributes associated with changesets of a pull-request. This value should be between 50 and 1000 as Stash will enforce a lower bound of 50 issues and an upper bound of 1000 issues.
<code>plugin.jira-integration.remote.page.max.issues=20</code>	Controls the maximum number of issues to request from JIRA. This value should be between 5 and 50 as Stash will enforce a lower bound of 5 issues and an upper bound of 50 issues.
<code>plugin.jira-integration.remote.timeout.connection=5000</code>	The connection timeout duration in milliseconds for requests to JIRA. This timeout occurs if the JIRA server does not answer. e.g. the server has been shut down. This value should be between 2000 and 60000 as Stash will enforce a lower bound of 2000ms and an upper bound of 60000ms. Value is in MILLISECONDS .
<code>plugin.jira-integration.remote.timeout.socket=10000</code>	The socket timeout duration in milliseconds for requests to JIRA. This timeout occurs if the connection to JIRA has been stalled or broken. This value should be between 2000 and 60000 as Stash will enforce a lower bound of 2000ms and an upper bound of 60000ms Value is in MILLISECONDS .

Liquibase

Property	Description
<code>liquibase.commit.block.size=10000</code>	The maximum number of changes executed against a particular Liquibase database before a commit operation is performed. Very large values may cause DBMS to use excessive amounts of memory when operating within transaction boundaries. If the value of this property is less than one, then changes will not be committed until the end of the change set.

Logging

Logging levels for any number of loggers can be set in the `stash-config.properties` file using the following

format:

```
logging.logger.<name>=<level>
```

For example, to configure all classes in the `com.atlassian.stash` package to `DEBUG` level:

```
logging.logger.com.atlassian.stash=DEBUG
```

To adjust the `ROOT` logger, you use the special name `ROOT` (case-sensitive):

```
logging.logger.ROOT=INFO
```

Notifications

Property	Description
<code>plugin.stash-notification.mail.max.comment.size=2048</code>	Controls the maximum allowed size of a single comment in characters (not bytes). Extra characters will be truncated.
<code>plugin.stash-notification.mail.max.description.size=2048</code>	Controls the maximum allowed size of a single description in characters (not bytes). Extra characters will be truncated.
<code>plugin.stash-notification.mentions.enabled=true</code>	Controls whether mentions are enabled
<code>plugin.stash-notification.max.mentions=200</code>	Controls the maximum number of allowed mentions in a single comment

Paging

These properties control the maximum number of objects which may be returned on a page, regardless of how many were actually requested by the user. For example, if a user requests `Integer.MAX_INT` branches on a page, their request will be limited to the value set for `page.max.branches`.

This is intended as a safeguard to prevent enormous requests from tying up the server for extended periods of time and then generating responses whose payload is prohibitively large. The defaults configured here represent a sane baseline, but may be overridden by customers if necessary.

Property	Description
<code>page.max.branches=1000</code>	Maximum number of branches per page.
<code>page.max.changes=1000</code>	Maximum number of changes per page. Unlike other page limits, this is a hard limit; subsequent pages cannot be requested when the number of changes in a changeset exceeds this size.

<code>page.max.changesets=100</code>	Maximum number of changesets (commits) per page.
<code>page.max.diff.lines=10000</code>	Maximum number of segment lines (of any type, total) which may be returned for a single diff. Unlike other page limits, this is a hard limit; subsequent pages cannot be requested when a diff exceeds this size.
<code>page.max.directory.children=500</code>	Maximum number of directory entries which may be returned for a given directory.
<code>page.max.directory.recursive.children=100000</code>	Maximum number of file entries which may be returned for a recursive listing of a directory. A relatively high number as this is used by the file finder which needs to load the tree of files upfront.
<code>page.max.groups=1000</code>	Maximum number of groups per page.
<code>page.max.index.results=50</code>	Maximum number of changesets which may be returned from the index when querying by an indexed attribute. For example, this limits the number of changesets which may be returned when looking up commits against a JIRA issue.
<code>page.max.projects=1000</code>	Maximum number of projects per page.
<code>page.max.repositories=1000</code>	Maximum number of repositories per page.
<code>page.max.source.length=5000</code>	Maximum length for any line returned from a given file when viewing source. This value truncates long lines. There is no mechanism for retrieving the truncated part short of downloading the entire file.
<code>page.max.source.lines=5000</code>	Maximum number of lines which may be returned from a given file when viewing source. This value breaks large files into multiple pages.
<code>page.max.tags=1000</code>	Maximum number of tags per page.
<code>page.max.users=1000</code>	Maximum number of users per page.
<code>page.scan.pullrequest.activity.size=500</code>	The size of the page Stash should use when scanning activities.
<code>page.scan.pullrequest.activity.count=4</code>	The number of pages of activities Stash should scan before giving up.

Password reset

Property	Description
<code>password.reset.validity.period=4320</code>	Controls how long a password reset token remains valid for. Default period is 72 hours. Value is in MINUTES.

Process execution

Property	Description
----------	-------------

<pre>process.timeout.execution=120 process.timeout.idle=60</pre>	<p>Controls timeouts for external processes, such as Git and Hg. The idle timeout configures how long the command is allowed to run without producing any output. The execution timeout configures a hard upper limit on how long the command is allowed to run even if it is producing output.</p> <p>Values are in SECONDS. Using 0, or a negative value, disables the timeout completely.</p> <p>USE AT YOUR OWN RISK!</p>
--	---

Pull requests

Property	Description
<code>plugin.stash-scm-git.pullrequest.merge.strategy.KEY.slug=no-ff</code>	<p>Control the merge strategy for a repository (where <code>KEY</code> is the project key and <code>slug</code> is the repository slug). Note that the URL for the browse page of a repository is of the following form: <a href="http://<stashdomain>/projects/<PROJECTKEY>/repos/<reposlug>/browse">http://<stashdomain>/projects/<PROJECTKEY>/repos/<reposlug>/browse</p> <p>Overrides project and global settings.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> <code>no-ff</code> – no fast-forward; the default setting. <code>ff</code> – allow fast-forward; will merge when necessary. <code>ff-only</code> – require fast-forward; will never create merge commits; fail if a merge is required. <code>squash</code> – collapse all incoming commits into a single commit directly to the target branch; never create a merge. <code>squash-ff-only</code> – collapse all the incoming commits into a single commit directly to the target branch, never creating a merge, but do so <i>only</i> if the source branch is fast-forward.
<code>plugin.stash-scm-git.pullrequest.merge.strategy.KEY=no-ff</code>	<p>Control the merge strategy for a project (where <code>KEY</code> is the project key).</p> <p>Overrides global settings. Is overridden by repository settings.</p> <p>Possible values are listed above.</p>
<code>plugin.stash-scm-git.pullrequest.merge.strategy=no-ff</code>	<p>Control the merge strategy globally. Is overridden by repository and project settings.</p> <p>Possible values are listed above.</p>
<code>pullrequest.diff.context=10</code>	<p>Defines the number of context lines to include around diff segments in pull request diffs. By default, Git only includes 3 lines. The default is 10, to try and include a bit more useful context around changes, until the ability to "expand" the context is implemented.</p>

<code>pullrequest.rescope.changesets.display=5</code>	Defines the maximum number of changesets per type (either added or removed) to display in a rescope activity.
<code>pullrequest.rescope.changesets.max=1000</code>	Defines the absolute maximum number of changesets that will be evaluated when attempting to determine, for a given rescope activity, which changesets were added to or removed from a pull request. Adjusting this setting can have significant memory footprint impact on the system. It is not recommended to be changed, but the option is provided here to support unique use cases.
<code>pullrequest.rescope.detail.threads=2</code>	Defines the maximum number of threads to use for precalculating rescope details. These threads perform the requisite processing to determine the commits added and removed when a pull request is rescope, where most rescopes do not add or remove any commits. Such "dead" rescopes are deleted during processing. The primary goal is to ensure all details have already been calculated when users try to view a pull request's overview.
<code>pullrequest.rescope.drift.threads=4</code>	Defines the maximum number of threads to use when processing comment drift for a pull request during rescope. Higher numbers here do <i>not</i> necessarily mean higher throughput! Performing comment drift will usually force a new merge to be created, which can be very I/O intensive. Having a substantial number of merges running at the same time can significantly <i>reduce</i> the speed of performing comment drift.

Readme parsing

Property	Description
<code>plugin.stash-readme.max.size=65536</code>	Controls the maximum allowed size of a readme file to parse. Value is in BYTES.

Ref metadata

Property	Description
<code>ref.metadata.timeout=2</code>	Controls timeouts for retrieving metadata associated with a collection of refs from all metadata providers collectively. This values is in SECONDS.

Resource throttling

These properties define concurrent task limits for the ThrottleService, limiting the number of concurrent Git operations of a given type that may be run at once. This is intended to help prevent Stash from overwhelming a server machine with

running processes. 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).

When the limit is 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 requests while accessing the Stash UI are rejected, users will see either a 501 error page indicating the server is under load, or a popup indicating part of the current page failed.

When Git client 'hosting' commands (pull/push/clone) are rejected, Stash does a number of things:

- Stash will return an error message to the client which the user will see on the command line: "Stash is currently under heavy load and is not able to service your request. Please wait briefly and try your request again"
- 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.

The hard, machine-level limits these are intended to prevent hitting are very OS- and hardware-dependent, so you may need to tune them for your instance of Stash. When hyperthreading is enabled for the server CPU, for example, it is likely that the server will allow sufficient concurrent Git operations to completely bury the I/O on the machine. In such cases, we recommend starting off with a less aggressive default on multi-cored machines – the value can be increased later if hosting operations begin to back up. These defaults are finger-in-the-wind guesstimates (which so far have worked well).

Additional resource types may be configured by defining a key with the format `throttle.resource.<resource-name>`.

When adding new types, it is strongly recommended to configure their ticket counts explicitly using this approach.

Property	Description
<code>throttle.resource.scm-command=25</code>	Limits the number of operations that support the UI, such as <code>git diff</code> , <code>git blame</code> , or <code>git rev-list</code> , that can run concurrently. This is intended to prevent these SCM commands from competing with the running of push and pull operations.
<code>throttle.resource.scm-command.timeout=2</code>	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.
<code>throttle.resource.scm-hosting=1.5*cpu</code>	Limits the number of SCM 'hosting' operations, such as <code>git clone</code> , <code>git push</code> and <code>git pull</code> over HTTP or SSH that may be running concurrently. This is intended primarily to prevent pulls, which can be very memory-intensive, from pinning a server's resources. There is limited support for mathematical expressions; <code>+</code> , <code>-</code> , <code>*</code> , <code>\</code> and <code>()</code> are supported. You can also use the <code>cpu</code> variable which is resolved to the number of cpus that are available.
<code>throttle.resource.scm-hosting.timeout=300</code>	Controls how long threads will wait for SCM hosting operations to complete when the system is already running the maximum number of SCM hosting operations. Value is in SECONDS.

<code>throttle.resource.busy.message.timeout=5</code>	<p>Controls how long a warning banner is displayed in the UI after a request is rejected due to excessive load.</p> <p>Value is in MINUTES. Using 0, or a negative value, disables displaying the banner.</p> <p>This is deprecated and replaced by <code>server.busy.on.ticket.rejected.within</code>, It is due to be removed in Stash 3.0.</p>
---	---

SCM – Cache

See [Scaling Stash for Continuous Integration performance](#) for more information about using the SCM Cache Plugin for Stash.

Property	Description
<code>plugin.stash-scm-cache.expiry.check.interval=300</code>	<p>Controls how frequently expired caches are checked and deleted from disk.</p> <p>Value is in SECONDS.</p>
<code>plugin.stash-scm-cache.minimum.free.space=1073741824</code>	<p>Controls how much space needs to be available on disk (specifically under <STASH-HOME>/caches) for caching to be enabled. This setting ensures that the cache plugin does not fill up the disk.</p> <p>Value is in BYTES.</p>
<code>plugin.stash-scm-cache.protocols=HTTP,SSH</code>	Controls which protocols caching is applied to.
<code>plugin.stash-scm-cache.refs.enabled=false</code>	Controls whether ref advertisement operations are cached.
<code>plugin.stash-scm-cache.refs.ttl=60</code>	<p>Controls how long the caches for ref advertisements are kept around when there no changes to the repository.</p> <p>Caches are automatically invalidated when someone pushes to a repository or when a pull request is merged.</p> <p>Time is in SECONDS.</p>
<code>plugin.stash-scm-cache.upload-pack.enabled=true</code>	Controls whether clone operations are cached.
<code>plugin.stash-scm-cache.upload-pack.ttl=14400</code>	<p>Controls how long the caches for clone operations are kept around when there no changes to the repository.</p> <p>Caches are automatically invalidated when someone pushes to a repository or when a pull request is merged.</p> <p>Time is in SECONDS.</p>

SCM – Git

Property	Description
<code>plugin.stash-scm-git.path.executable=git</code>	<p>Defines the default path to the Git executable. On Windows machines, the .exe suffix will be added to the configured value automatically if it is not present. In general, "git" should be an acceptable default for every platform, here, assuming that it will be available in the runtime user's PATH.</p> <p>With the new path searching performed by <code>DefaultGitBinaryHelper</code>, setting a default value here is unnecessary, as the plugin will quickly discard the value. This is left here purely for documenting how to set an explicit path.</p>
<code>plugin.stash-scm-git.path.libexec=</code>	<p>Defines the path to the Git libexec directory (containing the git-core directory). This path is hard-coded into the Git executable and is used for forking processes like git-http-backend. If this value is set, Stash will directly fork out those processes. This eliminates an unnecessary fork (git -> git-http-backend) and may improve scalability.</p>
<code>plugin.stash-scm-git.backend.http.buffer.size=32768</code>	<p>Defines the buffer size in bytes which is used when marshaling data between the Git process and the HTTP socket.</p>
<code>plugin.stash-scm-git.backend.ssh.buffer.size=32768</code>	<p>Defines the buffer size in bytes which is used when marshaling data between the Git process and the SSH socket.</p>
<code>plugin.stash-scm-git.backend.timeout.idle=1800</code>	<p>Defines the idle timeout for push/pull processes, applying a limit to how long the operation is allowed to execute without either producing output or consuming input. The default value is 30 minutes.</p> <p>This value is in SECONDS.</p>
<code>plugin.stash-scm-git.backend.timeout.execution=86400</code>	<p>Defines the execution timeout for push/pull processes, applying a hard limit to how long the operation is allowed to run even if it is producing output or reading input. The default value is 1 day.</p> <p>This value is in SECONDS.</p>

<pre>plugin.stash-scm-git.diff.renames=copies</pre>	<p>Defines whether copy and/or rename detection should be performed. By default, both rename <i>and</i> copy detection are performed. Only files modified in the same commit are considered as rename or copy origins, to minimize overhead.</p> <p>The possible settings are:</p> <ul style="list-style-type: none"> • <code>copy</code> or <code>copies</code> – Applies <code>--find-copies</code> • <code>rename</code> or <code>renames</code> – Applies <code>--find-renames</code> • <code>off</code> – Disables rename <i>and</i> copy detection <p>When using <code>copy</code> or <code>copies</code>, the value may optionally be suffixed with a "+" to use <code>--find-copies-harder</code>. This setting should be used with caution, as it can be very expensive. It considers every file in the repository, even files not modified in the same commit, as possible origins for copies.</p> <p>When copy and/or rename detection is enabled <code>plugin.stash-scm-git.diff.renames.threshold</code> can be used control the similarity index required for a change to be identified as a copy or rename.</p>
<pre>plugin.stash-scm-git.diff.renames.threshold=50</pre>	<p>Defines the threshold, as a percentage, for a file to be detected as a rename or a copy. This setting is only applied if copy and/or rename detection is enabled. The default threshold applied is 50% similarity (defined in Git itself).</p> <p>Git diff and Git diff-tree <i>do not honor</i> 100 (identical files only) for the threshold. They ignore the threshold and apply the default 50% threshold instead. A configured threshold of 100 will be applied as 99. Similarly, a configured threshold that is 0, or negative, will be applied as 1.</p>
<pre>plugin.stash-scm-git.environment.variablesize=2000</pre>	<p>Defines the maximum number of characters that can be added to a single environment variable. Different operating systems (and even different versions of the same operating system) have different hard limitations they apply to environment variables. This default is intended to be low enough to work on all supported platforms out of the box, but still high enough to be usable. It is configurable in case it proves to be too high on some platform.</p>
<pre>plugin.stash-scm-git.pullrequest.merge.auto.forceadd=false</pre>	<p>Defines whether conflicted files should be added to the index using Git <code>add --force</code>, during automatic merges. By default, this behaviour is <i>off</i> – simple Git <code>add</code> is "safer". However, when merging across branches with discrepant <code>.Gitignore</code> settings, enabling this setting may allow the system to create a conflicted diff (where without it a diff to the common ancestor will be shown instead).</p> <p>Note: This value has <i>no effect</i> on real pull request merges. It is <i>only</i> applied during automatic merges, for producing a pull request's change tree and diff.</p>

<code>plugin.stash-scm-git.pullrequest.merge. auto.timeout=120</code>	Defines the maximum amount of time any command used to perform a merge to support the "merge" diff mode is allowed to execute <i>or</i> idle. Because the commands used generally do not produce output, there is no separate idle timeout. This value is in SECONDS.
<code>plugin.stash-scm-git.pullrequest.merge. real.timeout=300</code>	Defines the maximum amount of time any command used to merge a pull request is allowed to execute <i>or</i> idle. Because the commands used generally do not produce output, there is no separate idle timeout. This value is in SECONDS.

Server busy banners

Property	Description
<code>server.busy.on.ticket.rejected.within=5</code>	Controls how long a warning banner is displayed in the UI after a request is rejected due to excessive load. Value is in MINUTES. Using 0, or a negative value, disables displaying the banner.
<code>server.busy.on.queue.time=60</code>	Controls how long requests need to be queued before they cause a warning banner to appear. Value is in SECONDS. Using 0, or a negative value, disables displaying the banner.

SMTP

Property	Description
<code>mail.timeout.connect=60</code> <code>mail.timeout.send=60</code> <code>mail.test.timeout.connect=30</code> <code>mail.test.timeout.send=30</code>	Controls timeouts for establishing an SMTP connection and sending an e-mail. Shorter timeouts should be applied for when sending test e-mails, as the test occurs in user time. Values are in SECONDS.
<code>mail.error.pause.log=300</code>	Controls how frequently logs will go to the standard log file about mail sending errors. All errors are logged to the <code>atlassian-stash-mail.log</code> file, but Stash will periodically log a warning to the standard log file if there are errors sending messages. Value is in SECONDS

<code>mail.error.pause.retry=5</code>	Controls how long Stash will wait before retrying to send a message if an error occurs. Value is in SECONDS
<code>mail.threads=1</code>	Controls the number of threads to use for sending emails. Setting this to a higher value will put greater load on your mail server when Stash generates a lot of emails, but will make Stash clear its internal queue faster.
<code>mail.max.message.size=1048576</code>	Controls the maximum allowed size of a single mail message, which is the sum of the subject and body sizes. Value is in BYTES.
<code>mail.max.queue.size=157286400</code>	Controls the maximum allowed size for the mail queue (any new message will be rejected if the mail queue reaches that size). Value is in BYTES.

SSH command execution

Property	Description
<code>plugin.ssh.command.timeout.idle=86400</code>	Controls timeouts for all SSH commands, such as those that service Git and hg operations over SSH. The idle timeout configures how long the command is allowed to run without writing any output to the client. For SCM commands, the <code>plugin.*.backend.timeout.idle</code> properties defined above will be applied to the underlying process. The default value is 1 day. The value is in SECONDS.

Syntax highlighting

See [Configuring syntax highlighting for file extensions](#) for more information.

Stash only applies syntax highlighting to source files, not to diffs.

Property	Description
<code>syntax.highlighter.<language>.executable=exe1,exe2</code>	Controls the language highlighter used for a given set of hashbang executables. The <code><language></code> refers to the highlighter key defined by <code>highlight.js</code> .
<code>syntax.highlighter.<language>.extensions=ext1,ext2</code>	Controls the language highlighter used for a given set of file extensions. The <code><language></code> refers to the highlighter key defined by <code>highlight.js</code> .

Enabling SSH access to Git repositories in Stash


A Stash administrator can enable SSH access to Git repositories in Stash. This 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 their Stash account to be able to use SSH access to repositories.

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](#)
- [Creating SSH keys](#)

Enabling SSH access


To enable SSH access:

1. Go to the Stash administration area and click **Server settings** (under 'Settings').
2. Under 'SSH access', check **SSH enabled**.
3. Enter values for **SSH port** and **SSH base URL**, according the information in the sections below.
4. Click **Save**.

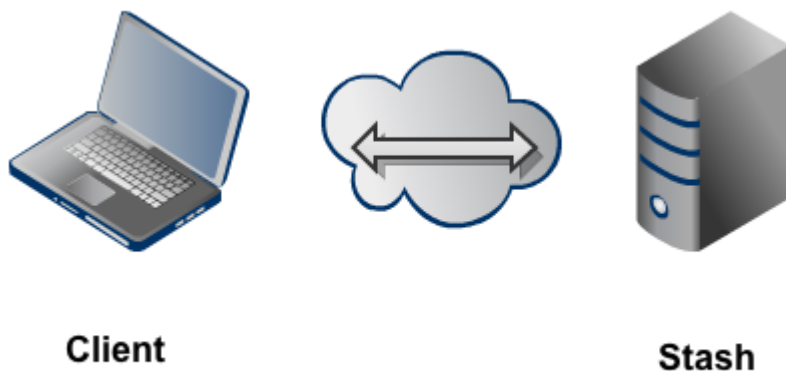
SSH base URL



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. See [Specifying the base URL for Stash](#).

 For example, if the **SSH base URL** is not set and the **Stash base URL** is `https://stash.atlassian.com` and the SSH port is 7999, the SSH URL for the repository `Jira` in the project `Atlassian` will be `ssh://git@stash.atlassian.com:7999/ATLASSIAN/jira.git`


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.




Port forwarding	SSH base URL	Stash base URL	SSH port	Resulting SSH URL for a repo
	Not set	<code>https://stash.atlassian.com</code>	7999	<code>ssh://git@stash.atlassian.com:7999/<projectname>/<reponame>.git</code>
 Port 22 -> 7999	<code>https://stash.atlassian.com</code>	<code>https://stash.atlassian.com</code>	7999	<code>ssh://git@stash.atlassian.com/<projectname>/<reponame>.git</code>

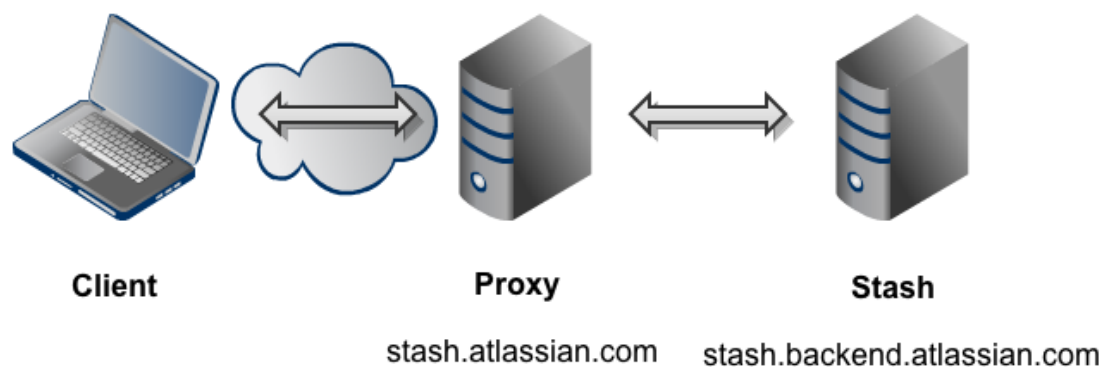
When running Stash behind a proxy

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).

 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`



Port forwarding	SSH base URL	SSH port	Stash base URL	Resulting SSH URL for a repo
✗	<code>ssh://stash.backend.atlassian.com:7999</code>	7999	<code>https://stash.backend.atlassian.com</code>	<code>ssh://git@stash.backend.atlassian.com:7999/<projectname>/<reponame>.git</code>
✓ Port 22 -> 7999	<code>ssh://stash.atlassian.com</code>	7999	<code>https://stash.backend.atlassian.com</code>	<code>ssh://git@stash.atlassian.com/<projectname>/<reponame>.git</code>
✓ Port 44 -> 7999	<code>ssh://stash.atlassian.com:44</code>	7999	<code>https://stash.backend.atlassian.com</code>	<code>ssh://git@stash.atlassian.com:44/<projectname>/<reponame>.git</code>

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.mycompany.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).

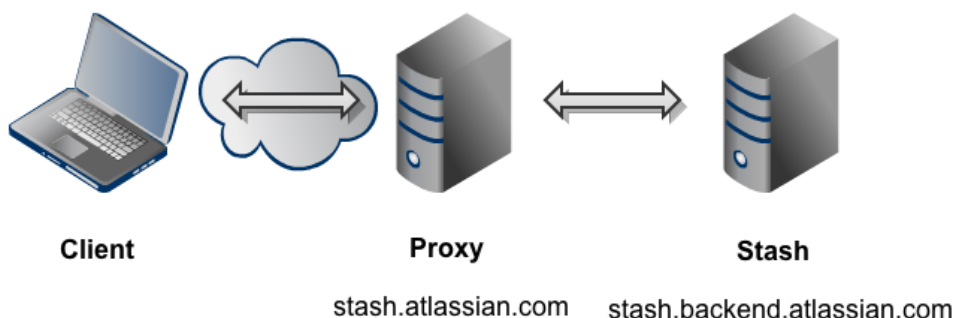
This would allow your users to use a URL without a port number in it, like this:

```
git clone ssh://git@stash.mycompany.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 HTTPS URLs consistent.



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 HTTPS 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.

i HAProxy is not an Atlassian product, so Atlassian does not guarantee to provide support for its configuration. This section is provided for your information only – use it at your own risk. We recommend that you refer to the [HAProxy documentation](#).

Installing HAProxy

Your Operating System may support installing HAProxy via it's 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 HTTPS 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

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 127.0.0.1 with the IP address of the machine running Stash.

You can check your configuration by running:

```
haproxy -f haproxyconf.txt -c
```

To run haproxy, simply start it using

```
haproxy -f haproxyconf.txt
```



If you use HAProxy to additionally proxy HTTP traffic, ensure that the running `mode` configuration is set to `http`:

```
backend http
    mode http
    bind *:80
    server localhost-stash-http 127.0.0.1:7990
```

Using the default SSH port

You can configure HAProxy 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 daemon is running on port 22. You have several options:

- Configure HAProxy 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 HAProxy to only listen on that interface.
- Move the default ssh daemon to listen on another port and let HAProxy bind on port 22.

We do not provide instructions on the last two options, except for how to configure HAProxy.

Use the same configuration as the last example, but change the bind port to 22, e.g.

```
...
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.

```
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](#).

Proxying and securing Stash

This page provides an overview of some common network topology options for running Stash, including running Stash behind a reverse proxy and securing access to Stash by using HTTPS (HTTP over SSL).

Note that Stash does not need to run behind a web server – it is capable of serving web requests directly using the bundled Tomcat application server. On this page, 'connecting to Stash' really means connecting to Tomcat, which is used to serve Stash content.

On this page:

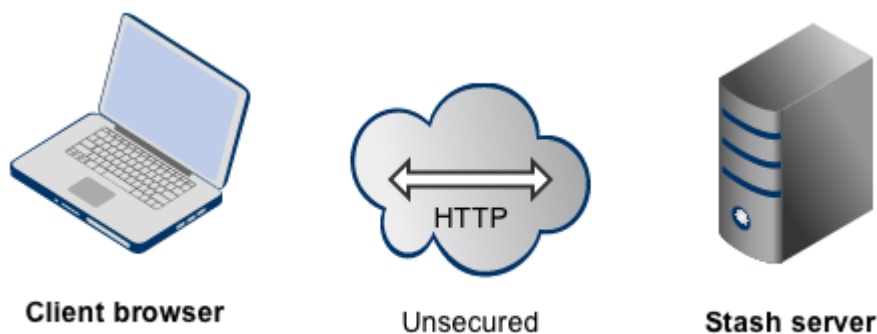
- [Connecting to Stash directly over HTTP](#)
- [Securing access to Stash using HTTPS](#)
- [Using a reverse proxy for Stash](#)
- [Securing a reverse proxy using HTTPS](#)

Connecting to Stash directly over HTTP

Connecting directly to Stash (that is, Tomcat) is the default install configuration, as described in the Stash install documentation:

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

When set up this way, the user accesses Stash directly over HTTP, without using SSL – all communication between the user's browser and Stash will be unsecured.



You may also wish to consider the following:

- Stash, by default, will listen for requests on port 7990 – this [port can be changed](#) if required.
- The address with which to access Stash, by default, will be `http://<computer name>:7990`. Change the [base URL for Stash](#) if required.
- You can [set the context path](#) for Stash if you are running another Atlassian application, or Java web application, at the same hostname and context path as Stash.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).

Securing access to Stash using HTTPS

Access to Stash can be secured by enabling HTTPS (HTTP over SSL) for the Tomcat application server that is bundled with Stash. You should consider doing this, and making secure access mandatory, if Stash will be internet-facing and usernames, passwords and other proprietary data may be at risk.

When set up in this way, access to Stash is direct, and all communication between the user's browser and Stash will be secured using SSL.

See [Securing Stash with Tomcat using SSL](#) for configuration details.



Note that:

- Stash will listen for requests on port 8443, according to the instructions in [Securing Stash with Tomcat using SSL](#). This port can be changed if required.
- The address with which to access Stash, by default, will be `https://<computer name>:8443`. Change the [base URL for Stash](#) if required.
- Any existing [links with other applications](#) will need to be reconfigured using this new URL for Stash.
- You can [set the context path](#) for Stash if you are running another Atlassian application, or Java web application, at the same hostname and context path as Stash.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).

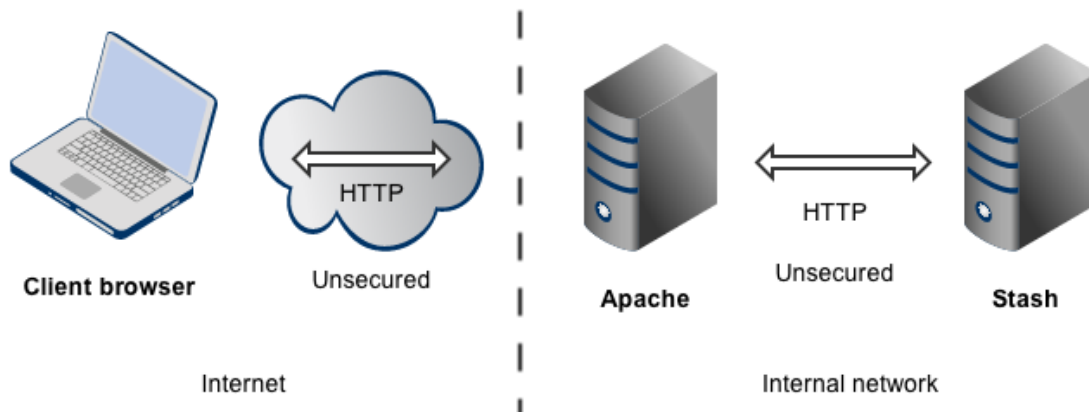
Using a reverse proxy for Stash

You can run Stash behind a reverse proxy, for example Apache HTTP Server. You may wish to do this if you want to:

- use a different port number to access Stash
- use a different context path to access Stash

When set up this way, external access to Stash is via a reverse proxy, without using SSL. All communication between the user's browser and Apache, and so Stash, will be unsecured, but users do not have direct access to Stash. An example scenario is where Apache provides a gateway through which users outside the firewall can access Stash.

See [Integrating Stash with Apache HTTP Server](#) for configuration details.



Note that:

- Stash, by default, will listen for requests on port 7990 – this port can be changed if required.
- Stash (Tomcat) needs to know the URL (proxy name) that Apache serves.
- The address with which to access Stash will be `http://<proxy name>:7990`. Change the [base URL for Stash](#) if required.
- Any existing [links with other applications](#) will need to be reconfigured using this new URL for Stash.
- You can [set the context path](#) for Stash if you are running another Atlassian application, or Java web application, at the same hostname and context path as Stash.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).

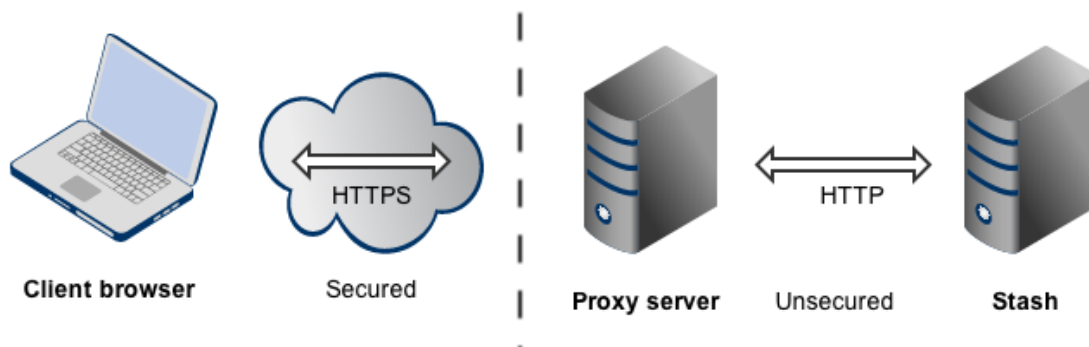
Securing a reverse proxy using HTTPS

You can run Stash behind a reverse proxy, such as Apache HTTP Server or nginx, that is secured using HTTPS (HTTP over SSL). You should consider doing this, and making secure access mandatory, if usernames, passwords and other proprietary data may be at risk. An example scenario is where Apache HTTP Server provides a gateway through which users outside the firewall can access Stash.

When set up in this way, external access to Stash is via a reverse proxy, where external communication with the proxy uses HTTPS. All communication between the user's browser and the reverse proxy will be secured, whereas communication between the proxy and Stash will not be secured (it doesn't use SSL).

See the following pages for configuration details:

- [Securing Stash with Apache using SSL](#)
- [Securing Stash behind nginx using SSL](#)



Note that:

- The reverse proxy (for example, Apache) will listen for requests on port 443.
- Stash, by default, will listen for requests on port 7990. Stash (Tomcat) needs to know the URL (proxy name) that the proxy serves.
- The address with which to access Stash will be `https://<proxyName>:<proxyPort>/<context path>`, for example `https://mycompany.com:443/stash`
- Any existing [links with other applications](#) will need to be reconfigured using this new URL for Stash.
- Stash (Tomcat) should be configured to refuse requests on port 7990 and to redirect those to the proxy on port 443.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).
- It would be possible to set up an SSL connection between the proxy server and Tomcat (Stash), but that configuration is very unusual, and not recommended in most circumstances.

Securing Stash with Tomcat using SSL

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

If you are setting up a production instance you should consider [using a CA certificate](#), briefly described below.

There are other network topology options for running Stash, including running Stash behind a reverse proxy. For an overview of some common options, see [Proxying and securing Stash](#).

When Stash is set up following the instructions on this page, access to Stash is direct, and all communication between the user's browser and Stash will be secured using SSL.

On this page:

1. [Generate a self-signed certificate](#)
 2. [Configure HTTPS in Tomcat](#)
- [Exporting the self-signed certificate](#)
[Requesting a CA certificate](#)
[Troubleshooting](#)

Related pages:

- [Integrating Stash with Apache HTTP Server](#)
- [Securing Stash with Apache using SSL](#)



Client browser



Secured



Stash server

Note that:

- Stash will listen for requests on port 8443, according to the instructions in [Securing Stash with Tomcat using SSL](#). This port can be changed if required.
- The address with which to access Stash, by default, will be `https://<computer name>:8443`. Change the [base URL for Stash](#) if required.

- Any existing [links with other applications](#) will need to be reconfigured using this new URL for Stash.
- You can [set the context path](#) for Stash if you are running another Atlassian application, or Java web application, at the same hostname and context path as Stash.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).



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:

Windows	<code>"%JAVA_HOME%\bin\keytool" -genkey -alias tomcat -keyalg RSA</code>
Linux, MacOS and Unix	<code>\$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA</code>

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:

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


```
<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 here instead of 443 to avoid conflicts).

2. Comment out the existing Connector directive for port 7990 in conf/server.xml, so as to disable HTTP access, if you want all access to Stash to make use of HTTPS. That is, comment out this directive:

```
<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" />
```

3. Start, or re-start, Stash. You will be able to access Stash at <https://localhost:8443/> in your browser.

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:

Windows	<code>%JAVA_HOME%\bin\keytool" -export -alias tomcat -file file.cer</code>
Linux, MacOS and Unix	<code>\$JAVA_HOME/bin/keytool -export -alias tomcat -file file.cer</code>

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](#), [Thawte](#) or [TrustCenter](#). The instructions below are adapted from the [Tomcat documentation](#).

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](#).
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	<code>"%JAVA_HOME%\bin\keytool" -certreq -keyalg RSA -alias tomcat -file certreq.csr -keystore <MY_KEYSTORE_FILENAME></code>
Linux, MacOS and Unix	<code>\$JAVA_HOME/bin/keytool -certreq -keyalg RSA -alias tomcat -file certreq.csr -keystore <MY_KEYSTORE_FILENAME></code>

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	<code>"%JAVA_HOME%\bin\keytool" -import -alias tomcat -file file.cer</code>
Linux, MacOS and Unix	<code>\$JAVA_HOME/bin/keytool -import -alias tomcat -file file.cer</code>

Troubleshooting

Here are some troubleshooting tips if you are using a self-signed key created by `keytool`, or a CA certificate, 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 `.key store` 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`

1. Add the following attribute to the connector tag you uncommented:

```
keystoreFile="<location of keystore file>"
```

Incorrect password

```
java.io.IOException: Keystore was tampered with, or password was incorrect
```

You used a different password than "changeit". You must either use "changeit" for both the keystore password and for the key password for Tomcat, or if you want to use a different password, you must specify it using the `keystorePass` attribute of the Connector tag, as described above.

Passwords don't match

```
java.io.IOException: Cannot recover key
```

You specified a different value for the keystore password and the key password for Tomcat. Both passwords must be the same.

Wrong certificate

```
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:

```
<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

```
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"
  protocol="org.apache.coyote.http11.Http11Protocol"
  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.

Wrong certificate type

If the certificate from the CA is in PKSC12 format, add the `keystoreType` attribute to the SSL Connector in `conf/server.xml`.

```
keystoreFile="/opt/local/wildcard_atlassian_com.p12"
keystorePass="removed"
keystoreType="PKCS12" />
```

Certificate chain is incomplete

If the root certificate and intermediary certificate(s) aren't imported into the keystore before the entity/domain

certificate, you will see the following error:

```
[root@dev atlas]# /usr/java/jdk1.7.0_17/bin/keytool -import -alias
tomcat -file my_entity_cert.crt
Enter keystore password:
keytool error: java.lang.Exception: Failed to establish chain from reply
```

Most likely, the CA sent a compressed file containing several certificates. The import order matters so you must import the root certificate first, followed by one or many intermediate certificates, followed lastly by the entity/domain certificate. There are many resources online that provide guidance for [certificate installation for Tomcat \(Java-based\) web servers using keytool](#).

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. You may wish to do this if you want to:

- use a different port number to access Stash
- use a different context path to 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 Tomcat using SSL](#). For an overview of other network topology options, see [Proxying and securing Stash](#). Otherwise, if you want to install Stash in an environment that incorporates Apache HTTP Server, this document is for you.

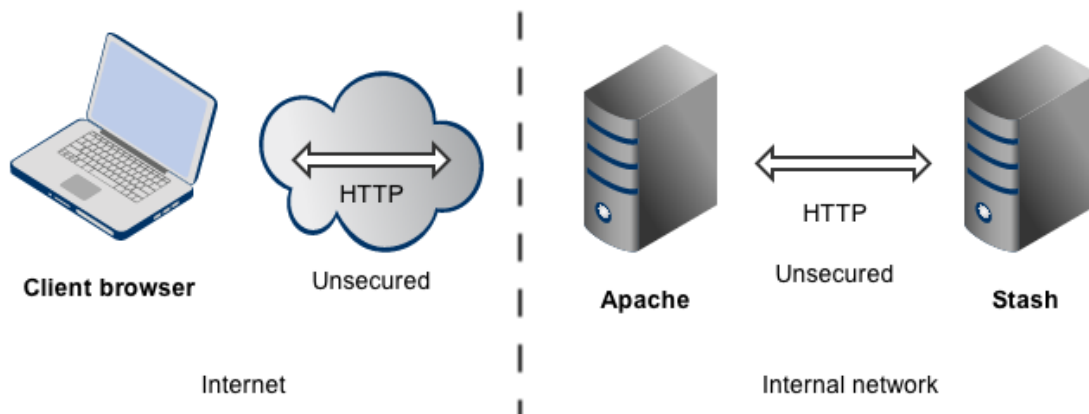
When Stash is set up following the instructions on this page, external access to Stash is via a reverse proxy, without using SSL. All communication between the user's browser and Apache, and so Stash, will be unsecured, but users do not have direct access to Stash.

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 Apache using SSL](#)
- [Securing Stash with Tomcat using SSL](#)



Note that:

- Stash, by default, will listen for requests on port 7990 – this port can be changed if required.
- Stash (Tomcat) needs to know the URL (proxy name) that Apache serves.
- The address with which to access Stash will be `http://<proxy name>:7990`. Change the [base URL for Stash](#) if required.
- Any existing [links with other applications](#) will need to be reconfigured using this new URL for Stash.
- You can [set the context path](#) for Stash if you are running another Atlassian application, or Java web application, at the same hostname and context path as Stash.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).

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 7

The Stash distribution includes an instance of Tomcat 7, 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 7.0 Proxy Support HowTo](#) page.

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`, `proxyName`, and `proxyPort` attributes as shown below. Instead of `mycompany.com`, set the `proxyName` attribute to your 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 URLs.

```
<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 Tomcat's `Connector` directive 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 7.0 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. Go to the Stash administration area and click **Server settings** (under 'Settings'), and change **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:

```
<Context path="/stash" docBase="${catalina.home}/atlassian-stash"
  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
ProxyPassReverse   /stash http://private.mycompany.com:9900/stash
```

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, follow the instructions on [Securing Stash with Apache using SSL](#). 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.

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 match 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** directive in your `httpd.conf`, and turn up logging (**"LogLevel debug"**) to get more info.
 3. Check out the [Stash Knowledge Base](#).

Securing Stash with Apache using SSL

You can run Stash behind a reverse proxy, such as Apache HTTP Server or nginx, that is secured using HTTPS (HTTP over SSL). You should consider doing this, and making secure access mandatory, if usernames, passwords and other proprietary data may be at risk.

There are other network topology options for running Stash; for an overview of some common options, see [Proxying and securing Stash](#).

When Stash is set up following the instructions on this page, external access to Stash is via Apache HTTP Server as a reverse proxy, where external communication with the proxy uses HTTPS. All communication between the user's browser and the Apache will be secured, whereas communication between the Apache and Stash will not be secured (it doesn't use SSL).

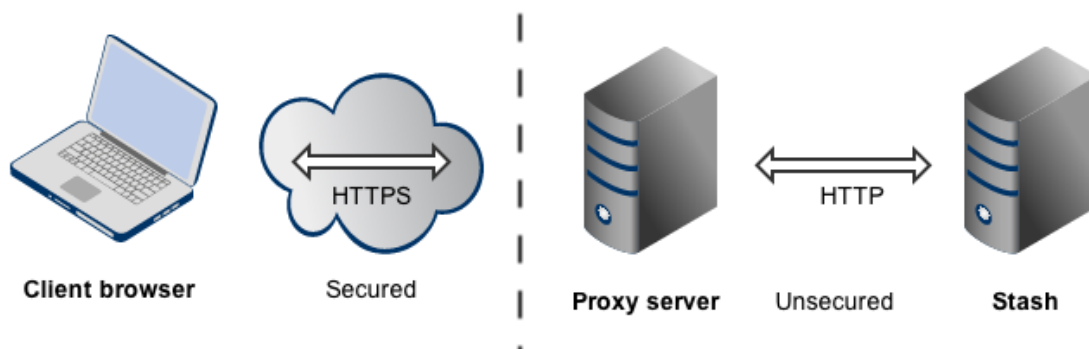
- The steps on this page would normally be performed after [integrating Stash with Apache HTTP Server](#).

On this page:

- Step 1: Configure the Tomcat Connector for SSL
- Step 2: Set up a virtual host in Apache HTTP Server
- Step 3: Create SSL certificate and key files
- Step 4: Update the base URL for 'https'
- Using a self-signed certificate

Related pages:

- Integrating Stash with Apache HTTP Server
- Securing Stash with Tomcat using SSL
- Securing Stash behind nginx using SSL



Note that:

- The reverse proxy (for example, Apache) will listen for requests on port 443.
- Stash, by default, will listen for requests on port 7990. Stash (Tomcat) needs to know the URL (proxy name) that the proxy serves.
- The address with which to access Stash will be `https://<proxyName>:<proxyPort>/<context path>`, for example `https://mycompany.com:443/stash`
- Any existing [links with other applications](#) will need to be reconfigured using this new URL for Stash.
- Stash (Tomcat) should be configured to refuse requests on port 7990 and to redirect those to the proxy on port 443.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).
- It would be possible to set up an SSL connection between the proxy server and Tomcat (Stash), but that configuration is very unusual, and not recommended in most circumstances.

Step 1: Configure the Tomcat Connector for SSL

Find the normal (non-SSL) `Connector` directive in Tomcat's `server.xml` file, and change the `redirectPort`, `scheme`, `proxyName` and `proxyPort` attributes as follows:

```
<Connector port="7990"
  protocol="HTTP/1.1"
  connectionTimeout="20000"
  useBodyEncodingForURI="true"
  redirectPort="443"
  compression="on"

  compressableMimeType="text/html,text/xml,text/plain,text/css,application/json,application/javascript,application/x-javascript"
  secure="true"
  scheme="https"
  proxyName="mycompany.com"
  proxyPort="443" />
```

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.

Start, or restart, Stash.

Step 2: 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"
  SSLCertificateChainFile "/usr/local/apache2/conf/server.crt"
  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 `SSLCertificateKeyFile` 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 `SSLCertificateChainFile` is optional. Please consult with the CA vendor to verify if this is required. This directive sets the optional all-in-one file where you can assemble the certificates of Certification Authorities (CA) which form the certificate chain of the server certificate.

The `ProxyPass` and `ProxyPassReverse` directives should be set up in the manner described in [Step 5](#) of the [Integrating Stash with Apache HTTP Server](#) page. In particular, if Stash is to run on a separate machine from Apache, you should use that domain (and perhaps the port number and context path) in the `ProxyPass` and `ProxyPassReverse` directives.

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.

Start, or restart, Apache.

Step 3: Create SSL certificate and key files

In [Step 2](#), 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 2](#).

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

Step 4: Update the base URL for 'https'

Open a browser window and log into Stash using an administrator account. Go to the Stash administration area and click **Server settings** (under 'Settings'). Change **Base URL** to use 'https', for example, "https://stash.mycompany.com").

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:

```
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:

```
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:

```
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.

Securing Stash behind nginx using SSL

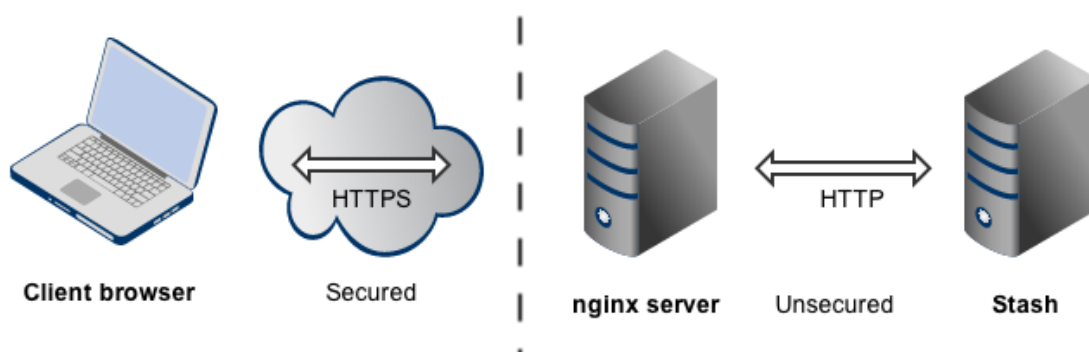
This page describes how to establish a network topology in which the nginx server acts as a [reverse proxy](#) for Stash. Typically, such a configuration would be used when Stash is installed in a protected zone 'behind the firewall', and nginx provides a gateway through which users outside the firewall can access Stash.

The configuration described on this page results in a scenario where:

- External client connections with nginx are secured using SSL. Connections between nginx and Stash are unsecured.
- Stash and nginx run on the same machine.
- Stash is available at <https://mycompany.com:7990/stash>.

On this page:

- [Step 1: Configure the Tomcat Connector](#)
- [Step 2: Set a context path for Stash](#)
- [Step 3: Change Stash's base URL](#)
- [Step 4: Configure nginx](#)
- [Resources](#)



Please note that:

- We assume that you already have a running instance of nginx. If not, refer to the [nginx documentation](#) for instructions on downloading and installing nginx.
- SSL certificates must be installed on the server machine.
- Any existing [links with other applications](#) will need to be reconfigured using the new URL for Stash.
- Securing Git operations between the user's computer and Stash is a separate consideration - see [Enabling SSH access to Git](#).

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 Tomcat using SSL](#). Otherwise, if you want to install Stash in an environment that incorporates nginx, this document is for you. (You can of course run Stash behind nginx without securing client connections to nginx using SSL – we don't describe this option on this page.)

Note that the [Atlassian Support Offering](#) does not cover nginx integration. Assistance with nginx may be obtained through the Atlassian community from answers.atlassian.com or from an [Atlassian Expert](#).

Step 1: Configure the Tomcat Connector

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

```
<Connector port="7990"
  protocol="HTTP/1.1"
  connectionTimeout="20000"
  useBodyEncodingForURI="true"
  redirectPort="443"
  compression="on"

  compressableMimeType="text/html,text/xml,text/plain,text/css,application/json,application/javascript,application/x-javascript"
  secure="true"
  scheme="https"
  proxyName="mycompany.com"
  proxyPort="443" />
```

For more information about configuring the Tomcat Connector, refer to the [Apache Tomcat 7.0 HTTP Connector Reference](#).

Step 2: 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 would be accessed at:

```
http://mycompany.com:7990/
```

For the example configuration on this page, we want Stash to be accessed at:

```
https://mycompany.com:7990/stash
```

In Tomcat's `server.xml` file, set the context path to `/stash`:

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

If you use a context path, it is important that the same path is:

- appended to the context path of Stash's base URL ([Step 3](#)).
- used when setting up the location for the `proxy_pass` directive ([Step 4](#)).

Step 3: Change Stash's base URL

After re-starting Stash, open a browser window and log into Stash using an administrator account. Go to the Stash administration area and click **Server settings** (under 'Settings'), and change **Base URL** to match the proxy URL (the URL that the nginx server will be serving).

For this example, use `http://mycompany.com:7990/stash` (Note the context path with this.)

Step 4: Configure nginx

Edit `/etc/nginx/nginx.conf`, using the example server configuration below, to configure nginx as a proxy server.

Put the `proxy_pass` directive in the location block, and specify the protocol, name and port of the proxied server in the parameter (in our case, it is `http://localhost:7990`):

```
server {
    listen      443;
    server_name mycompany.com;

    ssl          on;
    ssl_certificate      <path/to/your/certificate>;
    ssl_certificate_key  <path/to/your/certificate/key>;
    ssl_session_timeout  5m;
    ssl_protocols        SSLv2 SSLv3 TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers          RC4:HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    location /stash {
        proxy_pass      http://localhost:7990/stash;
        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Server $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_redirect   off;
    }
}
```

Refer to http://nginx.org/en/docs/http/nginx_http_proxy_module.html.

Changes made in the configuration file will not be applied until the command to reload configuration is sent to nginx or it is restarted. To reload the configuration, execute:

```
nginx -s reload
```

This command should be executed under the same user that started nginx.

Resources

You may find the following resources helpful in setting up Stash behind nginx:

- http://nginx.org/en/docs/http/configuring_https_servers.html
- <http://www.cyberciti.biz/tips/using-nginx-as-reverse-proxy.html>
- <https://mywushublog.com/2012/08/atlassian-tools-and-nginx/>

Moving Stash to a different context path

There are various reasons why you may wish to change the context path for Stash. Two of those are:

- You are running Stash behind a proxy.
- You have another Atlassian application, or Java web application, available at the same hostname and context path as Stash, and are experiencing login problems (see [Login and session conflicts with multiple Atlassian applications](#)).

Related pages:

- [Integrating Stash with Apache HTTP Server](#)
- [Login and session conflicts with multiple Atlassian applications](#)



Upgrade Note

Since the manual steps of this process modify the Stash distribution, you will need to repeat Steps 1-6 each time you upgrade Stash. See the [Stash upgrade guide](#).

Changing the context path for Stash:

1. Navigate to the directory where you are running Stash from. This is *the install directory that you extracted the Stash distribution to*, not your Stash home directory.
2. Stop Stash. This can be done using `bin/stop-stash.bat` on Windows or `bin/stop-stash.sh` on OSX or Linux.
3. Edit `conf/server.xml` and find the element below:

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

Update the `path` attribute to reflect the context path that you want Stash to be accessible at, e.g. `/stash`:

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

Then save the file.

4. Start Stash. This can be done using `bin/start-stash.bat` on Windows or `bin/start-stash.sh` on OSX or Linux.

Stash should now be available at the same host as before under the new context path. For example a server that was at <http://localhost:7990> will now be reachable at <http://localhost:7990/stash>.

5. Once Stash has started, go to the administration area and click **Server Settings** (under 'Settings'). Append the new context path to your base URL:

```
https://my-stash-hostname:7990/stash
```

6. Click **Save**.



Stash + Apache

Note that if you are running Stash behind Apache:

- You will need to make sure that the host or context path that Stash is exposed on is not also being used by another web application that is listening on a different port.
- If you have updated the Stash context path using the steps outlined above, you will need to update your Apache configuration, as described in [Integrating Stash with Apache HTTP Server](#).



Application Links

If you had Application Links set up before changing the context path in Stash, you will have to recreate those using the new Stash URL. See [Linking Stash with JIRA](#).

Changing the port that Stash listens on

You may wish to change the port that Stash listens on from the default '7990' to a different value if another application is already running on that port.

To change the port, edit the `conf\server.xml` file in the `<Stash installation directory>`.

Find the following lines in `server.xml`:


```
<Server port="8006" shutdown="SHUTDOWN">
...
<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"/>
```

You need to modify both the server port (default is 8006) and the connector port (default is 7990) to ports that are free on your machine. The server port is required by Tomcat but is not user-facing in any way. The connector port is the port you use to access Stash. For example, in the snippet above, the URL would be <http://example.com:7990>.

✔ Hint: You can use netstat to identify free ports on your machine. See more information on using netstat on [Windows](#) or on [Linux](#).

⚠ If you are using a firewall, you should ensure that it is configured to allow http/https traffic over the connector port you have chosen.

Related pages:

- [Specifying the base URL for Stash](#)

Running Stash with a dedicated user

For production installations, we recommend that you create a new dedicated user that will run Stash on your system. This user:

- Should be local.
- Should *not* have admin privileges.
- Should be a non-privileged user with read, write and execute access on the Stash install directory and [home directory](#).
- Should only have read access to your repositories.

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

For **Linux**, here is an example of how to create a dedicated user to run Stash:

```
$ sudo /usr/sbin/useradd --create-home --home-dir /usr/local/Stash --shell
/bin/bash Stash
```

Data recovery and backups

An effective backup strategy is essential:

- for avoiding data loss in the event of any system breakdown
- for restoring Stash after any system breakdown
- as part of the Stash upgrade process.

It is important to understand that there is a tight coupling between the Stash file system on disk and the database that Stash uses.

The following types of data are stored in each:

- The Stash home directory on the file system contains data about your repositories, as well as cache and log files (see [Stash home directory](#) for more detail).
- The Stash database contains data about pull requests (pointers to branches in the repos, comments and pull request diffs) and user management.

Any backup strategy that captures both the file system and database while Stash is still available to users runs the risk that the backed up Git repositories are corrupted or that the data in the database doesn't reflect the

repository state on disk. Therefore, strategies for backing up and restoring Stash data must keep the repository data and the database perfectly synchronised. Atlassian only recommends the backup and restore strategies described on this page:

- Use the [Stash backup client](#).
- Perform a [full cold backup](#).

With either strategy, schedule the backup window so as to minimise the impact on Stash availability. You might consider checking the access logs to determine patterns of lowest usage to help with this.

We highly recommend that you establish a data recovery plan that is aligned with your company's policies.

On this page:

- [Using the Stash backup client](#)
 - [Download and installation](#)
 - [Backing up Stash using the backup client](#)
 - [What is backed up](#)
 - [Cancelling the client backup](#)
 - [Restoring Stash into a new DB](#)
 - [Sample JDBC properties](#)
- [Using your organisation's backup processes](#)
 - [Backing up Stash when using the internal database](#)
 - [Backing up Stash when using an external database](#)
 - [Restoring Stash from a cold backup](#)

Related pages:

- [Connecting Stash to an external database](#)
- [Supported platforms](#)
- [Scheduling tasks on Linux](#)
- [Scheduling tasks on Windows](#)

Using the Stash backup client

The Stash backup client locks access to Stash, the repositories managed by Stash and the Stash database (this state is called 'maintenance mode'). It checks that all Git and database operations have completed, then performs a concurrent backup of the Stash database (for example MySQL), repositories and data. The client then unlocks Stash from maintenance mode.

A user will get an error message if they try to access the Stash web interface, or use the Stash hosting services, when Stash is in maintenance mode.

The backup client creates a single tar file for the backup in the specified location.

The client supports Windows and Linux platforms, and Stash versions 2.7 and higher.

As an indication of the unavailability time that can be expected when using the Stash backup client, in our testing and internal use we have seen downtimes for Stash of 7–8 minutes with repositories totalling 6 GB in size.

Download and installation

You can download the Stash backup client from the [Atlassian Marketplace](#).

[Download](#)

Unzip the client into a directory on the Stash server.

Backing up Stash using the backup client

The backup client must be run from somewhere with access to the Stash home directory. Usually, you will run the backup client directly on the Stash server. Run the client with the following command:

```
java -jar <path/to/stash-backup-client.jar>
```

Configuration options are kept in the `backup-config.properties` file that is included with the client – this file is automatically read when the client is run. The required properties include:

<code>stash.home</code>	The full path to the Stash home directory. On Windows, you must use two backslashes (\\) or a single forward slash (/) to separate paths.
<code>backup.home</code>	The full path to the location where the backup files should be saved. If <code>backup.home</code> is not specified, the backup client will use the current working directory. The client will create a <code>backup</code> subdirectory to store backups and a <code>log</code> directory to store logs within <code>backup.home</code> . On Windows, you must use two backslashes (\\) or a single forward slash (/) to separate paths.
<code>stash.baseUrl</code>	The Stash base URL.
<code>stash.user</code>	The username of the Stash sysadmin user that will perform the backup.
<code>stash.password</code>	The password of the Stash sysadmin user that will perform the backup.

Alternatively, these properties can be given on the command-line, when they need to be prefixed with "-D", and be placed before the "-jar" parameter. For example:

```
java -Dstash.password="pass" -Dstash.user="user" -Dstash.baseUrl="http://stash"
-jar stash-backup-client.jar
```

What is backed up

The client backs up the following:

- The database Stash is connected to (either the internal or external DB)
- Managed Git repositories
- The Stash audit logs
- Installed plugins and their data

The backup does NOT include the following files and directories:

- `export/*`
- `log/*` (except for the audit logs)
- `data/db*` (HSQL data in the DB is backed up, but the files on disk are not)
- `tmp`
- the `plugins` directory (except for the `installed-plugins` directory)

Note that the `caches` directory is included in the backup because it contains previously indexed heads.

Cancelling the client backup

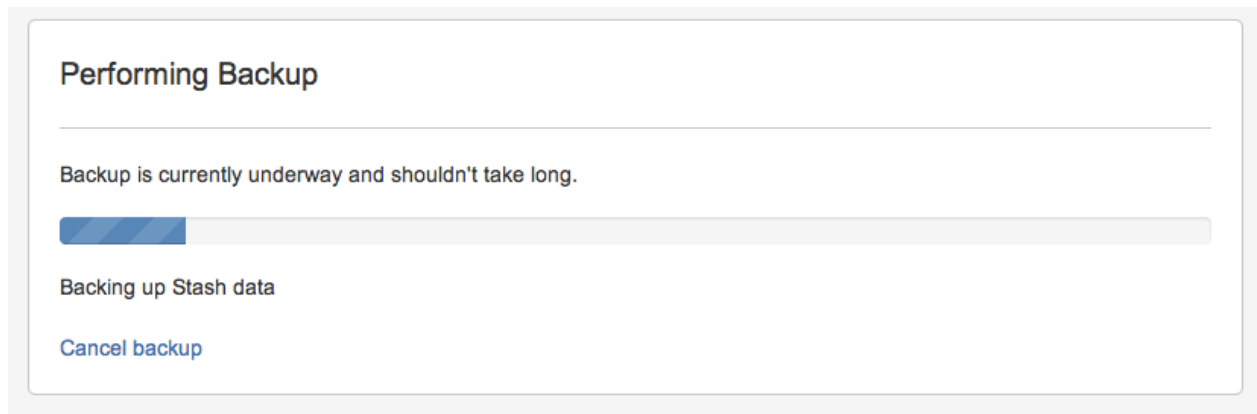
You can cancel the running client backup operation if necessary.

To cancel the backup:

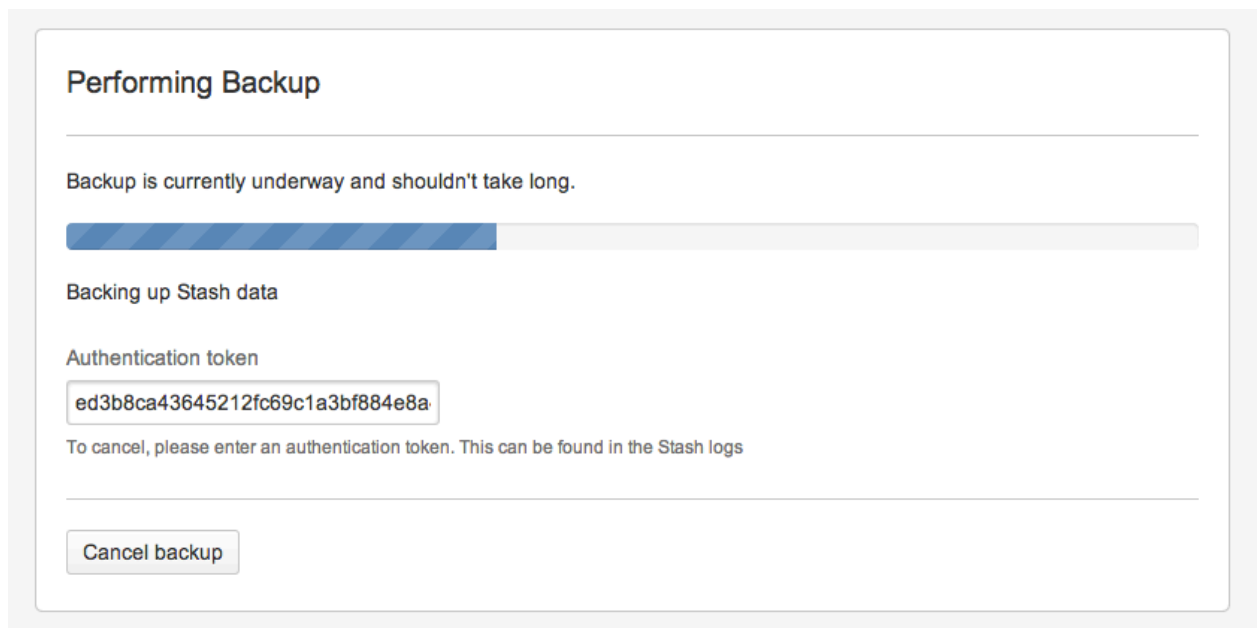
1. Copy the cancel token echoed by the client in the terminal (or the Command Prompt on Windows):

```
Contacting Stash
Using Stash 2.7.0
Stash has been locked for maintenance. It may be unlocked with token: 3e1e4991cb6bbd900f2d6a86197ffac508fc0c05
(3%) Starting database backup on Stash. It may be cancelled with token: ed3b8ca43645212fc69c1a3bf884e8a425477d24
(3%) Waiting for SCM operations to finish
(11%) Verifying Stash home
(15%) Verifying Stash home
(15%) Verifying Stash home
(23%) Backing up Stash home
```

2. Go to the Stash interface in your browser. Stash will display the this screen:



3. Click **Cancel backup**, enter the cancel token:



4. Click the **Cancel backup** button.

Restoring Stash into a new DB

When restoring Stash, the restore client must be run on the machine that Stash should be restored to. In order to ensure accidental restores do not delete existing data, the restore client will only restore into an empty home directory and an empty database.

Run the client with the following command:

```
java -jar <path/to/stash-restore-client.jar> <path to backup file>
```

The restore client also uses the `backup-config.properties` file, where the properties can include:

<code>stash.home</code>	The full path to a directory that the restore client will populate with the Stash home data. This directory must be empty. On Windows, you must use two backslashes (\\) or a single forward slash (/) to separate paths.
<code>jdbc.override</code>	By default, the restore client will restore into the same database that was backed up. If <code>jdbc.override</code> is set to <code>true</code> , the restore client will restore into the database specified by the below <code>jdbc</code> properties. The database must be empty.
<code>jdbc.driver</code>	The driver class that Stash should use to login to the new database. See examples below.
<code>jdbc.url</code>	The connection details for the new database, formatted as a JDBC URL. See examples below.
<code>jdbc.user</code>	The username that Stash should use to login to the new database.
<code>jdbc.password</code>	The password that Stash should use to login to the new database.

When given on the command-line, the system properties need to be prefixed with `-D`, and be placed before the `-jar` parameter. For example:

```
java -Dstash.home="path/to/stash/home" -jar stash-restore-client.jar
```

Sample JDBC properties

Sample `jdbc.driver` and `jdbc.url` properties are shown below.

Database	<code>jdbc.driver</code>	<code>jdbc.url</code>
MySQL	<code>com.mysql.jdbc.Driver</code>	<code>jdbc:mysql://HOSTNAME:PORT/DATABASE?autoReconnect=true&characterEncoding=utf8&useUnicode=true&sessionVariables=storage_engine%3DInnoDB</code>
Oracle	<code>oracle.jdbc.driver.OracleDriver</code>	<code>jdbc:oracle:thin:@//HOSTNAME:PORT/SERVICE</code>
PostgreSQL	<code>org.postgresql.Driver</code>	<code>jdbc:postgresql://HOSTNAME:PORT/DATABASE</code>
SQL Server	<code>com.microsoft.sqlserver.jdbc.SQLServerDriver</code>	<code>jdbc:sqlserver://HOSTNAME:PORT;databaseName=DATABASE;</code>

Using your organisation's backup processes

When you back up Stash using your standard process you must ensure that you keep the repository data and the database perfectly synchronised, by:

- Shutting down Stash before performing the backup.
- Restoring the database and file system at the same time.
- Using the same version or snapshot of the database and file system.

Backing up Stash when using the internal database

When Stash uses the built-in HSQL database, the database files are stored in the Stash file system. See [Stash home directory](#) for more detail.

Making a backup of Stash involves copying the Stash home directory.

Note that Atlassian does not recommend using the internal database for a production instance.

Backing up Stash when using an external database

When Stash uses an external database, both the [Stash home directory](#) and the external database must be backed up. You should follow your database vendor's recommendations regarding database backup tools and procedures.

Restoring Stash from a cold backup

Recovering a Stash instance typically involves restoring both the file system backup and the database backup.

Releases

You can get automated notifications about major Stash releases by subscribing to the [Atlassian dev tools blog](#).

The change logs for the major releases (linked below) have details of the related minor releases.

Stash 2.9

19 November 2013

- Branch listing improvements
- SSH access keys for projects and repositories
- Pull request inbox – my pull requests
- Build status during branch creation
- Support for PostgreSQL 9.3
- Extra keyboard shortcuts
- Extra merge strategy option
- Support for changing usernames

Read the [Stash 2.9 release notes and change log](#).

Stash 2.8

1 October 2013

- Stash branching model
- Branch creation from within Stash
- Branch creation from within JIRA
- Automated merges
- Branch listing page
- Move Git repositories between Stash projects
- Improved integration with Atlassian SourceTree
- Small improvements

Read the [Stash 2.8 release notes and change log](#).

Stash 2.7

20 August 2013

- JIRA issue transitions
- Support for multiple JIRA instances

- Autolink JIRA issues in markdown
- Backup and restore client (beta)
- Small improvements

Read the [Stash 2.7 release notes and change log](#).

Stash 2.6

22 July 2013

- Fork synchronisation
- Audit logging
- Repository Quicksearch
- Application navigator
- Public repositories list
- Small improvements

Read the [Stash 2.6 release notes and change log](#).

Stash 2.5

12 June 2013

- Public access to projects and repositories
- Edit a pull request's destination branch
- Get more context in diffs
- OpenJDK is now supported
- Small improvements

Read the [Stash 2.5 release notes and change log](#).

Stash 2.4

6 May 2013

- Forks
- Repository permissions
- Personal repositories
- Small improvements
- Deprecation of Java 6

Read the [Stash 2.4 release notes and change log](#).

Stash 2.3

26 March 2013

- Crowd single sign-on (SSO)
- Branch deletion
- Git submodule recognition
- SCM Cache plugin for Stash

Read the [Stash 2.3 release notes and change log](#).

Stash 2.2

05 March 2013

- Git repository hooks
- API for hook integrations
- Merge checks for pull requests

Read the [Stash 2.2 release notes and change log](#).

Stash 2.1

05 February 2013

- Pull request integration with JIRA
- Build status API
- Project avatars
- Pull request inbox
- Improved pull request title and description generation

Read the [Stash 2.1 release notes](#).

See the [change log](#) for Stash 2.1.x minor releases.

Stash 2.0

04 December 2012

- Branch permissions
- Markdown support
- Mentions
- Enterprise licenses for 1000 and 2000 users
- Deprecation of Internet Explorer 8

Read the [Stash 2.0 release notes](#).

See the [change log](#) for Stash 2.0.x minor releases.

Stash 1.3

09 October 2012

- Pull requests
- Notifications
- Improved keyboard shortcuts
- README – simple project documentation

Read the [Stash 1.3 release notes](#).

See the [change log](#) for Stash 1.3.x minor 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

This page describes how to upgrade Stash from a previous version.

- For the latest and greatest Stash release, see [Releases](#).

For production environments we recommend that you test the Stash upgrade on a QA server before deploying to production.

Please also read:

- the [Supported platforms](#) page for the full list of supported platforms for Stash.
- the [End of support announcements for Stash](#).
- the [Integrating Stash with Atlassian applications](#) page for version compatibility information.

On this page:

- [Upgrade steps](#)
- [Upgrading to Stash 2.9](#)
- [Upgrading to Stash 2.8](#)
- [Upgrading to Stash 2.7](#)
- [Upgrading to Stash 2.6](#)
- [Upgrading to Stash 2.5](#)
- [Upgrading to Stash 2.4](#)
- [Upgrading to Stash 2.3](#)
- [Upgrading to Stash 2.2](#)
- [Upgrading to Stash 2.1](#)
- [Upgrading to Stash 2.0](#)
- [Upgrading to Stash 1.3](#)
- [Upgrading to Stash 1.2](#)
- [Upgrading to Stash 1.1](#)
- [Upgrading from Stash 1.0.x to 1.1 or higher](#)

Related pages:

- [Releases](#)
- [Getting started](#)
- [Administering Stash](#)

Upgrade steps

This section provides general instructions for upgrading Stash. See also the specific notes on this page for the version of Stash you are upgrading to. 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.

See [Data recovery and backups](#) for more information.

3. Download and install Stash as usual

Download [Stash](#) from the Atlassian download site and unpack the archive file to a suitable location.

- Note that you should *not* simply copy the unpacked Stash directory over an existing Stash directory. Copying a new Stash directory over the existing one is not supported – that will *always* corrupt the system. This is because each version of Stash includes versioned jar files, such as `stash-model-2.4.1.jar`. If you copy these, you end up with multiple versions of Stash's jar files in the classpath, which leads to runtime corruption.

Update the value of `STASH_HOME` in the new Stash installation directory to point to your existing Stash [home directory](#) (but note that if you use a `STASH_HOME` environment variable to specify the home directory location then no change is required):

- On Windows, update the home directory location as defined by the `STASH_HOME` line of the new `<Stash installation directory>/bin/setenv.bat`.
- On Linux and Mac, update the home directory location as defined by the `STASH_HOME` line of the new `<Stash installation directory>/bin/setenv.sh`.



If you made custom changes to the configuration of your existing Stash installation

If you made custom changes to the configuration of your existing Stash installation you will have to make these changes for the new installation as well. For example:

- Changes to port, context path, and/or access protocol (`conf/server.xml`)
- Use of 64-bit libraries on Windows instead of the out-of-the-box 32-bit libraries (`bin/tomcat7.exe`, `bin/tcnative-1.dll`)

Do not simply copy the configuration files from your existing Stash installation to your new installation. Carefully review the differences between your customized version and the default version and re-apply your custom changes to the new configuration files to prevent overwriting configuration changes between different versions of Stash.



If you are using MySQL

Stash does not ship with the MySQL database driver.

You 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 home 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 to Stash 2.9

Please also see:

- the [Upgrade steps](#) section above.
- the [End of support announcements for Stash](#).
- the [Stash 2.9 release notes](#).

Note that Stash does not support Git 1.8.4.3.

When you first start Stash after upgrading to Stash 2.9 a repository upgrade task runs that optimizes the pull request refs for all repositories managed by Stash. It's important that you do not interrupt this upgrade process. You can track the progress of this in the Stash logs.

Known issues

Type	Key	Summary	Status
<div>  Authenticate to retrieve your issues </div>			







(No result found)





Upgrading to Stash 2.8


Please also see:

- the [Upgrade steps](#) section above.
- the [End of support announcements for Stash](#).
- the [Stash 2.8 release notes](#).

Known issues

Type	Key	Summary	Status
	STASH-4043	Branch List ahead/behind count incorrect	 Open
	STASH-4005	Clones over SSH fail for large repositories	 Open
	STASH-3884	Non-ASCII values used as branch model prefixes/branch names don't work in MSSQL	 Open

	STASH-3909	Creating a branch containing double quotes in the name will fail on Windows	 Open
	STASH-3908	Creating a branch in Stash with a prefix that matches an existing branch will fail	 Open

 [Authenticate](#) to retrieve your issues

5 issues

Upgrading to Stash 2.7

Please also see:

- the [Upgrade steps](#) section above.
- the [End of support announcements for Stash](#).













Repository System Information Plugin is now deprecated

The functionality of the [repository system information plugin](#) has now been moved into core Stash. The plugin will still work for Stash 2.x versions but is redundant as of Stash 2.7.


MySQL default isolation level

Stash 2.7.x uses READ_COMMITTED instead of the MySQL default isolation level (REPEATABLE_READ). This can result in exceptions when installing or upgrading to 2.7.x, if [binary logging](#) is enabled in your MySQL server. More details and a fix can be found in [this KB article](#).

Known issues

Type	Key	Summary	Status
	STASH-3829	Stash treats "<char - >" as An invalid or illegal XML character in the comments	 Open
	STASH-3828	Stash fails to start in Turkish environment	 Open
	STASH-3990	Mismatches in SQL Server database names cause inscrutable migration errors	 Open
	STASH-3891	Git's native .mailmap handling generates 500 errors in Stash for Git 1.8.2 to 1.8.4	 Open
	STASH-3803	Specifying 'since' and 'until' parameters on commits list only works if there aren't enough commits to trigger infinite scroll	 Open
	STASH-3773	Number of required	 Open

approvals doesn't get
updated when approving

 [Authenticate](#) to retrieve your issues

6 issues


Upgrading to Stash 2.6

Please also see:

- the [Upgrade steps](#) section above.
- the [End of support announcements for Stash](#).

Known issues

Type	Key	Summary	Status
	STASH-3687	Application navigator use wrong URL	 Open

 [Authenticate](#) to retrieve your issues











1 issues

Upgrading to Stash 2.5

Please also see:

- the [Upgrade steps](#) section above.
- the [End of support announcements for Stash](#).

Known issues

Type	Key	Summary	Status
	STASH-3702	Invalid markdown list rendering	 Open
	STASH-3635	Html in PR comments not encoded	 Open
	STASH-3624	SocketException on Solaris 10	 Open
	STASH-3589	Vertical alignment of back link on user profile screen is off	 Open
	STASH-3251	NPE in SshScmRequestComm andAdapter.java:238	 Open

 [Authenticate](#) to retrieve your issues

5 issues

Limited support for JIRA 4.4.x and earlier

JIRA 4.3+ allows for showing commits associated with issues in JIRA. However, viewing issues within Stash is





not supported for JIRA 4.4.x and earlier. See [JIRA compatibility](#) for details.


Upgrading to Stash 2.4

Please also see:

- the [Upgrade steps](#) section above.
- the [End of support announcements for Stash](#).

Known issues

Type	Key	Summary	Status
	STASH-3419	In full source display mode, longest line is cut off if horizontal scroll bar	 Open
	STASH-3163	Driver not available when running Stash as Windows service using \$STASH_HOME/lib	 Open

 [Authenticate](#) to retrieve your issues









2 issues


Upgrading to Stash 2.3

Please also see the [Upgrade steps](#) section above.

When upgrading to Stash 2.3 you also need to upgrade the SCM Cache plugin, due to recent Stash API changes.

Known issues

Type	Key	Summary	Status
	STASH-3461	Stash upgrade task does not check for database schema validity	 Open
	STASH-3308	Git command authentication failure causing the user to enter captcha before maximum attempts are met.	 Open
	STASH-3296	Using "Load newer activities" on pull requests drops some activities	 Open
	STASH-3279	URL not linked correctly when surrounding by parenthesis	 Open





 [Authenticate](#) to retrieve your issues


4 issues

Upgrading to Stash 2.2

Please see the [Upgrade steps](#) section above.

Known issues

Type	Key	Summary	Status
	STASH-3295	Comment Date format changes from relative to absolute dates unexpectedly	 Open
	STASH-3251	NPE in SshScmRequestComm andAdapter.java:238	 Open





 [Authenticate](#) to retrieve your issues


2 issues

Upgrading to Stash 2.1

Please also see the [Upgrade steps](#) section above.

Known issues

Type	Key	Summary	Status
	STASH-3319	atlas-install-plugin fails if the plugin is larger than 1 MB	 Open
	STASH-3163	Driver not available when running Stash as Windows service using \$STASH_HOME/lib	 Open

 [Authenticate](#) to retrieve your issues

2 issues

Install location for third-party libraries

As of Stash 2.1 you can install third-party libraries and jar files, such as the MySQL JDBC driver, into `<Stash home directory>/lib`. This has the advantage that files in this location are not overwritten, and lost, when you upgrade Stash.

Microsoft SQL Server JDBC driver

Stash 2.1 now uses the Microsoft SQL Server JDBC driver to access Microsoft SQL Server, instead of using the jTDS driver. In most cases, Stash will automatically swap to using the Microsoft driver on upgrade and **no configuration is required**.

If Stash was configured to use Microsoft SQL Server by manually entering a JDBC URL, please refer to [this guide](#).

Upgrading to Stash 2.0

This section provides specific notes for upgrading to Stash 2.0. See also the [Upgrade steps](#) section above.

Tomcat

For Stash 2.0, Tomcat has been upgraded from version 6 to 7. As part of that upgrade, the `server.xml` file has changed. If you have customised `server.xml` (for example, for `port`, `path` or `hostname`), you can not simply copy your own version across to the upgraded Stash; you must reapply your customisations to the `server.xml` file for the new version of Stash.

If you were running Stash as a Windows service and are upgrading from 1.x to 2.x you will need to reinstall the Stash service to make it use Tomcat 7.

To uninstall the Stash service you need to execute following commands from `<STASH DISTRIBUTION DIR>\bin`:

```
> net stop <service name>
> service.bat uninstall <service name>
```

You can call this command without the service name if you installed the Stash service with a default name.

After the service is uninstalled you can proceed with the [Upgrade steps](#) and [Running Stash as a Windows service](#) instructions to configure Stash 2.x running as a service.

Perl

Stash 2.0 requires Perl for its branch permission functionality. If Perl is unavailable, Stash 2.0 will not start.

On Windows machines, Perl will only have been installed by the Git installer if the correct install option was chosen at Step 4 of [Installing Stash on Windows](#).







Existing Git hooks


In order to support Branch Permissions, Stash 2.0 moves existing hooks in the `pre-receive` and `post-receive` folders under `<STASH_HOME>/data/repositories/NNNN/hooks` (where `NNN` is the internal repository id) to `.../hooks/pre-receive.d/10_custom` or `.../hooks/post-receive.d/10_custom`. Consequently, custom hooks that use relative path names (e.g. `./foo.sh` or `../dir/foo.sh`) will be broken by the upgrade to Stash 2.0.

Deprecation of Internet Explorer 8

Support for Internet Explorer 8 is deprecated from the release of Stash 2.0. The official end-of-support date is yet to be determined. See [Supported platforms](#) for details.

Known issues

Type	Key	Summary	Status
	STASH-2862	Non-ASCII values used in branch permissions don't work in MSSQL	 Open
	STASH-2898	Tomcat fails to start up on 64-bit Windows with error <code>tc-native-1.dll: Can't load IA 32-bit .dll</code>	 Open
	STASH-2889	Markdown: colon ':' in image urls is not accepted by current implementation of markdown	 Open

 [Authenticate](#) to retrieve your issues

3 issues

Upgrading to Stash 1.3

This section provides specific notes for upgrading to Stash 1.3. See also the [Upgrade steps](#) section above.

Email server

An email server must be configured in Stash so that email notifications for pull request events can be sent. Please see [Setting up your mail server](#) for details.





Upgrading from Stash 1.3 beta


✓ [Click to see information about upgrading from the beta...](#)

Stash 1.3 uses improved commenting compared to the 1.3 beta. This means that when you upgrade from the beta to Stash 1.3:

- Pull request comments made in the beta against the diff will only appear in the activity, and not in the diff, in Stash 1.3. Future comments made on the same pull request in Stash 1.3 will behave as expected.
- Reviewers, participants and watchers were added after the beta was released. After you upgrade, existing pull requests will not have participants or watchers. You can add reviewers by editing the pull requests and any future commenters will be added as a participant and watcher to the pull request as you would expect.

Known issues

Type	Key	Summary	Status
	STASH-2818	The latest custom JIRA issue key regex is not used to reindex the old commits	 Open
	STASH-2744	Pull Request race condition between push and attributed activity author	 Open





 [Authenticate](#) to retrieve your issues


2 issues

Upgrading to Stash 1.2

Please see the [Upgrade steps](#) section above.

Known issues

Type	Key	Summary	Status
	STASH-2746	Windows service behaves poorly (doesn't shut down, reports successful startup too early)	 Open
	STASH-2658	Web resource compression occurs before web resource copying on incremental plugin builds	 Open

 [Authenticate](#) to retrieve your issues

2 issues

Upgrading to Stash 1.1

Please see the [Upgrade steps](#) section above.

Upgrading from Stash 1.0.x to 1.1 or higher

Please also see the [Upgrade steps](#) section above.

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.** Known issues can be seen in [the STASH project on our issue tracker](#).
- **Stash Knowledge Base.** 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 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.

End of support announcements for Stash

End of support matrix for Stash

The table below summarises the end of support announcements for *upcoming* Stash releases. If a platform is not (or is no longer) supported by **Stash 2.9.x**, it is *not* listed in this table.

Platform/functionality	Announcement date	Stash end of support
Deprecation of Internet Explorer 8	22 July 2013	From Stash 3.0 (announcement)
Deprecation of Java 6	9 May 2013	From Stash 3.0 (announcement)

Why is Atlassian ending support for these platforms?

Atlassian is committed to delivering improvements and bug fixes as fast as possible. We are also committed to providing world class support for all the platforms our customers run our software on. However, as new versions of databases, web browsers etc. are released, the cost of supporting multiple platforms grows exponentially, making it harder to provide the level of support our customers have come to expect from us. Therefore, we no longer support platform versions marked as end-of-life by the vendor, or very old versions that are no longer widely used.

End of support announcements on this page (most recent announcements first):

- [Deprecation of Internet Explorer 8 \(announced 22 July 2013\)](#)
- [Deprecation of Java 6 \(announced 9 May 2013\)](#)

Deprecation of Internet Explorer 8 (announced 22 July 2013)

In version 3.0, Stash will no longer support Internet Explorer 8, and will only support Internet Explorer 9 and above. Stash 3.0 is expected to be released later in 2013. See [Supported platforms](#).

Deprecation of Java 6 (announced 9 May 2013)

In version 3.0, Stash will no longer support Java 6.0, and will only support Java 7.0 and above. Stash 3.0 is expected to be released later in 2013. See [Supported platforms](#).

Stash 2.9 release notes

19 November 2013

Introducing Stash 2.9

Today we're excited to announce Stash 2.9, which makes managing your branches, pull requests and access to repositories much easier.

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

Please also check the [End of support announcements for Stash](#).

The Stash 2.9 changelog is [at the bottom](#) of this page.



Try it for FREE ➞

Branch listing improvements

We've made the branches listing a much more useful overview of the state of your repository:

- The new **Pull requests** status helps you to track the review and merge work that still needs to be done.
- The **Behind/Ahead** column helps you identify work in progress as well as stale branches, and now you can choose the base branch for the comparison.
- The new search makes it easy to find branches when you have a ton of them. Even better, if you're using the Stash [branch model](#), you can filter by branch type simply by searching for the prefix – for example, search for "feature/" to see all your feature branches.
- As before, the **Builds** column allows you to see the build status of branches at a glance.
- The **Actions** menus have altered tasks, such as creating a pull request, and checking out the branch in [Atlassian SourceTree](#).
- There are new navigation shortcuts – see them in **Keyboard shortcuts** under the Stash Help menu.

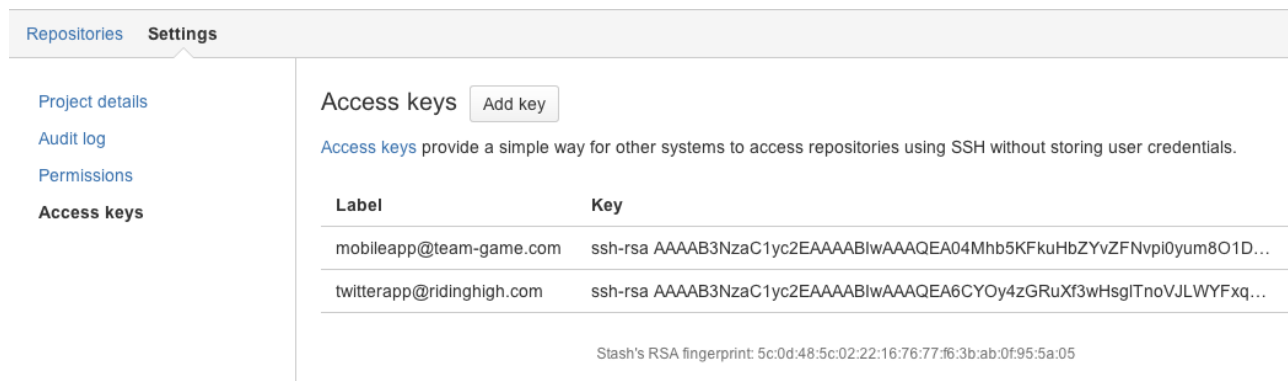
Files Commits Compare Branches Pull requests 14 Settings						
<div> master ... <input type="text" value="Search branches"/> Learn more </div>						
Branch	Behind/Ahead	Updated	Pull requests	Builds	Actions	
bugfix/STASHDEV-2623-branch-perms-max-pattern-validation	3	6 mins ago	OPEN		...	
STASHDEV-5560-ssh-urls-access-keys	8 1	5 hours ago	OPEN	✓	...	
master <small>DEFAULT BRANCH</small>		10 hours ago	DECLINED	!	...	
lower-case-slugs	60 1	4 days ago		!	...	
release/2.8	323	4 days ago	MERGED	✓	...	
release/2.7	359	4 days ago	MERGED	✓	...	
release/2.6	385	4 days ago	MERGED	✓	...	

Read more about the [branches listing...](#)

SSH access keys for projects and repositories

Stash now provides a simple way for other systems to perform read-only Git operations on repositories managed in Stash. This allows systems such as your build server to authenticate with Stash to checkout and test source code, without having to store user credentials on another system, and without requiring a specific user account.

These new access keys complement the personal account keys that have been available since Stash 1.1 – but they work for repositories rather than user accounts.



Access keys [Add key](#)

[Access keys](#) provide a simple way for other systems to access repositories using SSH without storing user credentials.

Label	Key
mobileapp@team-game.com	ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA04Mhb5KFkuHbZYvZFNvpi0yum8O1D...
twitterapp@ridinghigh.com	ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA6CYOy4zGRuXf3wHsglTnoVJLWYFqx...

Stash's RSA fingerprint: 5c:0d:48:5c:02:22:16:76:77:f6:3b:ab:0f:95:5a:05

Read more about [access keys ...](#)

Pull request inbox

Our pull request inbox has always provided easy access to the pull requests that need your review. In Stash 2.9, we've improved the inbox so that it also shows your own open pull requests. Toggle between these two sets directly from your inbox, and keep track of all your open pull requests seamlessly.



My pull requests

[Reviewing](#) [Created](#)

Repository	Title	Author	Reviewers
stash	Bugfix/testing merges	Vivien Leong	

Read more about [pull requests in Stash...](#)

Small improvements

Build status during branch creation

When you're creating a new branch, Stash displays the current build status for the source branch – is the branch good enough to branch from? [Read more about build status ...](#)

Create branch

Repository Stash / stash

Branch type Bugfix

[Learn about branch types](#)

Branch from feature/STASHDEV-3033-u... !

Branch name bugfix/

Support for PostgreSQL 9.3

See [Supported platforms](#).

Extra keyboard shortcuts

We've added new keyboard shortcuts to help you navigate around the branch listing screen quickly – choose **Keyboard shortcuts** from the Stash Help menu to see these. Read more about the [branch listing screen](#).

Extra merge strategy option

We've added the `squash-ff-only` option. As with `squash`, it collapses all the incoming commits into a single commit directly to the target branch, never creating a merge, but it does so *only* if the source branch is fast-forward. If not, a `MergeException` will be thrown. See [Stash config properties](#).

Support for changing usernames

You can [change the username](#) for a user account that is hosted in Stash's internal user directory. If a user is renamed in your LDAP directory, Stash will pick that up in the next synchronisation, or when the user next logs in. All content and attributes associated with the original username will be automatically associated with the new username, including pull requests, comments and permissions. See the [Crowd 2.7 release notes](#).

Change log

This section will contain information about the Stash 2.9 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).


If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The issues listed below are the highlights of all those that have been resolved for the Stash 2.9 releases.

19 November 2013 – Stash 2.9.1

Type	Key	Summary
	STASH-3360	List of Owned Pull Requests
	STASH-2821	add support for deployment ssh keys
	STASH-3993	Enable the ability to iterate on changes during the review process while maintaining "clean history"
	STASH-3951	Support Postgres 9.3

	STASH-3906	Developer documentation: Plugin example mistakes
	STASH-3711	Documentation improvement - constraints to Git Repo Size
	STASH-3469	Pull request references in the Git repository are never removed after merge or decline
	STASH-4021	SSH SCM Cache causes " c.a.s.i.s.g.p.ssh.GitSshScmReq uest upload-pack did not complete. It will be assumed the command failed" errors
	STASH-4018	Duplicate entries on the project list
	STASH-4011	File Search screen should not show README file
	STASH-3991	Stash dependency leaking into plugin dependencies
	STASH-3977	JIRA Feature Discovery behaves badly when switching tabs
	STASH-3961	Ref syncing raises RepositoryRefsChangedEvents with bad RefChanges
	STASH-3956	Ref syncing fails when notes are pushed
	STASH-3949	Source view fails with "Resetting to invalid mark" on larger source files
	STASH-3945	TrustedApplicationsAuthListener .authenticationFailure produces NullPointerExceptions
	STASH-3943	ORA-01795: maximum number of expressions in a list is 1000
	STASH-3914	BranchSelectorField doesn't work when appended to an intermediate non-DOM element with jQuery
	STASH-3871	Stash 2.7 fails to start with MySQL when binary logging is enabled
	STASH-3824	SSH operations hang when using PuTTY or TortoiseGit
	STASH-3535	PagedTable JS component spinner hiding race conditions

 Authenticate to retrieve your issues

[21 issues](#)

Stash 2.8 release notes

1 October 2013

Introducing Stash 2.8 – Git branch workflows

Today we're excited to announce Stash 2.8, which makes it easy for your team to use a branching workflow for your Git development process.

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

Please also check the [End of support announcements for Stash](#).

The Stash 2.8 changelog is [at the bottom](#) of this page.



Try it for FREE ➞

Stash branching model

Stash 2.8 introduces a 'branching model' that provides key advantages for streamlining your Git development workflow:

- Developers can make consistent naming decisions when [creating branches in Stash](#), guided by the naming conventions specified in the branching model.
- Reduce the need for manual maintenance of release branches, because consistent branch naming allows Stash to [automatically cascade merges](#) to those branch types.

Branching model

Development*	<input type="button" value="Use default branch"/>
The integration branch used for development. Feature branches are merged back into this branch	
Production	<input type="button" value="Select branch"/>
The branch used for deploying releases. Typically, this branches from the development branch and changes are merged back into the development branch	

Branch prefixes

Use the prefixes below as part of your branch names to categorize them and take advantage of automatic branching workflows. [Learn more](#)

Bugfix	<input type="text" value="bugfix/"/>	<input checked="" type="checkbox"/>
Typically used for fixing bugs against a release branch		
Feature	<input type="text" value="feature/"/>	<input checked="" type="checkbox"/>
Used for specific feature work. Typically, this branches from and merges back into the development branch		
Hotfix	<input type="text" value="hotfix/"/>	<input checked="" type="checkbox"/>
Typically used to quickly fix the production branch		
Release	<input type="text" value="release/"/>	<input checked="" type="checkbox"/>
Used for release tasks and long-term maintenance. Typically, this branches from the development branch and changes are merged back into the development branch		


Read more about the new [branching model](#) in Stash.

Branch creation from within Stash


You can now create new branches in your Git repositories directly from the Stash web UI. When a branching

model is configured, Stash guides you into making consistent choices for branch names:


Create branch

Repository  Paul Watson / stash ▾

Branch type **Bugfix** ▾
[Learn about branch types](#)

Branch from  STASH-2867-context-lines ▾

Branch name bugfix/ STASH-2887-context-lines




Create branch Cancel

Read more about [creating branches](#) in Stash.

Branch creation from within JIRA and JIRA Agile


As part of our push to provide tighter integration between Stash and JIRA, you can now start creating a branch from within a JIRA issue, whether you are in JIRA or JIRA Agile. This gives you a faster workflow from picking an issue to starting coding:



Dates

Created: 2 hours ago
 Updated: an hour ago

Development

 [Create Branch](#)

Agile

[View on Board](#)

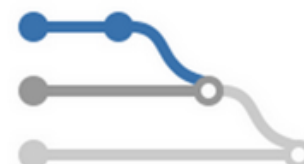
Stash will suggest the branch type and branch name, based on the JIRA issue type and summary – you can change these, of course.

Your Admin needs to have set up linking with JIRA before you'll see this. Read more about [creating branches](#) from JIRA.

Automated merges

When a branching model is configured in Stash that defines a 'release' branch for a repository, Stash can automatically merge changes to newer release branches. Cascading merges can reduce the manual maintenance for branches of that type.

Automated merging must be enabled for each repository, and is subject to a



number of conditions. Stash applies an ordering algorithm, based on patterns in the branch names, to determine the 'newer' branches.

Read more about [automatic branch merging](#) in Stash.

Branch listing pages

To complement branch workflows, Stash now provides an overview of all branches in a repository. The **Behind/Ahead** information shows by how many commits a branch has diverged from the default branch (for example, `master`) for the repository, and can help you to identify work in progress as well as stale branches. The **Actions** menus provide branch tasks such as viewing the branches source files, creating a pull request, creating a new branch, and checking out the branch in [Atlassian SourceTree](#).

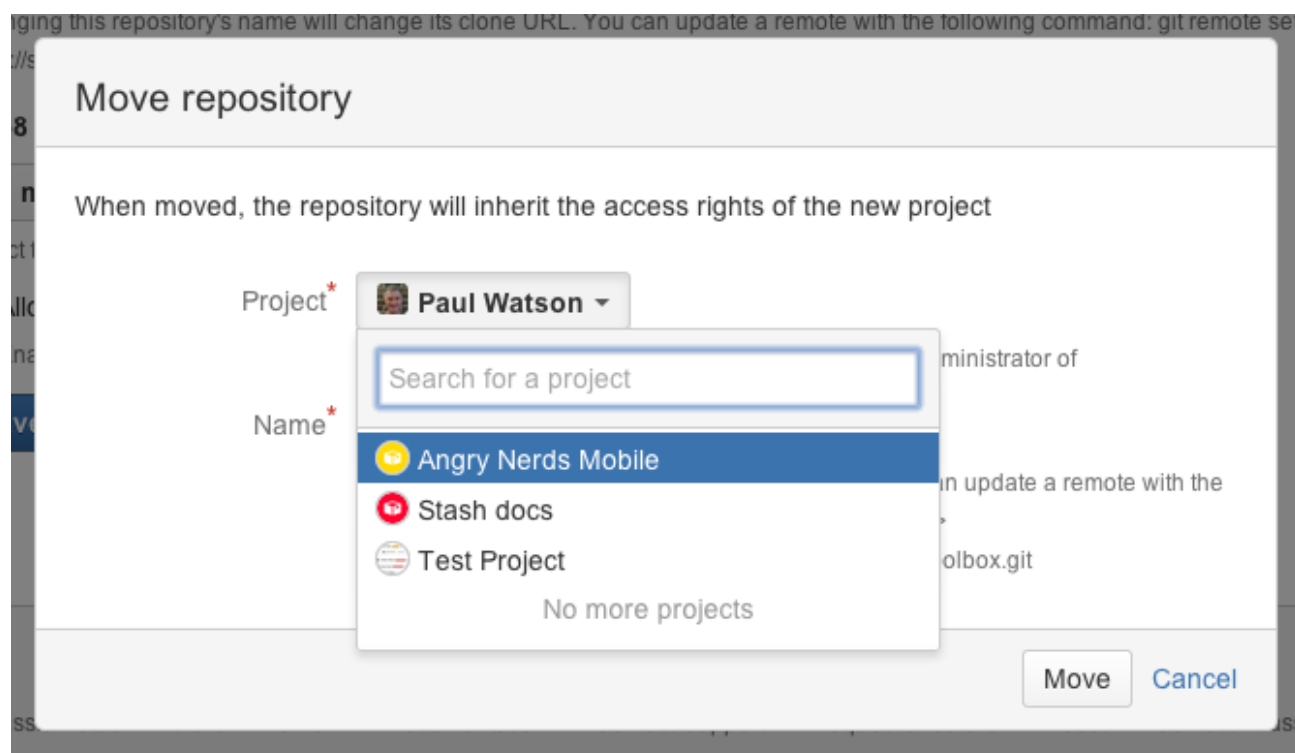
Files	Commits	Compare	Branches	Pull requests 12
Branch	Behind/Ahead	Updated	Builds	Actions
master <small>DEFAULT BRANCH</small>		2 hours ago		...
release/2.8	103 0	16 hours ago		...
feature/STASHDEV-4934-merge-dialog-pluggable-cancel-m...	114 2	17 hours ago		...
feature/devsummary-sokefull	923 16	18 hours ago		...
STASHDEV-4612-rest-coverage	16 6	18 hours ago		...

Read more about the [branches listing](#) for a repository in Stash.

Move Git repositories between Stash projects

You can now move a repository from one Stash project to another, if you have [project administrator](#) permissions for both projects.

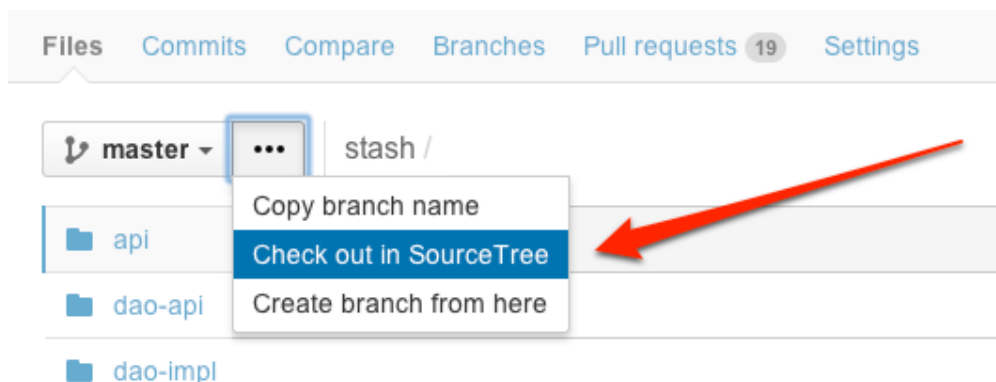
To move a repository, go to **Settings > Repository details** for the repository you wish to move, and click **Move repository**:



Improved integration with Atlassian SourceTree

You've been able to clone a repository from Stash using [Atlassian SourceTree](#) for a while. Now there are more ways that you can use Stash and SourceTree together, to help you use Git faster and more easily. Choose **Check out in SourceTree** from the Actions menu when you are:

- Viewing files or commits in a repository:



- Viewing branches in a repository:

Updated	Builds	Actions
10 mins ago		...
43 mins ago	✓	...
49 mins ago	✓	...
49 mins ago	✓	...
1 hour ago	✓	...
2 hours ago	✓	...

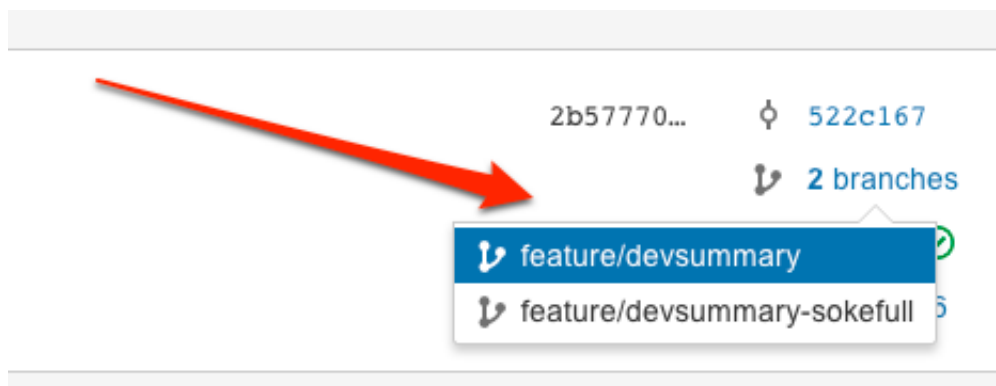
Small improvements

Stash backup client is now production-ready

The Stash backup client introduced with Stash 2.7.0 is now supported for use in production environments. See [Data recovery and backups](#).

See the branches that a commit is visible on

When browsing a commit you can now see all the branches that the current commit is visible on. This makes it easier to determine if a particular change has been merged to the master branch, for example, or not:



SSH support in the SCM Cache Plugin

The SCM Cache Plugin now supports SSH. This allows your CI build server to securely access the Stash server over SSH. See [Scaling Stash for Continuous Integration performance](#) for configuration information.

Auto-expand directories with only one sub-directory

In the files listing for a repository, Stash will now automatically expand a directory that contains only one sub-directory. This can save time where there is a series of nested single directories – Stash will continue expanding directories until it finds two or more sub-directories.

Change log






This section will contain information about the Stash 2.8 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).


If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The issues listed below are the highlights of all those that have been resolved for the Stash 2.8 releases.

5 November 2013 – Stash 2.8.3









Type	Key	Summary
	STASH-3979	Document move repository changes in 2.8 API changelog
	STASH-4026	Switching branches in file view causes internal error with branch names containing a "/"
	STASH-4018	Duplicate entries on the project list
	STASH-4000	Occasional 500 errors from stash - NPE in XSRF token validation
	STASH-3997	Branch permission patterns are case sensitive
	STASH-3991	Stash dependency leaking into plugin dependencies
	STASH-3987	Backups may be generated with out-of-date database data
	STASH-3986	Stash REST API does not consistently provide a nextPageStart on paginated resources

	STASH-3977	JIRA Feature Discovery behaves badly when switching tabs
	STASH-3914	BranchSelectorField doesn't work when appended to an intermediate non-DOM element with jQuery
	STASH-3356	Stash fails to connect to MySQL database if the user password contains special symbols
	STASH-3288	UI broken after inactive user added to pull request
	STASH-3253	Inconsistent behaviour for nextPageStart attribute on commits REST resource when specifying a since parameter

 [Authenticate](#) to retrieve your issues

13 issues

15 October 2013 – Stash 2.8.2


Type	Key	Summary
	STASH-3928	Create users without setting a password via REST
	STASH-3906	Developer documentation: Plugin example mistakes
	STASH-3973	Hook callback sockets are not closed correctly
	STASH-3961	Ref syncing raises RepositoryRefsChangedEvents with bad RefChanges
	STASH-3956	Ref syncing fails when notes are pushed
	STASH-3952	Ahead and behind information is not visible on the branch page anonymously
	STASH-3949	Source view fails with "Resetting to invalid mark" on larger source files
	STASH-3947	Searching nested group memberships fails for users with a large number of groups
	STASH-3945	TrustedApplicationsAuthListener.authenticationFailure produces NullPointerExceptions
	STASH-3943	ORA-01795: maximum number


of expressions in a list is 1000

 Authenticate to retrieve your issues

10 issues

2 October 2013 – Stash 2.8.1

Type	Key	Summary
	STASH-3922	Cannot set a reviewer when editing a pull request

 Authenticate to retrieve your issues

1 issues

1 October 2013 – Stash 2.8.0

Type	Key	Summary
	STASH-3561	Create branch on Stash like on github
	STASH-3347	Delete branch from UI
	STASH-3122	API to hook into pull request merge process.
	STASH-2796	Possibility to move a git repository from one project to another
	STASH-2691	Display branches that particular commit is visible on
	STASH-3821	REST /commits should default "until" to HEAD when specifying "since"
	STASH-3888	Updating a trusted application fails with unique constraint violation on uk_trusted_app_restrict
	STASH-3881	stash fails to display reviews/commits where commit message includes <U+2028>
	STASH-3875	The default project permission is not getting cleared from the cache
	STASH-3819	ExternalProcessImpl.areOutputPumpsRunning is not thread-safe
	STASH-3793	git rev-list fails on startup
	STASH-3788	Pull request events are not audited



STASH-3777

Stash cannot connect to other Atlassian applications on non-standard HTTPS port



STASH-3748

since and until parameters combined within REST API makes jira-key attribute go missing



STASH-3204

DataIntegrityViolationException when configuring Trusted Application



Authenticate to retrieve your issues

15 issues

Stash 2.7 release notes

20 August 2013

Introducing Stash 2.7

Today we're pleased to announce Stash 2.7, which introduces a range of JIRA integration improvements, to help optimise your development workflows. We've also included a brand new Stash backup and restore solution.

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

Please also check the [End of support announcements for Stash](#).

The Stash 2.7 changelog is [at the bottom](#) of this page.

Try it for FREE ➞

JIRA issue transitions

Now you can easily [transition](#) a JIRA issue from within Stash. For example, when creating a pull request you may want to transition the issue into review. Click on a linked JIRA issue anywhere in Stash to see a dialog with the available workflow steps:

JIRA Issues

- STASHDEV-1509 Can't start Stash on Windows if trying to r...
- STASHDEV-4809 Release 2.7**
- STASHDEV-4847 Update licensing plugin to 1.14
- ITPROJ-4 This is an improvement for you to transiti...

Stash Dev / STASHDEV-4809 Release 2.7

Code Review Reopen Close Issue

Details

Type: ☒ Development Task Status: In Progress

Priority: Minor Assignee: Jason Hinch [Atlassian]

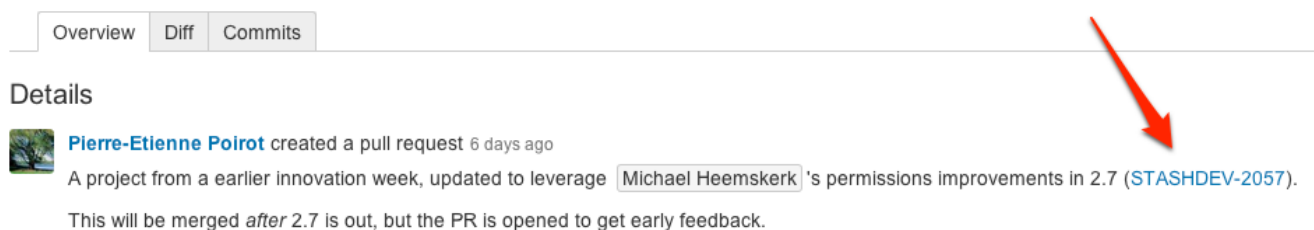
Description

<https://extranet.atlassian.com/display/STASH/Stash+2.7+Release+Checklist>

Your Admin needs to have set up linking with JIRA before you'll see this. Read more about [JIRA integration](#) in Stash.

Autolink JIRA issues in markdown

When you mention JIRA a issue key in Stash, for example in a pull request description or comment, the key gets automatically linked:



You can click on the linked key to see details for the issue. Note that issue keys in a URL will link directly to the JIRA instance, as you'd expect. Read more about [JIRA integration](#) in Stash.

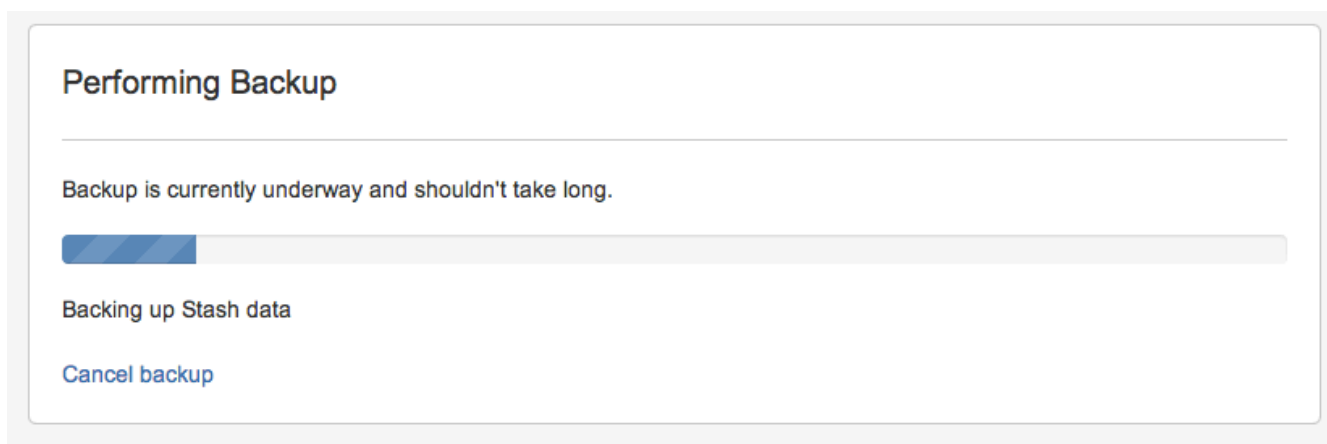
Support for multiple JIRA instances

Stash can now link to more than one JIRA server at a time, so different teams can work with their own projects in different JIRA instances, or a single team can link to issues across multiple JIRA servers.

Read more about [JIRA integration](#) in Stash.

Backup and restore beta

With the Stash 2.7 release we have made available a *beta* release of a backup and restore client for Stash. The Stash backup client locks access to Stash, checks that all Git and database operations have completed, and then performs a concurrent backup of the Stash database, repositories and data. You can download the Stash backup and restore client from the [Atlassian Marketplace](#).



Although we are confident about the performance of this tool, we are interested in hearing how well it meets your requirements in actual use – this is why we are releasing the client as a beta. We look forward to your feedback!

Read more about [backing up and restoring](#) your Stash data.

[Send feedback »](#)

Small improvements

Improved LDAP synchronisation performance

We have greatly improved the performance for LDAP synchronisation. For example, on a test LDAP server with 10,000 users and 10,000 groups, the time for a full synchronisation improved from 64 minutes to 4 minutes.

The Repository System Information plugin is now deprecated

The functionality of the [repository system information plugin](#) has now been moved into core Stash. The plugin will still work for Stash 2.x versions but is redundant as of Stash 2.7.

Syntax highlighters are now configurable

You can now update the extensions associated with different syntax highlighters using the `stash-config.properties` file in the [Stash home directory](#). For more details see [Configuring syntax highlighting for file extensions](#).

MySQL default isolation level

Stash 2.7.x uses `READ_COMMITTED` instead of the MySQL default isolation level (`REPEATABLE_READ`). This can result in exceptions when installing/upgrading to 2.7.x, if [binary logging](#) is enabled in your MySQL server. More details and a fix can be found in [this KB article](#).

Change log


This section will contain information about the Stash 2.7 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The issues listed below are just the highlights of all those that have been resolved for the Stash 2.7 release.


5 November 2013 – Stash 2.7.5


Type	Key	Summary
	STASH-3987	Backups may be generated with out-of-date database data

 [Authenticate](#) to retrieve your issues

1 issues





15 October 2013 – Stash 2.7.4

Type	Key	Summary
	STASH-3973	Hook callback sockets are not closed correctly


 [Authenticate](#) to retrieve your issues

1 issues

15 October 2013 – Stash 2.7.3 (Internal release only)

Type	Key	Summary
	STASH-3961	Ref syncing raises RepositoryRefsChangedEvents with bad RefChanges
	STASH-3956	Ref syncing fails when notes are pushed
	STASH-3952	Ahead and behind information is not visible on the branch page anonymously
	STASH-3949	Source view fails with "Resetting to invalid mark" on larger source files


	STASH-3947	Searching nested group memberships fails for users with a large number of groups
	STASH-3943	ORA-01795: maximum number of expressions in a list is 1000
	STASH-3888	Updating a trusted application fails with unique constraint violation on uk_trusted_app_restrict
	STASH-3881	stash fails to display reviews/commits where commit message includes <U+2028>
	STASH-3875	The default project permission is not getting cleared from the cache
	STASH-3777	Stash cannot connect to other Atlassian applications on non-standard HTTPS port
	STASH-3204	DataIntegrityViolationException when configuring Trusted Application

 [Authenticate](#) to retrieve your issues

11 issues




19 September 2013 - Stash 2.7.2

Type	Key	Summary
	STASH-3843	LazyReference\$InitializationException: java.lang.IllegalArgumentException: duplicate key: {username}
	STASH-3838	Public repositories not showing project after first 25


 [Authenticate](#) to retrieve your issues

2 issues

29 August 2013 - Stash 2.7.1

Type	Key	Summary
	STASH-3807	Show the first few lines of an exception on the console
	STASH-3805	Backup client fails if backup directory starts with same string as Stash home
	STASH-3804	Admin > Users returns empty

	STASH-3798	user list for Oracle database
	STASH-3793	nav-links and ref-syncing are not in the list of bundled plugins
	STASH-3788	git rev-list fails on startup
	STASH-3786	Pull request events are not audited
		StringIndexOutOfBoundsException: String index out of range: -1

 **Authenticate** to retrieve your issues

7 issues

20 August 2013 - Stash 2.7.0

Type	Key	Summary
	STASH-3634	De-Activated Users In Crowd Still Count Against the User License
	STASH-3720	Document .mailmap feature
	STASH-3698	Create an "End of Support Announcements for Stash" page
	STASH-3733	Bundle Repository System Info Plugin with Stash
	STASH-3594	Syntax Highlighting For Erlang Code
	STASH-2471	Support integration with multiple JIRA servers
	STASH-3632	Deactivated users synchronized when using JIRA for authentication
	STASH-2552	Provide a way to locate the repository directory on disk
	STASH-3651	JIRA issue link not obeying Display URL value in Application Links (JIRA Issues pop-up only)
	STASH-3588	Synchronising a directory fails on MySQL
	STASH-3577	Mouseover tooltip on "Approve" button always tells user to "Approve this pull request"
	STASH-3471	HistoryService lacks documentation
	STASH-3415	Hide upload public SSH key option, when SSH access is disabled



STASH-3301

About page doesn't respect that I'm logged in



STASH-2868

When using JIRA authentication, inactive users are shown in Stash



[Authenticate](#) to retrieve your issues

15 issues

Stash 2.6 release notes

22 July 2013

Fork synchronization with Stash 2.6

Today we're pleased to announce Stash 2.6, which introduces automated fork synchronization and audit logging. We've also added a repository search and polished some existing features.

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

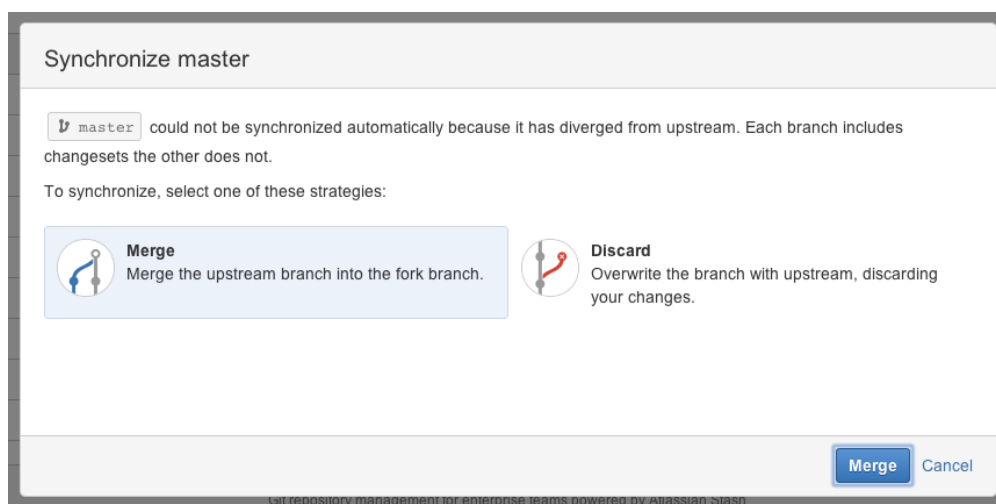
The Stash 2.6 changelog is [at the bottom](#) of this page.

Try it for FREE ➞

Fork synchronization

Fork syncing provides a way for your fork in Stash to be kept up-to-date with changes in the upstream repository, thus minimising the effort needed to do this manually. Stash can do this automatically, as long as you haven't modified the branches or tags in the fork.

If you have modified branches or tags in the fork, Stash will offer merge options:



Read more about [using forks](#) in Stash.

Audit logging

Stash now provides full audit logging. The most important audit events for each repository and project are displayed in Stash on the **Settings** tab (these are only visible to Stash admins and system admins):

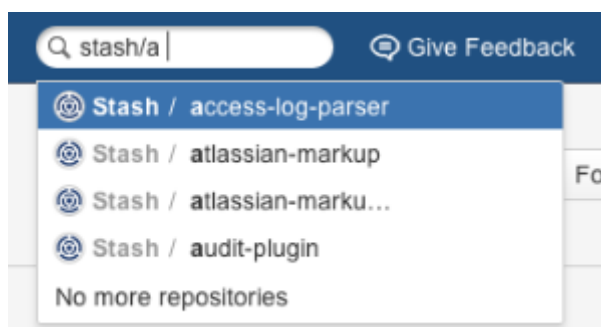
User	Action	Details	Time
Charles O'Far...	RestrictedRefRemovedEvent	Restricted reference with id 53 and value refs/heads/STASHDEV-3896-restore-navlinks with	Yesterday
Michael McGl...	RepositoryHookDisabledEvent	STASH/stash/bieber-baby/Bieber__Baby	Yesterday
Michael McGl...	RepositoryHookEnabledEvent	STASH/stash/bieber-baby/Bieber__Baby	Yesterday
Adam Ahmed	RestrictedRefAddedEvent	Restricted reference with id 54 and value <script>alert(1)</script> with groups [] users ["aahmed"]	4 days ago
Matthew Wat...	RestrictedRefAddedEvent	Restricted reference with id 53 and value refs/heads/STASHDEV-3896-restore-navlinks with groups [stash-users]	6 days ago
Jens Schuma...	RepositoryHookEnabledEvent	STASH/stash/com.atlassian.stash.stash-bundled-hooks:force-push-hook	26 Jun 2013

Stash also creates full audit log files that can be found in the `<Stash home directory>/log/audit` directory. The level of logging can be configured via system properties. Note that we recommend an automated backup of log files.

Read more about [audit logging](#) in Stash. Plugin developers can specify that their plugin events should be added to the 'Audit log' view, and the full audit log file.

Repository Quicksearch

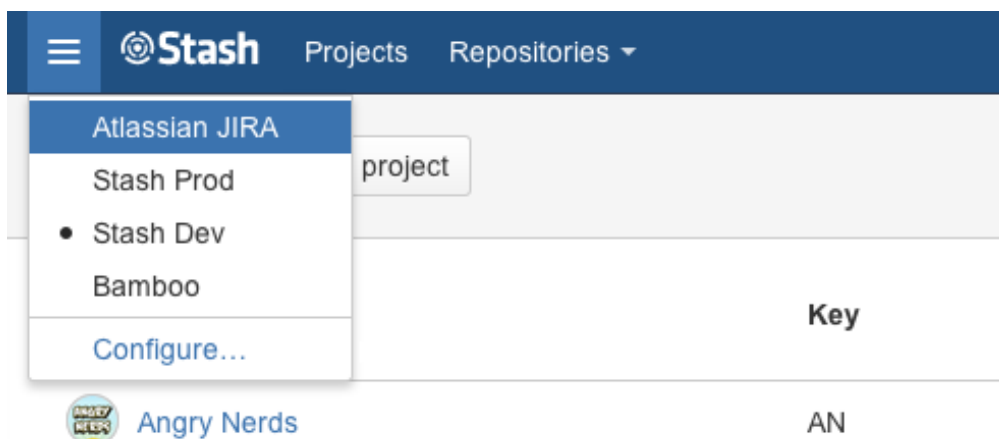
Quicksearch makes it easy to jump to a particular repository, without needing to know the name of the project it's in. Start typing to see a list of suggested repositories. If you type a '/', text before that is matched against project names, and text after is matched against repository names. Note that wildcards and antglobs are *not* supported.



Small improvements

Application navigator

The new application navigator, on the left of the header, allows you to switch to your other applications, such as Atlassian JIRA and Bamboo – or any other web application – all from the Stash header:



Stash administrators can configure which apps appear in the navigator – just click **Configure** in the application

navigator, or go to the Stash admin area and click **Application navigator**. See [Configuring the application navigator](#).

List of public repositories

Stash now displays a list of repositories for which anonymous access has been enabled – anonymous and logged-in users can choose **Repositories > View all public repositories**. See [Allowing public access to code](#).

Deprecation of Java 6

In version 3.0, Stash will no longer support Java 6.0, and will only support Java 7.0 and above. Stash 3.0 is expected to be released later in 2013. See [Supported platforms](#).

Deprecation of Internet Explorer 8

In version 3.0, Stash will no longer support Internet Explorer 8, and will only support Internet Explorer 9 and above. Stash 3.0 is expected to be released later in 2013. See [Supported platforms](#).

Change log


This section will contain information about the Stash 2.6 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The issues listed below are just the highlights of all those that have been resolved for the Stash 2.6.x releases.



12 August 2013 - Stash 2.6.4


Type	Key	Summary
	STASH-3952	Ahead and behind information is not visible on the branch page anonymously
	STASH-3888	Updating a trusted application fails with unique constraint violation on uk_trusted_app_restrict
	STASH-3742	Pull request merge fails if there are rename-add or directory-file conflicts
	STASH-3627	Horizontal Scrollbar Hides the Last Line
	STASH-3602	Blame line height is wrong on increased font size in browser
	STASH-3592	Adding buttons into stash.pull-request.toolbar.actions causes wrapping
	STASH-3204	DataIntegrityViolationException when configuring Trusted Application

 [Authenticate](#) to retrieve your issues

7 issues



5 August 2013 - Stash 2.6.3


Type	Key	Summary
	STASH-3696	Syntax highlight .pm extensions as Perl
	STASH-3694	LicenseServiceImpl#onGroupMembershipCreatedEvent performs potentially expensive operations causing slow LDAP synchronization

 [Authenticate](#) to retrieve your issues

2 issues






26 July 2013 - Stash 2.6.2


Type	Key	Summary
	STASH-3693	Outer sessions are not flushed during crowd synchronization slowing them down significantly for large directories
	STASH-3690	Fork sync issue, creates a merge commit with message that starts with "null"

 [Authenticate](#) to retrieve your issues

2 issues

















24 July 2013 - Stash 2.6.1


Type	Key	Summary
	STASH-3689	Stash search renders filenames with "null" as "\$1"
	STASH-3684	UserCreatedEvent should be transaction aware
	STASH-3681	DefaultCaptchaService#onUserCreated can be slow when synchronising large LDAP directories
	STASH-3676	CallbackLsTreeOutputHandler does not call onEndPage when the git process is interrupted
	STASH-3482	Java mail doesn't work when connecting Microsoft Exchange 2010

 Authenticate to retrieve your issues

5 issues

22 July 2013 - Stash 2.6.0

Type	Key	Summary
	STASH-3652	Provide a way to see all the repositories that are publicly accessible
	STASH-3550	Application Navigator in Stash
	STASH-3425	Sync branches/forks at the server automatically
	STASH-3390	Application Navigator
	STASH-3098	Finding a certain repository
	STASH-3075	Audit logging
	STASH-2983	Ability to Search for a Repository by Name
	STASH-3539	Only the first 25 SSH keys are shown for a user
	STASH-3659	Incorrect rendering of BranchSelector in RepositoryHook config-form
	STASH-3615	PageUtils.newRequest(0, Integer.MAX_VALUE) returns 1 item, but PageUtils.newRequest(0, 2) returns 2
	STASH-3593	Can't use PullRequestService.find method with SecurityService
	STASH-3514	Can't login with Crowd SSO user anymore if Crowd is started a bit later than Stash
	STASH-3506	Stash Footer
	STASH-3123	Support Tool incorrectly signals successful Support Request creation
	STASH-3057	Do a version check of Stash-Home directory before initializing server
	STASH-2465	Stash creates sessions for unauthenticated users

 Authenticate to retrieve your issues

16 issues

Stash 2.5 release notes

12 June 2013

Go public with Stash 2.5

Today we're pleased to announce Stash 2.5, which fosters external participation and lets you enjoy even more streamlined code collaboration with public access to projects and repositories.

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The Stash 2.5 changelog is [at the bottom](#) of this page.

Try it for FREE ➞

Public access to projects and repositories

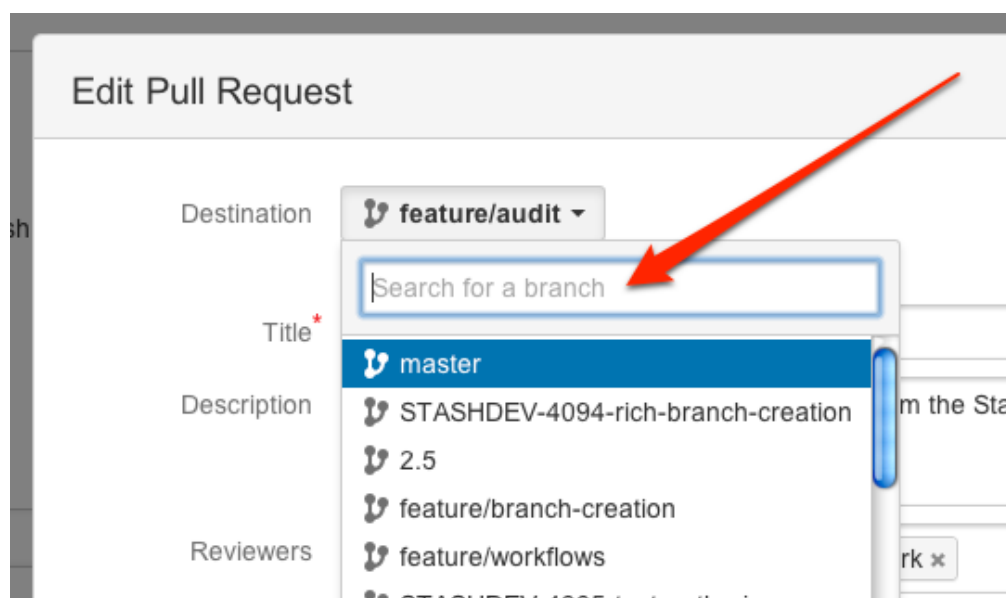
Public access provides anonymous (unauthenticated) users with *browse* access to specific repositories or entire projects in Stash. Anonymous users can see the source code in a repository, and clone the repository. Logged-in users get *read* access to a repository that has public access, so they can fork it and create pull requests. Public access doesn't allow your source code to be changed in any way.

You can configure public access to repositories and projects separately. You can also disable anonymous access by setting a global [system property](#). Please refer to the [Stash API changelog](#) for information about how plugin design is affected by public access.

Read more about [public access](#) in Stash.

Edit a pull request's destination branch

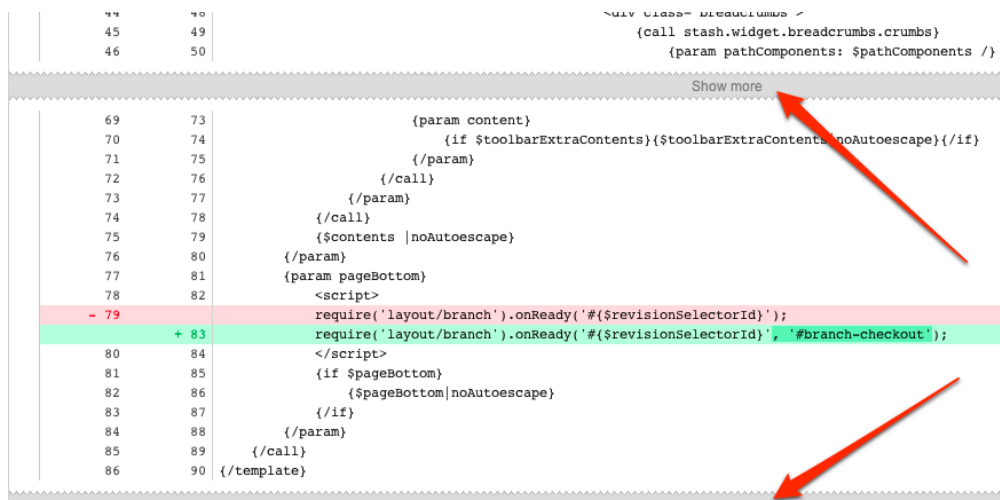
You can now change the destination branch for an active pull request – if you set the wrong destination when creating the PR, you no longer have to delete it and create a new one.



Read more about [pull requests](#) in Stash.

Get more context in diffs

We've added a way for you to see more lines of code in the diffs for pull requests and commits. We think this will really help with using pull requests for code reviews. Just click on the grey bars to see more context, up to the whole file if you want. There may also be grey bars at the top or bottom of the diff, if more context is available there.



Note that the expanded lines are displayed as greyed-out, to show that you can't comment on them.

Java OpenJDK is now supported

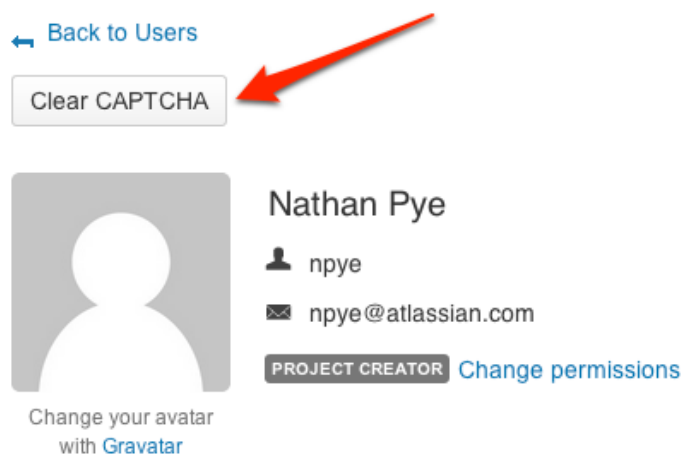
Java OpenJDK is widely used by Linux distributions. We've made your life easier by adding OpenJDK to the list of Java JDKs supported by Stash.

Just a reminder, Java 6 has been deprecated and will not be supported by Stash 3.0 (and later versions), which will be released later in 2013.

See [Supported platforms](#) for more information.

Small improvements

- JIRA issues that only appear in forks are now linked to the JIRA server.
- Stash now respects `.mailmap` files in Git repositories. See [Git resources](#) for details.
- The SCM Cache plugin is now bundled - see [Scaling Stash for Continuous Integration performance](#).
- We've added a bit more control for CAPTCHA:
 - There is now a [system property](#) switch to turn off (and hide) the **Maximum login attempts** control (look under 'Authentication' in the admin area).
 - Administrators are now able to clear a user's CAPTCHA status, from the user's profile:




Change log


This section will contain information about the Stash 2.5 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The issues listed below are just the highlights of all those that have been resolved for the Stash 2.5 release.







22 July 2013 - Stash 2.5.3


Type	Key	Summary
	STASH-3681	DefaultCaptchaService#onUserCreated can be slow when synchronising large LDAP directories

 [Authenticate](#) to retrieve your issues

1 issues




03 July 2013 - Stash 2.5.2

Type	Key	Summary
	STASH-3622	Elevation of global permission from Administrator to System administrator
	STASH-3604	changing an internal user's password as an "Administrator" fails
	STASH-3599	PermissionService#hasAnyUserPermission incorrectly requires a context user
	STASH-3593	Can't use PullRequestService.find method with SecurityService
	STASH-3590	New footer cuts off "t" explanation tooltip on the Pull Request list
	STASH-3197	README markdown doesn't display if "too large"

 [Authenticate](#) to retrieve your issues

6 issues

25 June 2013 - Stash 2.5.1

Type	Key	Summary
	STASH-3579	IE8: Adding reviewers doesn't work on Stash version 2.5.0
	STASH-3578	BAD_SIGNATURE or signature_invalid while creating Stash to JIRA application link
	STASH-3565	Typo in documentation when describing query string

parameters to find web fragments



STASH-3560

Commits added to pull request are not displayed in the correct order



STASH-3510

<resource name="view" > syntax in client-web-panel does not add web-resource dependency.



Authenticate to retrieve your issues

5 issues

03 July 2013 - Stash 2.5.2

Type	Key	Summary
	STASH-3622	Elevation of global permission from Administrator to System administrator
	STASH-3604	changing an internal user's password as an "Administrator" fails
	STASH-3599	PermissionService#hasAnyUserP ermission incorrectly requires a context user
	STASH-3593	Can't use PullRequestService.find method with SecurityService
	STASH-3590	New footer cuts off "t" explanation tooltip on the Pull Request list
	STASH-3197	README markdown doesn't display if "too large"



Authenticate to retrieve your issues

6 issues

25 June 2013 - Stash 2.5.1

Type	Key	Summary
	STASH-3579	IE8: Adding reviewers doesn't work on Stash version 2.5.0
	STASH-3578	BAD_SIGNATURE or signature_invalid while creating Stash to JIRA application link
	STASH-3565	Typo in documentation when describing query string parameters to find web fragments
	STASH-3560	Commits added to pull request are not displayed in the correct

order



STASH-3510

<resource name="view" > syntax in client-web-panel does not add web-resource dependency.



Authenticate to retrieve your issues

5 issues

11 June 2013 - Stash 2.5.0

Type	Key	Summary
	STASH-3476	Documentation: little documentation for overriding stash-default.properties
	STASH-2653	Support OpenJDK7
	STASH-2565	Support anonymous access in Stash
	STASH-3408	Allow @Context injection of ContainerRequest in all stash plugins
	STASH-3084	Support .mailmap
	STASH-2934	Change target branch for opened Pull Request
	STASH-2867	As a Stash User, I want to be able to specify the amount of 'context' provided for a diff
	STASH-3001	Allow CAPTCHA to be shut off completely at the command-line
	STASH-3668	Able to create a repository from Source Tree on a Stash project on which i do not have 'admin' access
	STASH-3521	D syntax highlighting is set to use the wrong extension.
	STASH-3513	Stash SSO NPE if Crowd fresh installation does not have SSO domain
	STASH-3439	Unable to find a JSON surrogate for an object of type com.atlassian.stash.internal.web.fragments.WebSectionData
	STASH-3403	JIRA issues not indexed in forks
	STASH-3400	MergeConflictHandler must use `git add -f` to ignore .gitignore
	STASH-3388	Issues with "My Pull Requests"

when there are more than nine
pull request entries




STASH-3332

Stash built from the source
distribution doesn't work



STASH-3129

README not displayed if both
README & README.md exist

 [Authenticate](#) to retrieve your issues

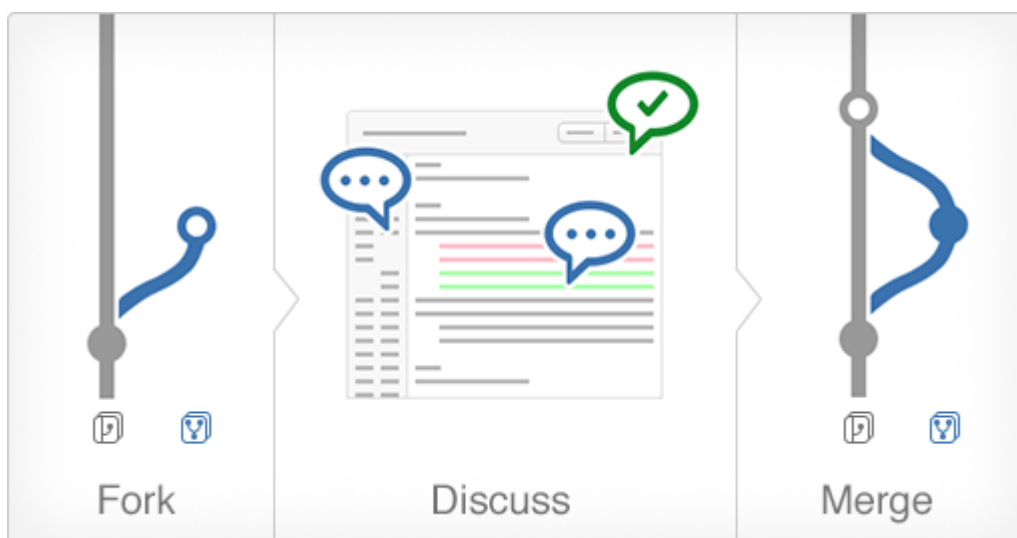
17 issues

Stash 2.4 release notes

6 May 2013

Introducing Stash 2.4 – Forking in the Enterprise

Today we're pleased to announce Stash 2.4, which introduces several key features to help you manage and collaborate on your Git repositories behind the firewall – forking, a workflow popularized in open source development, along with personal repositories and per-repository permissions.



If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The Stash 2.4 changelog is [at the bottom](#) of this page.

Try it for FREE ➞

Forks

Forks in Stash provide developers with a way to contribute code back to a repository for which they do not have write access. This extra control and flexibility of your development process lets you choose workflow scenarios that fit best. You can allow contractors to contribute to a project, or give developers the freedom to spike a project or prototype a feature, without risk to your core repositories.

Stash allows developers to fork a repository into any other project in Stash for which they have admin access. They can also create [personal forks](#) and then give others access by applying repository permissions.

Fork Angry Nerds / angry-nerds

Project

Angry Nerds Mobile

Where would you like to fork this repository into?

Name

angry-nerds

The repository's name will be used to create its URL
<https://stash.dev.internal.atlassian.com/scm/anm/angry-nerds.git>

Fork repository
Cancel

Read more about [using forks](#) in Stash.

Repository permissions

We've added repository permissions to Stash, which along with the already existing project and branch-level permissions, give you flexible yet fine-grained control over access to your repositories.

Repository permissions

Repository permissions allow you to extend access to users and groups beyond that already granted via [project permissions](#).

Individual Users

Name	Admin	Write	Read
Add Users			Read ▾ Add
Charles O'Farrell	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Giancarlo Lionetti	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Jens Schumacher	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Matthew Watson	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Paul Watson	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Stefan Saasen	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Thomas Bright	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Groups

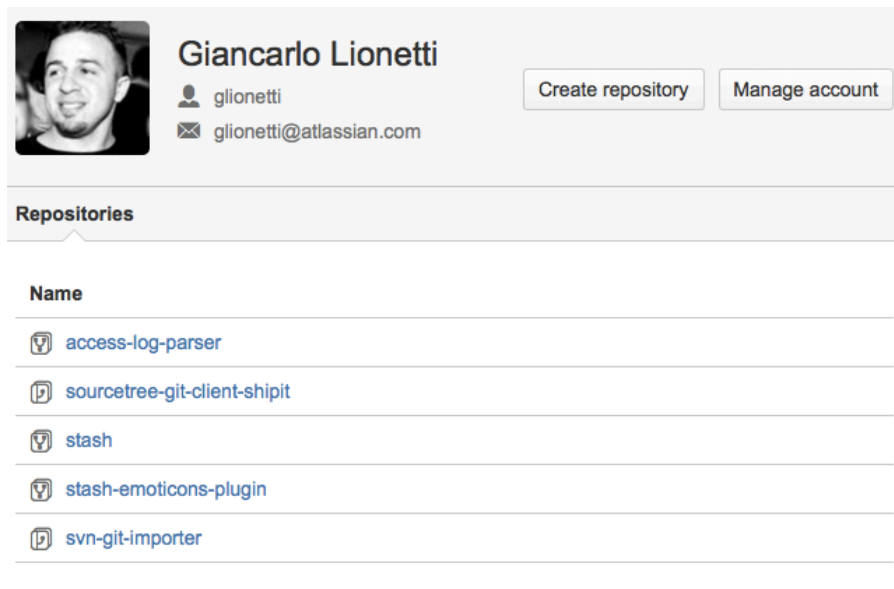
Name	Admin	Write	Read
Add Groups			Read ▾ Add
atlassian-docs	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
atlassian-staff	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
stash-users	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
sudoers-stash-dev	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Read more about [repository permissions](#) in Stash.

Personal repositories

You can now create personal repositories, unrelated to other projects, that you can use for such purposes as storing private snippets of work, kick-starting your own project, or contributing a bug-fix for a project that you are not a member of. By default, personal repositories are not visible to other Stash users. However, you can use [repository permissions](#) to open up access to other individuals and groups.

Your personal repositories are listed on the **Repositories** tab of your profile page. Every Stash user can see your profile page, but they can only see those repositories that you have given them permission to view.



Read more about [personal repositories](#) in Stash.

Deprecation of Java 6

This is to give advance notice that Stash 3.0 will not support Java 6.0, and will only support Java 7.0 and above. Stash 3.0 is expected to be released later in 2013.

API changes

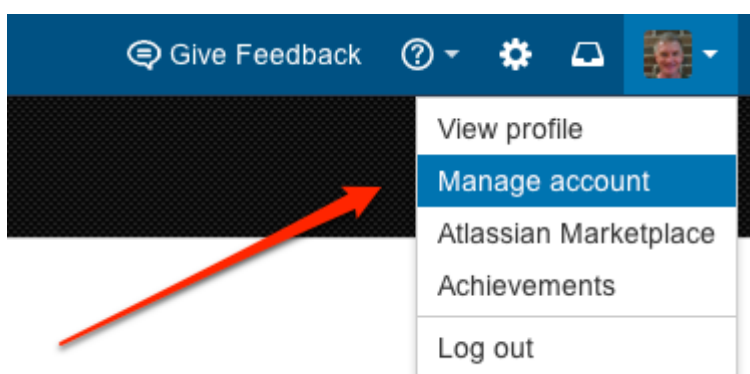
Please see the [API changelog](#) for any changes made for Stash 2.4.

Small improvements

Managing your Stash account

All the details of your Stash user account are now collected on your Stash account page. Use your account page to manage aspects of your Stash account, including changing your Stash password or avatar, and adding SSH keys.

To go to your account page simply choose **Manage account** from your user menu in the header:



Secret whitespace query parameter

Sometimes a diff is hard to read simply because multiple lines have only had the whitespace changed. To see just the important changes, use the `w=1` (`--ignore-all-space`) query parameter, which is the same as the `git diff` option. For example, the URL could look like:

```
https://stash-your_organisation.com/projects/STASH/repos/your_repo/commits/c573ea24e70c9d21ee?w=1#your_repo_path/DiffRequest.java
```

Before:

```

- 28      private DiffRequest(Repository repository, String sinceId, String untilId, List<String> paths, Whitespace whitespace) {
- 29          this.repository = repository;
- 30          this.sinceId = sinceId;
- 31          this.untilId = untilId;
- 32          this.paths = paths;
- 33          this.whitespace = whitespace;
- 34      }
- 35
And later in the diff...
+ 28      private DiffRequest(Repository repository, String sinceId, String untilId, List<String> paths, DiffWhitespace whitespace) {
+ 29          this.repository = repository;
+ 30          this.sinceId = sinceId;
+ 31          this.untilId = untilId;
+ 32          this.paths = paths;
+ 33          this.whitespace = whitespace;
+ 34      }

```

After:

```

      28      27
- 29      private DiffRequest(Repository repository, String sinceId, String untilId, List<String> paths, Whitespace whitespace) {
+ 28      private DiffRequest(Repository repository, String sinceId, String untilId, List<String> paths, DiffWhitespace whitespace) {
      30      29          this.repository = repository;
      31      30          this.sinceId = sinceId;
      32      31          this.untilId = untilId;
      33      32          this.paths = paths;
      34      33          this.whitespace = whitespace;
      35      34      }

```

Change log

This section will contain information about the Stash 2.4 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).


The issues listed below are just the highlights of all those that have been resolved for the Stash 2.4 release.

21 May 2013 - Stash 2.4.2

Type	Key	Summary
	STASH-3449	Log application version earlier, during startup
	STASH-3157	Improve column width layout for project and repository listings
	STASH-3475	Upgrade bundled Tomcat due to security vulnerabilities
	STASH-3463	Invalid "Authorization" headers for basic auth result in 500 errors
	STASH-3460	During directory synchronization, commit errors trigger cascading rollback failures
	STASH-3458	ResourceBundle ClassLoader lock contention under heavy load can soft-lock the system on Java 6
	STASH-3452	Cannot create pull request between same branch in different repos
	STASH-3448	Database errors during setup and migration produce useless messages
	STASH-3447	2.4 RestStashUser not backwards compatible
	STASH-3433	Files view shows an old version




of README.md


	STASH-3427	500 error if the user session has expired when opening the merge dialog
	STASH-3393	Forking Git commands fails silently on Solaris when there is not enough memory
	STASH-3096	WARNING: Problem with directory [/home/stash/atlassian-stash-data/lib], exists: [false], isDirectory: [false], canRead: [false]

 Authenticate to retrieve your issues

13 issues








8 May 2013 - Stash 2.4.1














Type	Key	Summary
	STASH-3411	Bad submodule file formatting causing 500 errors
	STASH-3410	ClientWebFragment NPE if an i18n bundle is available but no key given
	STASH-3405	2.4 Links to Stash Docs are pointing to the previous version


 Authenticate to retrieve your issues

3 issues

6 May 2013 - Stash 2.4.0

Type	Key	Summary
	STASH-3294	Document user directory related limitations
	STASH-3354	Add decorator for project settings
	STASH-3249	Make rename/copy thresholds in git diff configurable
	STASH-3118	Do not show login form for logged in users
	STASH-3112	Change Gravatar server URL
	STASH-2816	Provide configurable whitespace settings on the diff view
	STASH-2784	Document specifying a custom JIRA issue key regex

	STASH-2643	Repository Permissions
	STASH-2495	Repository forking
	STASH-3361	Restrict permissions on hook settings to repository administrators
	STASH-3351	Add decorator for project tab display
	STASH-3306	Single build link should be actual URL to build
	STASH-3071	Permalink to pull request comments
	STASH-2887	Improve Doc for 64bit TC
	STASH-3145	Doc Task - Update - Running Stash as a Windows service
	STASH-3045	Document process for running Stash as a 64-bit Windows service
	STASH-3367	Viewing old pull requests sometimes shows "Activity not found"
	STASH-3346	NPE with SSO plugin enabled & Crowd directory set without SSO Domain
	STASH-3285	Profile javascript errors for users with numeric usernames starting with zero
	STASH-3277	Administration menu highlight for current page is broken

 [Authenticate](#) to retrieve your issues

Showing 20 out of 24 issues

Stash 2.3 release notes

26 March 2013

With today's release of [Stash 2.3](#) we introduce features for enterprise teams (single sign-on), Git operations (submodule recognition and branch deletion) and CI performance scaling (the SCM Cache plugin).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The Stash 2.3 changelog is [at the bottom](#) of this page.

Try it for FREE ➞

Single sign-on with Crowd

Atlassian Crowd allows enterprise teams to integrate and deploy single sign-on (SSO) using popular directory

servers such as Active Directory or OpenLDAP. Now your team members need only log in once to any of your Atlassian applications (JIRA, Confluence, Bamboo) to be able to access all of them, without repeatedly typing in their password.

- *IT administrators* can centralize user management through Crowd and provide SSO for all Atlassian apps with minimal configuration.
- *End-users* enjoy the convenience of logging in once to any Atlassian application, avoiding the interruption of repeated authentication across other applications. Log in once, and you're automatically logged in to all applications connected to Crowd, including Stash.



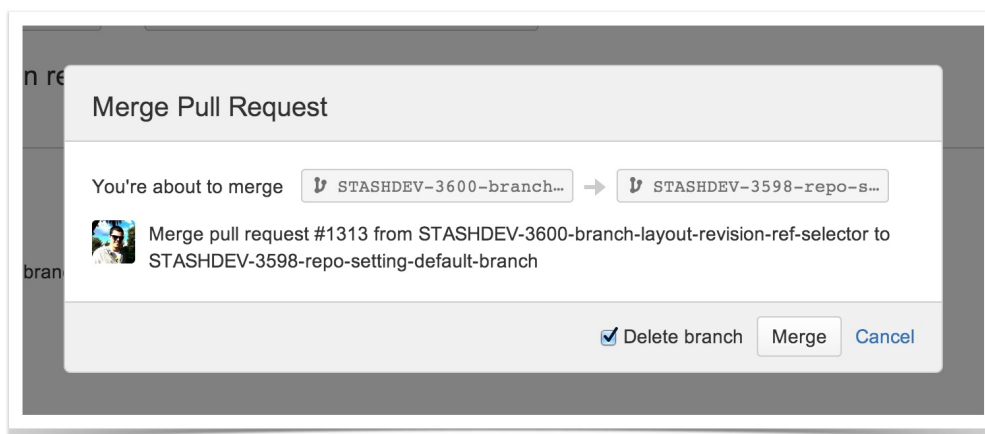
Read more about [using SSO with Stash](#).

Branch cleanup

We know that branches are often used for feature work, or for hot fixes that are destined to be applied to a stable or development branch. Once the code has been integrated into its destination branch it may often be desirable to delete the branch from Git as it is no longer required in the repository.

Stash now provides a simple way to delete the branch when you merge the pull request. As you might expect, Stash checks on a few things before allowing the deletion – the branch being merged will not be deleted if:

- The branch is the default repository branch.
- The user does not have permission to delete the branch.
- The branch is subject to an open pull request.

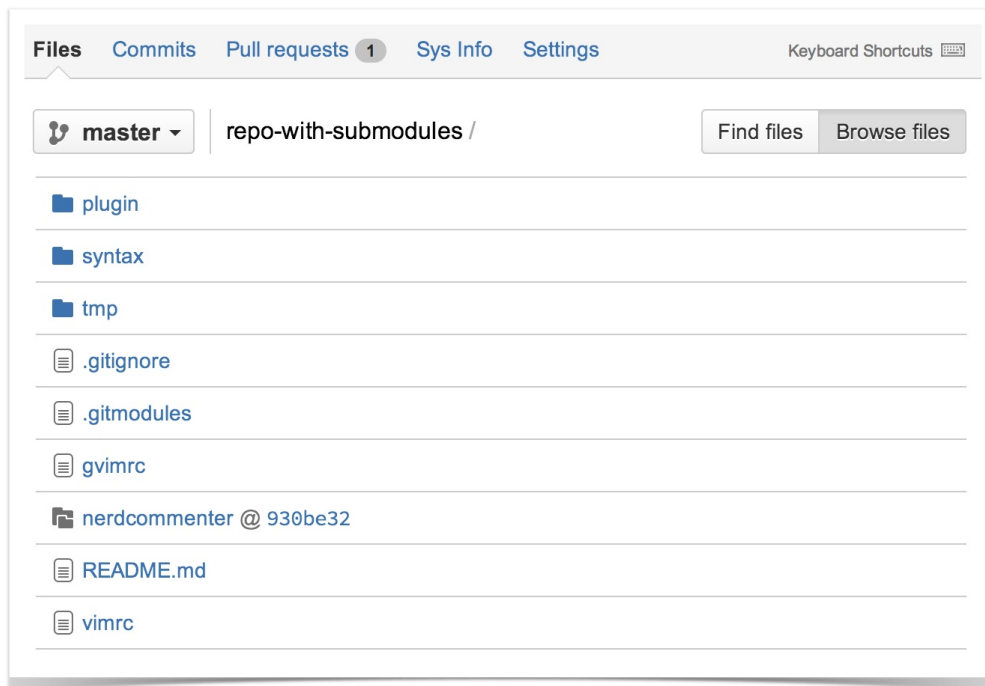


Read more about [branch deletion](#) when merging a pull request in Stash.

Git submodule recognition

Git submodules allow you to nest external Git repositories within the directory structure of your own repository. Submodules are commonly used to embed external codebases, such as shared libraries, that are updated independently of the main project.

Stash now makes it easy to identify Git submodules through the Stash user interface:



SCM Cache plugin for continuous integration performance

Stash instances with continuous integration (CI), or other automatic tooling set up to poll for changes, can end up with a high load on the Stash server due to frequent polling for changes, and bursts of repository clones when a change is detected.

The [Stash SCM Cache plugin](#) caches the pack files generated by Git during a clone operation on disk. This reduces the CPU load on your server and greatly improves response time when your server experiences repetitive requests from the continuous integration server.

Memory and CPU usage



Memory usage without SCM cache



Memory usage with SCM cache



CPU usage without SCM cache



CPU usage with SCM cache




Read more about [installing and configuring](#) the Stash SCM Cache plugin.


Change log

This section will contain information about the Stash 2.3 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

4 April 2013 - Stash 2.3.1


Type	Key	Summary
	STASH-3286	Unnecessary WARN log message when an unlicensed user attempts to log in.
	STASH-3285	Profile javascript errors for users with numeric usernames starting with zero
	STASH-3278	LDAP synchronisation fails with NumberFormatException
	STASH-3264	Crowd SSO integration is not enabled if Crowd has an empty SSO domain configuration

 Authenticate to retrieve your issues

4 issues

26 March 2013 - Stash 2.3.0

Type	Key	Summary
	STASH-2753	Option to delete a branch after pull request merge
	STASH-2625	Add Support for Submodules
	STASH-2493	Support Crowd SSO
	STASH-3228	Lowercase the project key in the clone url
	STASH-3202	Ability to review large Pull Requests
	STASH-3195	Longer repository names
	STASH-3189	Stash upm/requests links incorrectly default to staff picked
	STASH-3174	Public package is not exported in the build-integration plugin
	STASH-3117	NPE in ExternalProcessImpl.internalCancel
	STASH-2814	Stash does not remember my login

 Authenticate to retrieve your issues

10 issues

Stash 2.2 release notes

05 March 2013

With today's release of [Stash 2.2](#) we introduce several capabilities for customising your Git development workflow – Git repository hooks, an API for hook integrations, and merge checks for pull requests.

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

The Stash 2.2 changelog is [at the bottom](#).

Try it for FREE ➞

Git repository hooks

Git hooks allow you to customise your development team's workflow for any requirement they may have. Admins can enable and configure hooks for each repository, right within Stash, without having to install them on the file-system.

Stash comes bundled with repository hooks to let you start customising Stash straight off.

Hooks

Add hook

[Learn more](#)

Hooks allow you to extend what Stash does every time the repository changes (for example, when new code is pushed or when a pull request is merged). Hooks are installed by the system administrator and can be enabled by project administrators on a per-repository basis.

Pre receive - reject commits that don't match your policies



Reject Force Push

Reject all force pushes (git push --force) to this repository

Disabled

Enabled

Post receive - perform actions after commits are processed



HipChat Push Notification

Sends a notice to the specified HipChat room whenever someone pushes to the repository.

Disabled

Enabled

Read more about using [repository hooks](#) in Stash.

API for hook integrations

We've leveraged native Git hooks to create a new hooks API that allows developers to easily write their own hooks. Stash handles the persistence, packaging and per-repository configuration for your hooks, making it simple to extend Stash to suit your particular project's needs. Read more about [writing hooks for Stash](#) in our revamped developer docs.

Furthermore, you can mix-and-match bundled Stash hooks with hooks from the [Atlassian Marketplace](#). You can find and install these from within Stash – simply use the **Add hook** button on the hooks settings page to view available hooks from the marketplace.

Merge checks for pull requests

To help customise your workflow, you can now set checks to control when a pull request can be merged. Pull requests cannot be merged if the required checks have not been met. These checks are set separately on each repository in a Stash project:

- Require a minimum number of approvers – block merging of a pull request until it has been approved by the selected number of participants.
- Require a number of completed builds – stop pull requests from being merged if they have any unsuccessful builds, and until the set number of builds have completed.

Pull requests

☐ Requires approvers

At a minimum, pull requests must be approved by the number of users above before it can be merged

☐ Requires a minimum of successful builds

If there are more than the specified number of builds, all of them will have to be successful in order to merge the pull request




Read more about [merge checks](#) in Stash.

Change log


This section will contain information about the Stash 2.2 minor releases as they become available. These releases will be free to all customers with [active Stash software maintenance](#).

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).

05 March 2013 - Stash 2.2.0

Type	Key	Summary
	STASH-3140	Disable merging until a set number of people have approved the pull request
	STASH-3106	Block merging of pull requests until they have a successful branch build in bamboo
	STASH-2918	As a reviewer, I want my pull request approval withdrawn when new commits are pushed to a pull request
	STASH-2521	Add commit hooks
	STASH-3087	Include a link to driver changes for SQL Server in the upgrade guide
	STASH-2673	Show server SSH fingerprint in UI
	STASH-3088	Add a note about running stash with a dedicated user on *nix platform to the installation docs
	STASH-3081	Create documentation for changing port number
	STASH-3162	Pull Request REST resource returns only pull requests in an OPEN state
	STASH-3133	Stash 2.1.2 failing with MySQL 5.6 due to "specified key was too long" error
	STASH-3034	Merge Conflict Warning for Lack of Merge Rights
	STASH-2900	Commit hook permissions not set

up correctly when starting stash
from AMPS

 [Authenticate](#) to retrieve your issues

12 issues

Stash 2.1 release notes

5th February 2013

With today's release of [Stash 2.1](#), we've added a slew of new features to simplify your Git development workflow by providing more contextual awareness of key JIRA issue and Bamboo build information. Additional new features provide quick clicks to your recently viewed pull requests and notifications of your tasks. More contextual information, more tracking capabilities, more automation – that's Stash 2.1.

Try it for FREE ➞

Pull request integration with JIRA

Developers create pull requests when their code is ready for peer review before merging a development branch into the main code line. To make pull requests most effective, reviewers need more context around the changes: What bug or feature is this pull request resolving? What are the details of those issues? Are any of the issues still open?

Pull requests now tightly integrate with JIRA, putting issue details front and center. View the status of an issue, along with it's assignee and description to get the scoop without ever leaving Stash. This allows reviewers to

- Gain contextual awareness into the task which is being worked on by looking at descriptions, comments and attachments
- Quickly review the requirements for a new feature or bugfix
- Click straight through into JIRA, to keep issues up to date for upcoming releases

The screenshot shows the Stash web interface for a pull request. A modal window titled "JIRA Issues" is open, displaying a table of JIRA issues related to the pull request. The table has columns for Key, Summary, Assignee, P (Priority), and Status. Two issues are listed: STASHDEV-2893 (Closed) and STASHDEV-2931 (To review). A callout box highlights the text: "Important details about JIRA issues related to a pull request are at a reviewer's fingertips." Another callout box highlights the "2 JIRA Issues" badge on the right side of the pull request overview.

T	Key	Summary	Assignee	P	Status
	STASHDEV-2893	Plugin dev can specify a simple view	Michael Studman [...]	↓	Closed
	STASHDEV-2931	Hook settings need to be loaded/saved	Jason Hinch [Atlas...]	↓	To review

Read more about [pull requests](#) in Stash.

Build status API

Picture this: you're about to click the 'merge' button, and then you pause to consider: *did these changes pass all their tests?* To answer that, it's important to know what the latest build status of the branch is before the changes are integrated into master. But don't waste time navigating to your build system and sifting through test data to see whether the changes passed or failed. Stash 2.1 does it automatically. Our new build status API allows build servers, such as Bamboo and Jenkins, to publish build details to the pull request's overview, giving you a quick idea of whether the pull request is good to merge or not.

The screenshot shows the Stash web interface for a pull request. A modal window titled "Builds" is open, displaying a table of build status information. The table has columns for Commit, Message, and Status. Two builds are listed: Stash - Master (Failed) and Stash - Master Checks (Failed). A callout box highlights the text: "Builds".

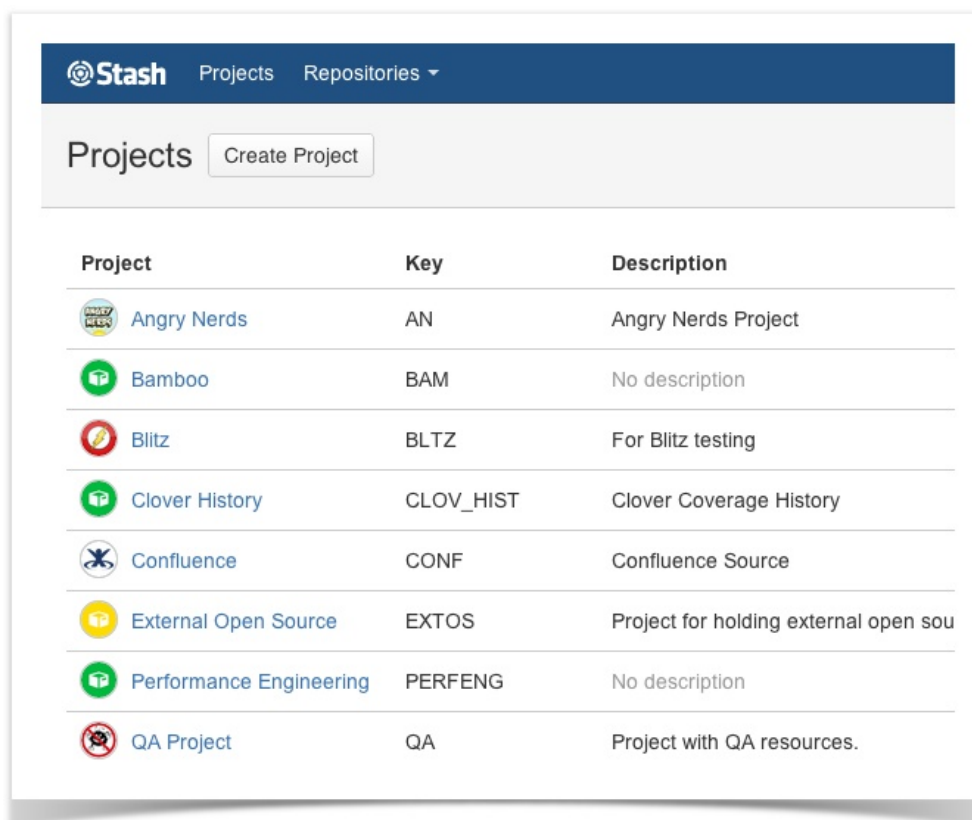
Commit	Message	Status
535b90f	STASHDEV-2931: Upg...	Failed
439ca9c	STASHDEV-2931: WIP	Failed
5ffaef23	STASHDEV-2931: Fix	Failed
976f35a	STASHDEV-2931: Bre	Failed
28315f6	STASHDEV-2931: Mov	Failed
4215fd0	STASHDEV-2931: App	Failed
21a85d3	STASHDEV-2954: explain versioning on pull requests and comments.	Failed
71dfeea	Automatic merge from 2.1 -> master # By Bryan Turner (2) and others # Via Bry...	Failed

But why stop with pull requests? The build status API will show you the build status of any commit, anywhere in Stash.

Read more about the [build status API](#) in our developer documentation.

Project Avatars

Give your projects some personality and make it easier to find them on the Dashboard with the new project avatars. Stash will randomly provide one of its colourful built-in avatars if you don't have one.



Pull request inbox

























When you work across multiple repositories and projects, it can be hard to keep track of all the reviews you have to complete. We solved the problem by providing a globally accessible inbox, displaying all Pull Requests that are waiting for your approval. Use the inbox as a task list, and leave no pull request behind!










Release log

A few handpicked tickets out of a list of more than 100 features and improvements.

New Features & Improvements	
	Pull requests show linked JIRA issue details
	Build status API
	Project avatars
	Track viewed files in a pull request
	Current branch is preserved when navigating between Files and Commits tab

	Improved pull request title & description generation
	Pull request inbox
	A user can see build status information in the pull request overview
	A user can see build status information in the commit details view
	A user can see build status information in the commit list
	Bundle the Scala runtime library and expose to Plugins
	Just in time comment drift calculation (less load on your server!)
	Add rich tooltips to disabled image diff toggles
	Add mention 'tip' to emails
	Added web sections for pull request metadata
	Stash now shows file mode changes (executable bit) in changesets and pull requests
	Add an "About" page
	Allow use of `@` when adding reviewers to a pull request
	Add default avatar to project create page
	Stash plugins can listen for user cleanup events
	Add endpoint to retrieve issue details for a pull request
	Improve error handling for reflog config upgrade task
	Allow for the hook callback address and port to be specified via a property
	Add [as a keyboard shortcut for toggling the diff tree
	Stash throttling: Improve experience for user pushing/pulling when number of concurrent git processes have been exceeded.
	Stash returns HTTP 500 and /mvc/error500 on git error during hosting operations
	Use \$STASH_HOME/lib for classpath library directory
	Cannot view a commit that has a vertical bar character username
	HTML returned from incorrect auth details for REST API request
	LifecycleAware component's onStart method is not called after plugin upgrade/enablement

	j and k keyboard shortcuts don't respect "Your unactioned pull requests" filter
	Thread local cache in ActiveObjects causes incorrect branch permissions to be evaluated
	service.bat does not call permgenservice.bat
	Comment tip re:markdown should link to help docs
	Hide "Add Users" from Stash dashboard and Group administration if internal directory is disabled
	Project key is appended to PR link in pull request list
	Cmd/Ctrl+Return doesn't submit comment when in preview

Stash 2.1 changelog

This page will contain information about the Stash 2.1 minor releases as these become available. These releases will be free to all customers with [active Stash software maintenance](#).

Don't have Stash 2.1 yet?





Take a look at all the features in the [Stash 2.1 release notes](#) and see what you are missing out on!




Upgrading from a previous version

If you are upgrading from an earlier version of Stash, please read the [Stash upgrade guide](#).


11 Feb 2013 - Stash 2.1.2 changes

Type	Key	Summary
	STASH-3097	Creating a project fails when uploading an avatar if user is not Admin or Sysadmin
	STASH-3073	415 not correctly returned when unsupported media type is used with REST
	STASH-3070	Pull Request Creation: long commit messages push the JIRA column outside the form in IE
	STASH-3052	Stop duplicate rows in cs_indexer_state

 [Authenticate](#) to retrieve your issues

4 issues

06 Feb 2013 - Stash 2.1.1 changes

Type	Key	Summary
	STASH-3080	Pull request JIRA links won't open

in a new tab



STASH-3076

Regression: Project list pagination is broken due to JavaScript error



STASH-3061

"Tip: You can use markdown" link in PR comment is before the comment submit button in the tab order



Authenticate to retrieve your issues

3 issues

Stash 2.0 release notes

4th December 2012

Meet the Enterprise-ready Stash 2.0 – Powered by Git. Controlled by You.

Imagine all the flexibility of Git with the control needed in the Enterprise. That's Stash 2.0. It comes packed with a heap of features including the highly anticipated branch permissions, @mentions, markdown support and a number of great improvements to pull requests. Additionally Stash 2.0 has been tested in very large environments to ensure you nail the landing of Git in your enterprise.










Try it for FREE ➞

Branch permissions

Per-branch "write" permissions for individuals and groups ensure that stable branches remain stable, and development branches foster collaboration. It's a whole new level of Enterprise security.

Some development workflows require that specific developers oversee merges into the master or release branches, while other developers work on bug-fix and feature branches. Branch Permissions let you turn this "gentlemen's agreement" into a seamless, enforceable process, reducing confusion and time wasted backing out changes that were merged prematurely.

Read more about [branch permissions](#) in Stash.

Branch permissions Add permission		
Branch	Users	Groups
 STASHDEV-2237-diff-scro...	 Adam Ahmed  Jens Schumacher	No groups have been permitted
 master	No users have been permitted	 stash-admins
 release/2.0	 Seb Ruiz	No groups have been permitted

Advanced branch permissions

Advanced Branch Permissions allow you to specify a pattern that is matched against branches and tags being pushed to Stash. This allows you to restrict pushes to multiple branches without the overhead of configuring them individually. Establishing naming conventions based on roles or functional areas (ie, "contractor_" or "userauth_") makes it even easier to set and standardize permissions throughout your organization.

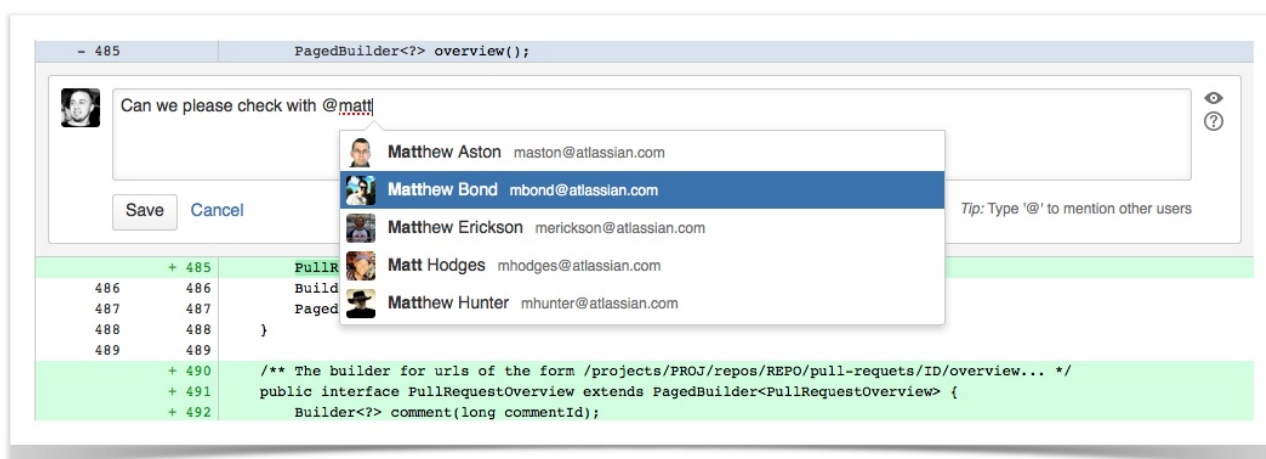
GateKeeper

Branch Permissions can act as a "GateKeeper", allowing you to select a person who is responsible for ensuring that all the code going into production has been properly tested and reviewed. The integration with Pull Requests ensures that only the GateKeepers can merge the changes into the branch you are protecting.

@Mentions

Now you can use 'mentions' to notify another Stash user about the pull request descriptions and comments you are writing. When you mention a user, Stash sends them an email notification, to help streamline communication between your team members.

You can use mentions when writing pull request comments – simply start typing '@' and then the users name, and choose from the list that Stash offers. You can use quotes for unusual names, for example if it has spaces.




Markdown support

Stash now gives you a boost with support for markdown in comments and descriptions. Bring your words to life and get your point across quickly.

- **Emphasize** parts of your comment or create lists to bring your points across.
- Share links to requirements and issues related to the feature you are implementing.
- Provide code examples, formatted just like in your IDE.


- Include screenshots into the discussion for any UI focused features.

Read more about [Markdown support](#) in Stash.


Giancarlo Lionetti
 Seb Ruiz Can you please provide some assistance with this? Looking for

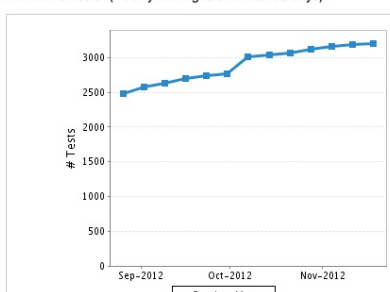
- Test data
- Code improvements

Reply · Edit · Yesterday (edited 7 mins ago)


Seb Ruiz
 Test count is still going up, which is good news.

```
@Test
public void testPullRequests() {
    assertProduces("/projects/FOO/repos/bar/pull-requests", nb().project("FOO").repo("BAR").pullRequests());
    assertProduces("/projects/FOO/repos/bar/pull-requests?state=open", nb().project("FOO").repo("BAR").pullRequests().open());
    assertProduces("/projects/FOO/repos/bar/pull-requests?state=declined", );
}
```

Number of tests (weekly averages over last 90 days):



Connected to <https://bamboo.extranet.atlassian.com>

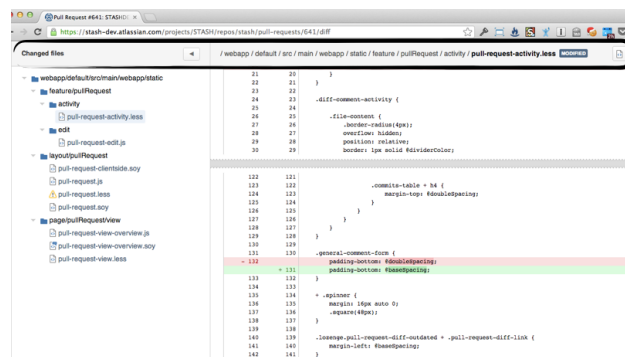
Reply · Edit · Delete · 11 mins ago (edited 1 min ago)

Improved Pull Requests

Beyond @Mentions and Markdown support, Stash 2.0 features a huge number of improvements to Pull Requests.

Improved Diff view

The Diff view in Pull Requests has been engineered to maximise the available screen real-estate when reviewing a Pull Request. The file header and tree navigation now stalk to provide an almost full screen experience without having to switch into a special mode.



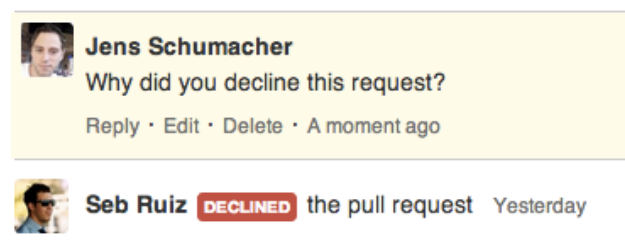
Faster page load times & keyboard shortcuts

The Pull Request tabs have been revamped to make them a whole lot faster to load and with the new keyboard shortcuts (1, 2, 3) it's also a whole lot faster to navigate between the tabs.

Comment indicators

With comment indicators on the filenames, it is now much easier to identify where the discussions are happening when in the Diff view.

New Comments are highlighted



New comments that have been added since your last visit are now highlighted in yellow, making it much easier to follow the conversation on the Pull Request overview.

Read more about [Pull Requests](#) in Stash.

Enterprise licenses

From growing startups to the Fortune 100, Atlassian Enterprise offers products and services that are built to match the needs of the largest enterprise customers. With the new Enterprise tiers for Stash, Atlassian has you covered no matter how big your organisation is. And of course, just like JIRA and Confluence, the Stash Enterprise licenses come with 24x7 Personalized Phone Support.

Try it for FREE ➞

The Stash 2.0 team

Development

Core team

Adam Ahmed
Bryan Turner
Charles O'Farrell
Jason Hinch
Jonathan Poh
Kostya Marchenko
Michael McGlynn
Michael Studman
Pierre-Etienne Poirot
Thomas Bright
Tim Pettersen
Xu-Heng Tjhin

Team leads

Matt Watson
Seb Ruiz

Architect

Michael Heemskerk

Product development lead

Stefan Saasen

Project manager

Anton Mazkvoi

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

Providing feedback:

Please log your [votes and issues](#). They help us decide what needs doing, and are much appreciated!

See the [changelog](#) for Stash 2.0.x minor releases.

Stash 2.0 changelog

This page will contain information about the Stash 2.0 minor releases as these become available. These releases will be free to all customers with [active Stash software maintenance](#).

Don't have Stash 2.0 yet?

Take a look at all the features in the [Stash 2.0 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 upgrade guide](#).


Stash 2.0.3

25 Jan 2013

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2980	Starting a PR's description with a code	Pierre-Etienne Poirot [Atlassian]	Pierre-Etienne Poirot [Atlassian]	↓	 Closed	Fixed	Jan 14, 2013	Feb 27, 2013

		block has the block automatically converted to normal text							
	STASH-2977	Thread local cache in Active Objects causes incorrect branch permissions to be evaluated	Jason Hinch [Atlassian]	Jason Hinch [Atlassian]		 Closed	Fixed	Jan 10, 2013	Feb 27, 2013
	STASH-2966	Should n't supply a response body to Git clients when returning 401 response	Tim Pettersen [Atlassian]	Tim Pettersen [Atlassian]		 Closed	Fixed	Jan 07, 2013	Aug 21, 2013
	STASH-2950	LifecycleAware component's onStart method is not called after plugin upgrade/enabled	Tim Pettersen [Atlassian]	Eli Bishop [Atlassian]		 Closed	Fixed	Oct 31, 2012	Aug 21, 2013













 Authenticate to retrieve your issues

4 issues


Stash 2.0.2

8 Jan 2013

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2930	Cannot view a commit that has a vertical bar character username	Charles O'Farrell [Atlassian]	Kah Loun Foong [Atlassian]	↓	 Closed	Fixed	Dec 17, 2012	Feb 27, 2013
	STASH-2928	Support non-standard location of Bash	Charles O'Farrell [Atlassian]	Michael McGlynn [Atlassian]	↓	 Closed	Fixed	Dec 17, 2012	Feb 27, 2013
	STASH-2869	Cmd/Ctrl+Return doesn't submit comment when in preview	Jonathan Poh [Atlassian]	Matthew Watson [Atlassian]	↓	 Closed	Fixed	Dec 03, 2012	Jan 04, 2013
	STASH-2813	Make the file path in Stash easy to copy	Jonathan Poh [Atlassian]	Alexander Dickson [Atlassian]	↓	 Closed	Fixed	Nov 04, 2012	Jan 04, 2013
	STASH-2808	j and k keyboard shortcuts don't respect "Your una..."	Michael McGlynn [Atlassian]	Wesley Walser [Atlassian]	↓	 Closed	Fixed	Oct 31, 2012	Jan 07, 2013
	STASH-2598	Hide "Add Users" from Stash dashboard and Group administration if	Michael Studman [Atlassian]	William Lovins	↓	 Closed	Fixed	Jun 22, 2012	Jan 07, 2013

internal
director
y is
disable
d











 [Authenticate](#) to retrieve your issues





6 issues


Stash 2.0.1

18 Dec 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2916	service.bat does not call permgen-service.bat	Thomas Bright [Atlassian]	Gurleen Anand [Atlassian]	↑	 Closed	Fixed	Dec 12, 2012	Aug 19, 2013
	STASH-2907	Allow for the hook callback address and port to be specified via a property	Jason Hinch [Atlassian]	Jason Hinch [Atlassian]	↓	 Closed	Fixed	Dec 09, 2012	Feb 27, 2013
	STASH-2891	Hooks path error on Cygwin	Charles O'Farrell [Atlassian]	Alexandre Garnier	↑	 Closed	Fixed	Dec 06, 2012	Feb 27, 2013
	STASH-2881	Project key is appended to PR link in pull request list	Adam Ahmed [Atlassian]	Adam Ahmed [Atlassian]	↓	 Closed	Fixed	Dec 05, 2012	Dec 10, 2012
	STASH-2880	when I view my own profile I get a "back	Charles O'Farrell [Atlassian]	Luis Miranda [Atlassian]	↑	 Closed	Fixed	Dec 05, 2012	Dec 28, 2012

		to users" admin link								
	STASH -2878	Stash fails to start with "Error validati ng Perl: {0}"	Jason Hinch [Atlassi an]	Adam Ahmed [Atlassi an]	↓	 Clos ed	Fixed	Dec 05, 2012	Feb 27, 2013	
	STASH -2866	Adds additio nal logging for the change set indexin g	Pierre- Etienne Poirot [Atlassi an]	Pierre- Etienne Poirot [Atlassi an]	↓	 Clos ed	Fixed	Dec 03, 2012	Dec 07, 2012	

 [Authenticate](#) to retrieve your issues

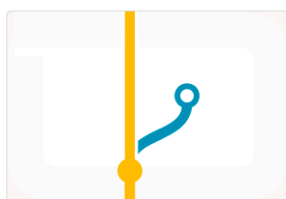
7 issues

Stash 1.3 release notes

9th October 2012

Meet the new, more social Stash – introducing Pull Requests

Pull requests provide your team with a quick and easy way to review code changes made on a branch, discuss those changes, and make further modifications before the branch is merged back to master or your main development branch.



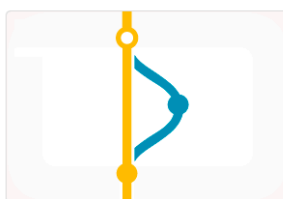
Branch

Develop features on a branch and create a pull request to get changes reviewed.



Discuss

Discuss and approve code changes related to the pull request.



Merge

Merge the branch with the click of a button.

Try it for FREE ➞

Pull requests

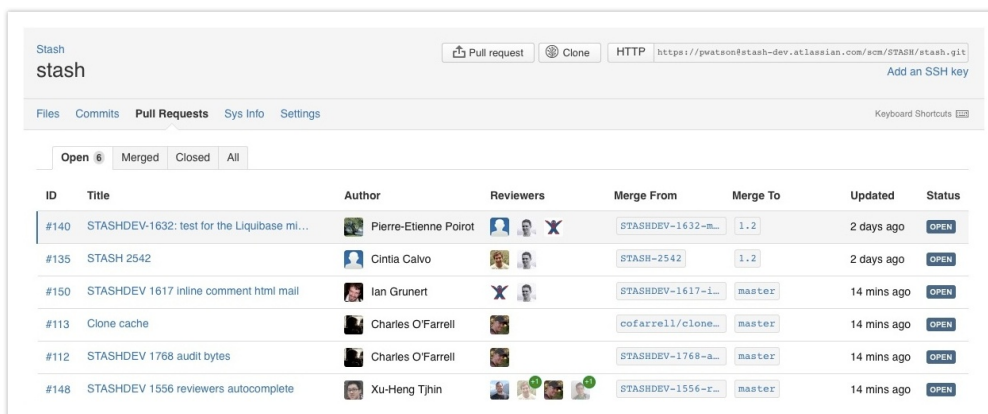
With Stash and Pull Requests, code reviews become an integral part of your development process. Development happens on branches and when code is ready to be merged into the main branch a Pull Request

is opened. Unless the code has been reviewed as part of a Pull Request, it does not get integrated back into the main branch. All the benefits of code review baked right into your workflow!

Creating a pull request is like starting a discussion. Your reviewers can see the changes you have made, comment on those changes and commit further changes and improvements to the branch if required. When everyone agrees, the branch can then be merged back to master or your main development branch.

Getting your code reviewed has never been easier – simply click the **Pull Request** button in the repository header, select the branch you've been working on, the branch you want to merge to, then add a short description and you're done.

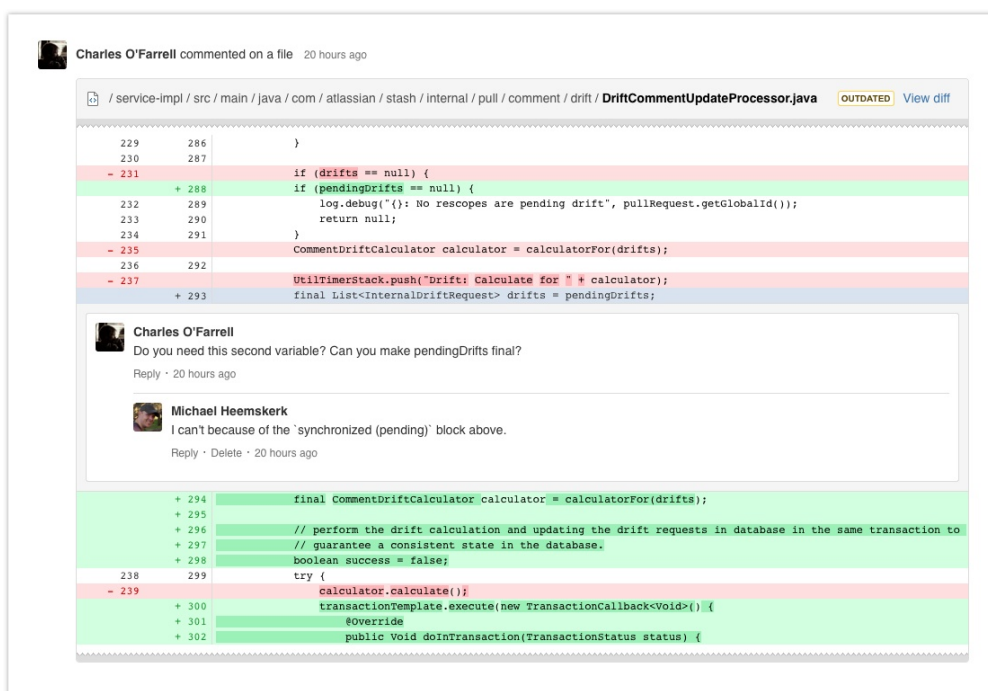
Read more about [using pull requests](#) in Stash.



ID	Title	Author	Reviewers	Merge From	Merge To	Updated	Status
#140	STASHDEV-1632: test for the Liquibase mi...	Pierre-Etienne Poirot		STASHDEV-1632-m...	1.2	2 days ago	OPEN
#135	STASH-2542	Cintia Calvo		STASH-2542	1.2	2 days ago	OPEN
#150	STASHDEV-1617 inline comment html mail	Ian Grunert		STASHDEV-1617-i...	master	14 mins ago	OPEN
#113	Clone cache	Charles O'Farrell		cofarrell/clone...	master	14 mins ago	OPEN
#112	STASHDEV-1768 audit bytes	Charles O'Farrell		STASHDEV-1768-a...	master	14 mins ago	OPEN
#148	STASHDEV-1556 reviewers autocomplete	Xu-Heng Tjhin		STASHDEV-1556-r...	master	14 mins ago	OPEN

Discussions

The essential thing about a pull request is the discussion that takes place around the code changes you are making. The overview captures the entire activity of the pull request. Comments on the diff, replies, or new commits to the branch.



Charles O'Farrell commented on a file 20 hours ago

/ service-impl / src / main / java / com / atlassian / stash / internal / pull / comment / drift / **DriftCommentUpdateProcessor.java** OUTDATED View diff

```

229      286      }
230      287
- 231      288      if (drifts == null) {
+ 232      289      if (pendingDrifts == null) {
233      290          log.debug("{}: No rescopes are pending drift", pullRequest.getGlobalId());
234      291          return null;
- 235      292      }
+ 236      293      CommentDriftCalculator calculator = calculatorFor(drifts);
- 237      294      UtilTimerStack.push("Drift: Calculate for " + calculator);
+ 238      295      final List<InternalDriftRequest> drifts = pendingDrifts;

```

Charles O'Farrell
Do you need this second variable? Can you make pendingDrifts final?
Reply · 20 hours ago

Michael Heemskerk
I can't because of the 'synchronized (pending)' block above.
Reply · Delete · 20 hours ago

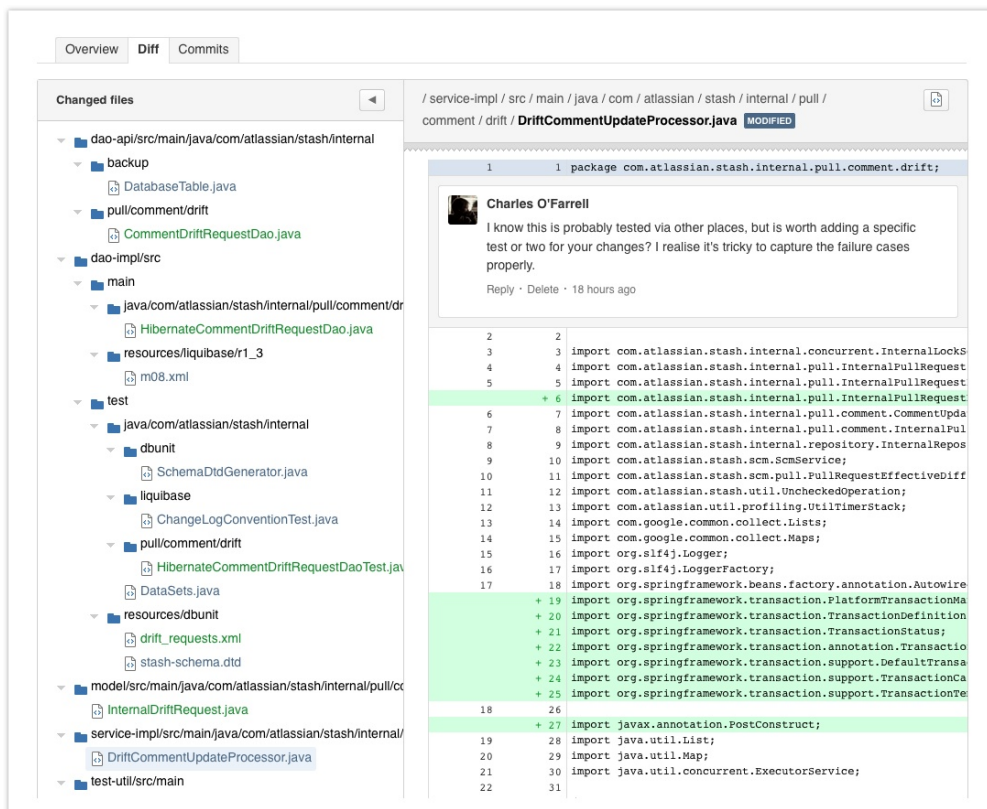
```

+ 294      final CommentDriftCalculator calculator = calculatorFor(drifts);
+ 295
+ 296      // perform the drift calculation and updating the drift requests in database in the same transaction to
+ 297      // guarantee a consistent state in the database.
+ 298      boolean success = false;
238      299      try {
- 239      300          calculator.calculate();
+ 301      transactionTemplate.execute(new TransactionCallback<Void>() {
+ 302          @Override
+ 303          public Void doInTransaction(TransactionStatus status) {

```

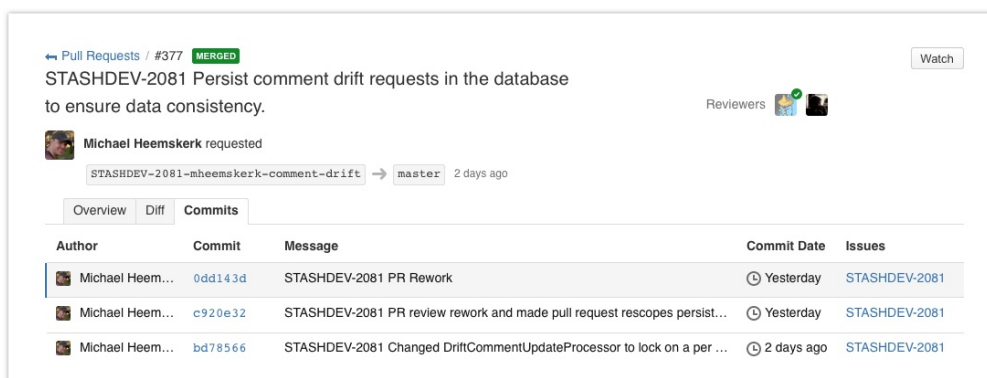
Changed Files

Diffs for a Stash pull request highlight the changes that would result from the merge. The diff tree shows all modified files and, as you'd expect, you can click on any file to see exactly what has been added, deleted or modified. Threaded comments right in the diff allow meaningful and contextual conversations about your code.



Commits

A pull request is dynamic! Not only can there be a lively discussion about code changes, but participants can commit new changes to the branch. Stash auto-updates the **Commits** tab of the pull request, so you can see exactly which commits will be merged. Stash is smart about comments, moving them along when lines are added or removed. If a line with a comment gets removed, you can still view the comment in the activity, but Stash marks the diff as *outdated* to let you know that this piece of code has been changed in recent commits.



Notifications

Whether someone added you as reviewer, commented on the pull request or merged your changes, Stash ensures you know what's going on by sending you email notifications about pull requests relevant to you.

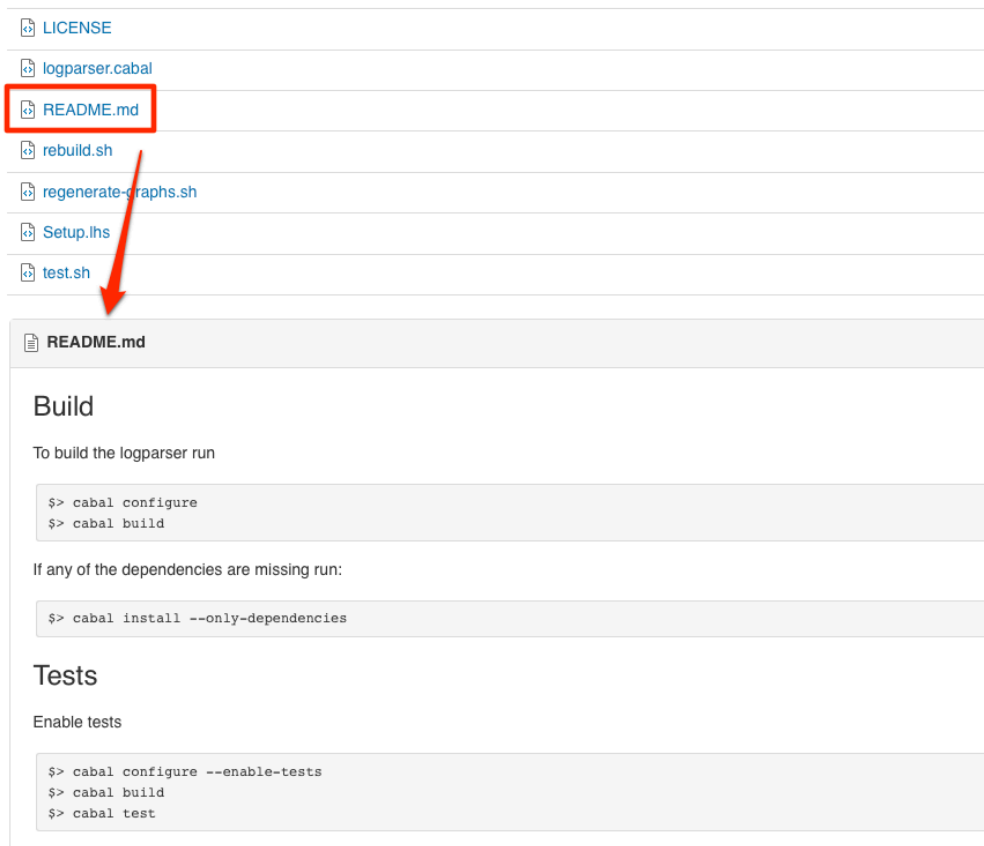
Keyboard shortcuts

Stash has an evolving set of keyboard shortcuts, to help you work faster. Click the link when looking at any repository to refresh your memory about the available shortcuts.

Keyboard Shortcuts	
Global	
Close dialog	ESC
Show this dialog	?
Recent repositories	G then R
Changeset	
Next file	J
Previous file	K
Hide/show the file tree	T
Pull Request List	
Next pull request	J
Previous pull request	K
Open pull request	RETURN or O
Highlight your pull requests	T
Directory Browsing	
Next file/directory	J
Previous file/directory	K
Open file/directory	RETURN or O
Move up a directory	U
Find files	F
Within A Repository	
Change branch/tag	B
Commit List	
Next changeset	J
Previous changeset	K
Open changeset	RETURN or O
Hide/show merges	T
Within A Pull Request	
Post comment	% RETURN
+1 pull request	1
Add a general comment	M
Source View	
Move to containing directory	U

README – simple project documentation

Stash now provides an appealing, yet simple, way to document the project right in the repository by rendering the content of .md and .txt files in the file view of the repository. If the file uses [Markdown](#), that gets rendered straight to the screen.



Try it for FREE ➞

The Stash 1.3 team

Development

Core team

Adam Ahmed
 Bryan Turner
 Federico Silva Armas
 Ian Grunert
 Jason Hinch
 Jonathan Poh
 Kostya Marchenko
 Michael McGlynn
 Michael Studman
 Pierre-Etienne Poirot
 Thomas Bright
 Tim Pettersen
 Xu-Heng Tjhin

Team leads

Matt Watson
 Seb Ruiz

Architect

Michael Heemskerk

Product Development Lead

Stefan Saasen

Project manager

Anton Mazkovo

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

Providing feedback:

Please log your [votes](#) and [issues](#). They help us decide what needs doing, and are much appreciated!

See the [change log](#) for Stash 1.3.x minor releases.

Stash 1.3 changelog

This page will contain information about the Stash 1.3 minor releases as these become available. These releases will be free to all customers with [active Stash software maintenance](#).

Don't have Stash 1.3 yet?

Take a look at all the features in the [Stash 1.3 release notes](#) and see what you are missing out on!











A green rectangular button with rounded corners. On the left is a white download icon (a square with a downward arrow). To the right of the icon is the word "Download" in white text.***Upgrading from a previous version of Stash***

If you are upgrading, please read the [Stash upgrade guide](#).

Stash 1.3.1

7 Nov 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-3231	JIRA indexing does not pick up commits for a couple of hours.	Unassigned	Michael Heems kerk [Atlassian]	↑	 Closed	Fixed	Mar 14, 2013	Aug 21, 2013
	STASH-2815	Use the server-side constant for page.max.changes on the client-side	Adam Ahmed [Atlassian]	Adam Ahmed [Atlassian]	↓	 Closed	Fixed	Nov 07, 2012	Nov 07, 2012
	STASH-2794	Invalid commit command on the "How to manually merge a pull request" dialog	Michael Heems kerk [Atlassian]	Charles Thomas	↓	 Closed	Fixed	Oct 26, 2012	Nov 07, 2012
	STASH-2777	Repository Settings section is not showing the general settings tab as active	Jason Hinch [Atlassian]	Jason Hinch [Atlassian]	↓	 Closed	Fixed	Oct 19, 2012	Oct 22, 2012
	STASH-2762	SAL UpgradeTask framework doesn't work in Stash	Tim Pettersen [Atlassian]	Tim Pettersen [Atlassian]	↑	 Closed	Fixed	Oct 16, 2012	Feb 27, 2013

	STASH -2757	Plugins cannot set active tab in repository settings	Seb Ruiz [Atlassian]	Seb Ruiz [Atlassian]	↓	 Closed	Fixed	Oct 11, 2012	Oct 22, 2012
	STASH -2542	When starting the server display the server URL at the bottom of the trace	Cintia Calvo [Atlassian]	Sten Pittet [Atlassian]	↓	 Closed	Fixed	May 22, 2012	Feb 27, 2013

 [Authenticate](#) to retrieve your issues

7 issues

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

 [Download](#)

Providing feedback:

Please log your [votes](#) and [issues](#). They help us decide what needs doing, and are much appreciated!



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...](#)

Database Migration

The database migration is currently underway and shouldn't take long. If you cancel the migration, Stash will rollback to the previous database.



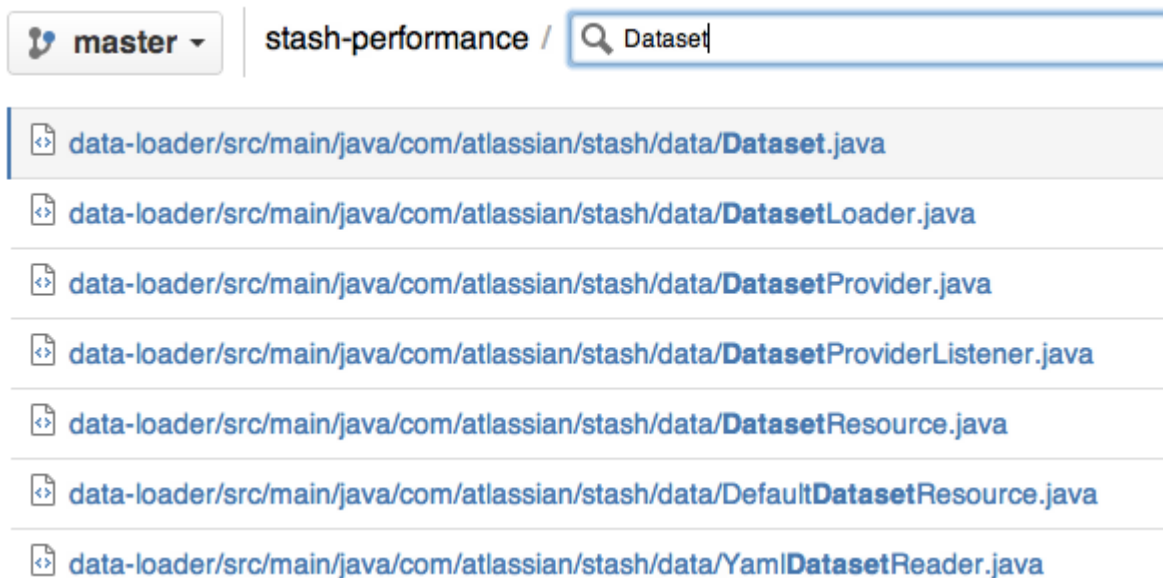
Executing changeset 3 of 30, containing 165,828 changes with id 1343362915704-37

Cancel Migration

3

File search

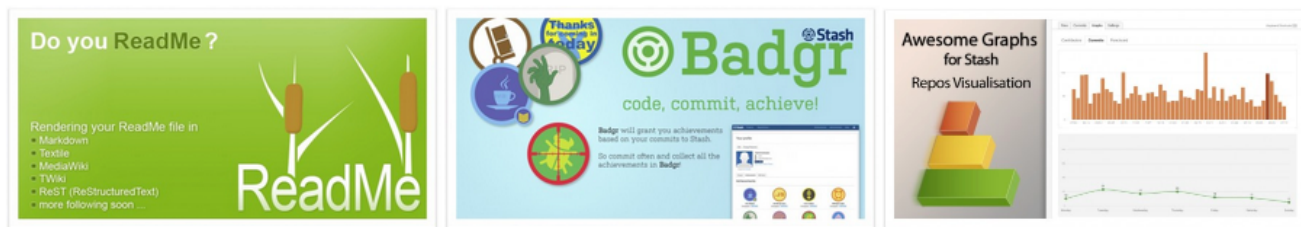
Stash's new file search ensures that you can quickly find any file in your repository, without needing to check out the source. Just start typing any part of the file name into the search field and you'll get a list of matches, fast. And you can filter by path, CamelCase (for example, AttrM to match AttributeMap) and file extension.



4

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.











5

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.

 master ▾

Author	Commit	Message
 Xu-Heng Tjhin	b7e9041	STASHDEV-1451 - Removed 'add comment' button for threaded
 Michael Hee...	e78800e	Upgraded Spring from 3.1.1 to 3.1.2
 Michael Hee...	c2508c4	NONE: Fix for error on PR creation when there are too many con
 Michael Hee...	71400be M	Merge remote-tracking branch 'origin/1.3-pull-request-review'
 Michael Hee...	60ab0b0 M	Merge branch '1.3-pull-request-review' Conflicts: api/src/main/jav
 Tim Pettersen	bca3e01 M	Merge branch 'master' of ssh://stash-dev.private.atlassian.com:7
 Tim Pettersen	d12a681	STASHDEV-1335: fix up batshit insane PostgresTypeConverter c
 Michael Hee...	48b0ec0	Fix broken tests

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 Mazkovoï

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!

 [Download](#)




Upgrading from a previous version of Stash

If you are upgrading, please read the [Stash upgrade guide](#).


Stash 1.2.4

20 September 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH	Stash	Unassi	Matthe		 Clos	Fixed	Sep 20,	Sep 20,

-2719	on Windows occasionally kills processes or causes BSOD on VMs	igned	w Watson [Atlassian]	ed	2012	2012
-------	---	-------	----------------------------	----	------	------





 [Authenticate](#) to retrieve your issues


1 issues

Stash 1.2.3

14 September 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2708	Mixed case usernames sometimes counted twice for licensing purposes	Tim Pettersen [Atlassian]	Tim Pettersen [Atlassian]	↑	 Closed	Fixed	Sep 13, 2012	Feb 27, 2013
	STASH-2703	Foreign key recreation fails on mysql when upgrading from 1.1 to 1.2+	Tim Pettersen [Atlassian]	Tim Pettersen [Atlassian]	↓	 Closed	Fixed	Sep 11, 2012	Feb 27, 2013







 [Authenticate](#) to retrieve your issues


2 issues

Stash 1.2.2

22 August 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2682	Stash gets an error when sending a support zip email	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]	↓	 Closed	Fixed	Aug 22, 2012	Oct 01, 2012
	STASH-2536	msysgit installed on Windows, still get "git was not found on the PATH for Stash"	Pierre-Etienne Poirot [Atlassian]	Rene Verschuur [Atlassian]	↓	 Closed	Fixed	May 18, 2012	Aug 21, 2012
	STASH-2522	Difficult to fix PATH when installing git on startup	Pierre-Etienne Poirot [Atlassian]	Matthew Watson [Atlassian]	↑	 Closed	Fixed	May 14, 2012	Aug 21, 2012



 Authenticate to retrieve your issues


3 issues

Stash 1.2.1

17 August 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2668	Upgrade to version 1.2 on PostgreSQL corrupts Trusted Application links with Stash	David Pinn [Atlassian]	David Pinn [Atlassian]	↑	 Closed	Fixed	Aug 10, 2012	Feb 27, 2013

 [Authenticate](#) to retrieve your issues

1 issues

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](#)
- [Image diffs](#)

 [Download](#)

Providing feedback:

Please log your [votes](#) and [issues](#). They help us decide what needs doing, and are much appreciated!

1

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...](#)

Groups SSH keys

2

Fast browsing

SSH keys

[Add Key](#)

Label	Key
jens.schumacher@example.com	ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDA6lF53Z6LfrHEor1NCTI9PNxFu36ljvAcXzv1LifqEcUi...

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.

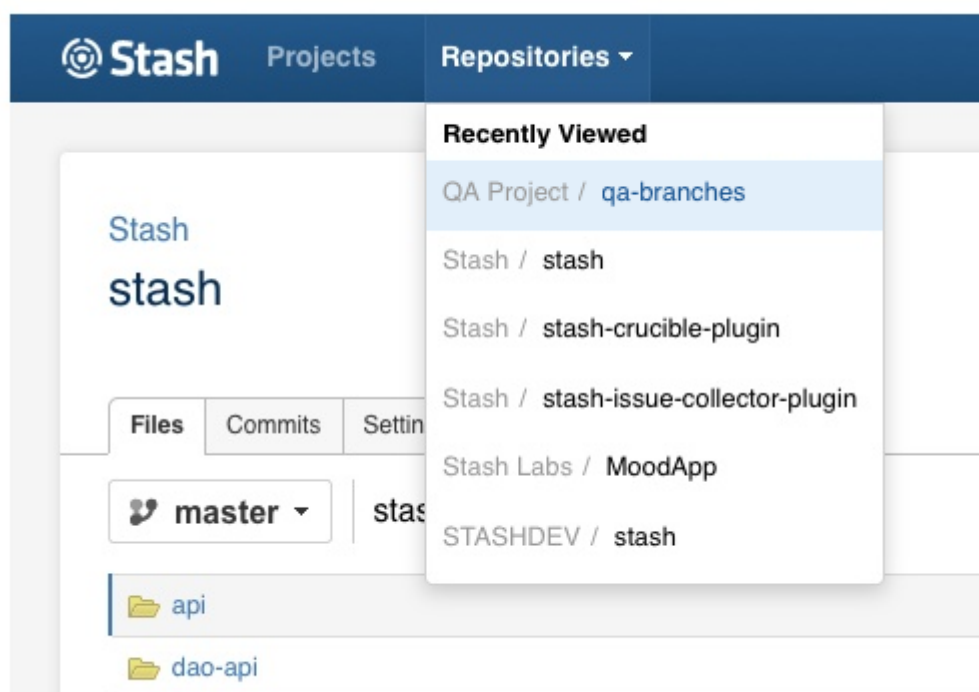
Mouse-less productivity

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.

3

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...](#)

4

Image diffs

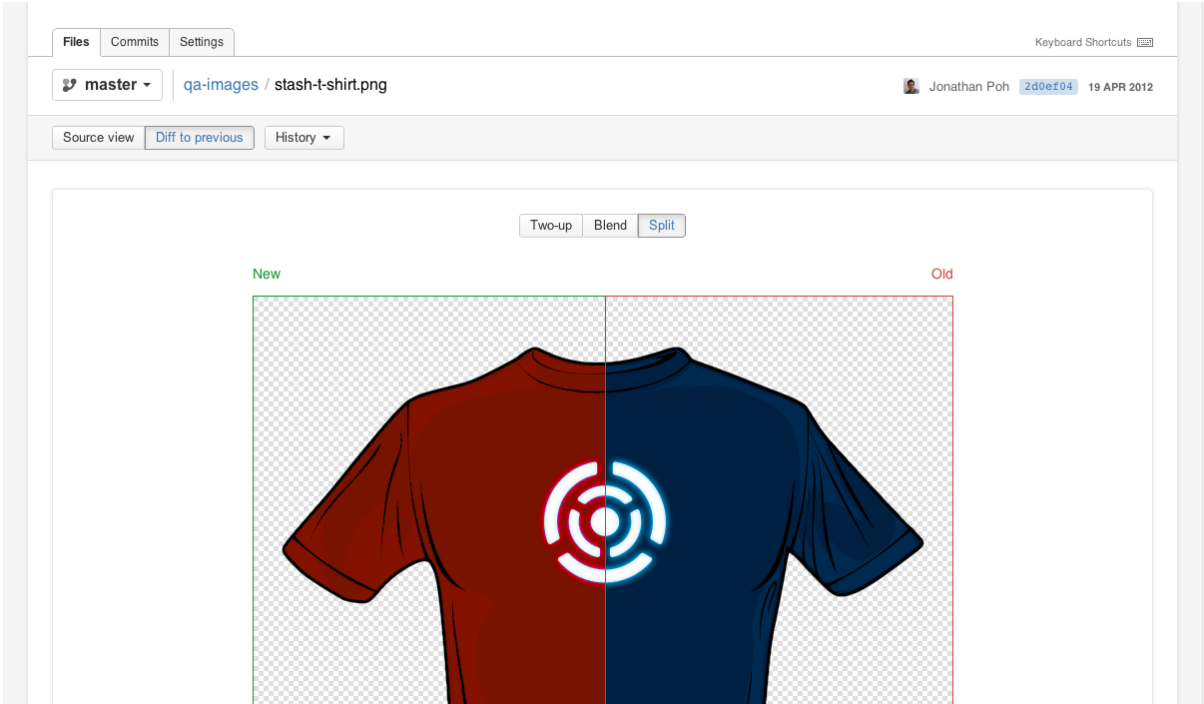
Global Permissions

Individual Users	Add Users	System Administrator	Administrator	Project Creator	Stash User
 Adam Ahmed		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Administrator		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Anton Mazkvoi		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Extranet Bamboo User		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 David Pinn		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Extranet Crucible User		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
 Federico Silva Armas		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Geoff Crain		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 Giuseppe Liotti		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

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 removed between two revisions.



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 Mazkovoï

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 1.1 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.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2676	Persistent Cross Site Scripting Vulnerability	Vitaly Osipov [Atlassian]	Vitaly Osipov [Atlassian]	↑	 Closed	Fixed	Aug 20, 2012	Oct 27, 2013
	STASH-2630	Unable to delete Application Link in Stash	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]	↑	 Closed	Fixed	Jul 05, 2012	Feb 27, 2013
	STASH-2627	Admin pages are including web resources with context "atl.general" instead of "atl.admin".	Sebastian Ruiz [Atlassian]	Eli Bishop [Atlassian]	↑	 Closed	Fixed	Jul 02, 2012	Jul 02, 2012
	STASH-2624	JiraKey Indexer doesn't take system property into account	Unassigned	Stefan Kohler	↑	 Closed	Fixed	Jun 28, 2012	Jun 28, 2012
	STASH-2486	Upgrade UPM to support Atlassian Marketplace plugins	Matthew Watson [Atlassian]	Stefan Kohler	↑	 Closed	Fixed	May 02, 2012	Feb 27, 2013





 [Authenticate](#) to retrieve your issues


5 issues

Stash 1.1.1

22 June 2012

This is a bug fix release. The issues addressed in this release of Stash are shown below.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2592	Cannot clone repositories with '.' in name over HTTP	Tim Pettersen [Atlassian]	Jan-Heinrik Heuing	↑	 Closed	Fixed	Jun 21, 2012	Jun 22, 2012
	STASH-2580	STASH_HOME not defined when using Stash as a service	Michael Heemscker [Atlassian]	Alexandre Garnier	↓	 Closed	Fixed	Jun 15, 2012	Oct 04, 2012

 [Authenticate](#) to retrieve your issues

2 issues

Stash 1.0 release notes

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](#)

 [Download](#)

Providing feedback:

Please log your [votes](#) and [issues](#) . They help us decide what needs doing, and are much appreciated!

1

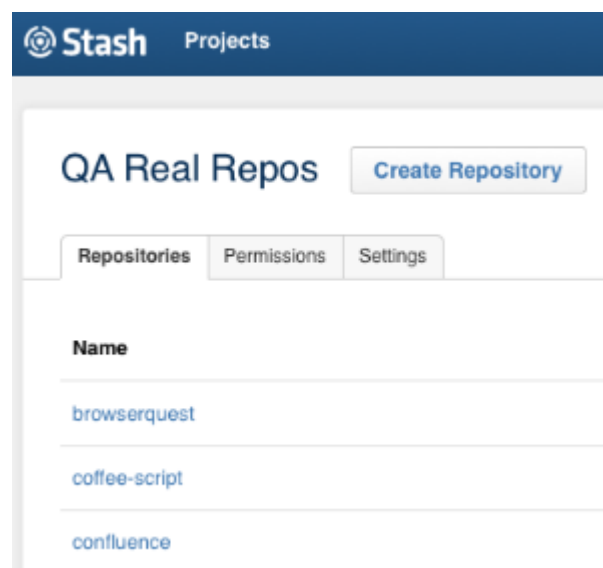
Git repository management

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.

3

Built for Git, focused on Enterprise

Project Administrator Can administer the project and create new repositories. Administrators have complete access to all repositories in the project.	Users <div>Add Users</div> <div> Sebastian Ruiz sruiz </div>	Groups <div>Add Groups</div> <div>atlassian-dev-tools</div>
Contributor Can clone, pull from and push to all repositories in the project.	Users <div>Add Users</div> <div> Giancarlo Lionetti glionetti Jeff Park jpark </div>	Groups <div>Add Groups</div> <div>stash-bamboo-users</div>
Observer Have full read access to all repositories in the project. They can clone and pull from repositories, but cannot push commits back.	Users <div>Add Users</div> <div> Extranet Crucible User eacrcru Federico Silva Armas farmas </div>	Groups <div>Add Groups</div> <div>aaa accounts-payable</div>

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.







4

Browse your source and history

User Directories

The table below shows the user directories currently configured for Stash.

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 in the first directory where Stash has permission to make changes. It is recommended that each user exist only in a single directory.

Directory Name	Type	Order
Stash Internal Directory	Internal	 
Extranet Crowd <small>You cannot edit this directory because you are logged in through it, please log in as a locally authenticating user to edit it.</small>	Atlassian Crowd	 
caviar-internal-directory <i>(inactive)</i>	Internal	 
<div>Add Directory</div>		

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.

5

JIRA integration

Stash


Clone

git clone https://gilonetti@stash-dev.atlassian.com/git/STASH/stash

Files

Commits

Settings



Adam Ahmed

18 APR 2012

Merge branch 'STASH-2408' into 1.0

80f1916

MERGE

Parents: [925b2d1](#) + [6fd2959](#)

Issues: STASH-2408

Showing diff to

925b2d1

Changed files

/webapp/default/src/main/webapp/static/page/users/profile.soy **MODIFIED**

View source

webapp/default/src/main/webapp/static/page/users

profile.soy

8

8

*/

9

9

{template .profile}

10

10

{webResourceManager.requireResourcesForContext('internal.page.userProfile')}

+ 11

+ 11

{webResourceManager.requireResourcesForContext('atl.userprofile')}

11

12

{call stash.layout.entity

12

13

{param windowTitle: cav_118n('stash.users.profile.window.title', 'Profile') /}

13

14

{param entityName: cav_118n('stash.users.profile.heading', 'Your profile') /}

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.

All

Comments

Work Log


History

Activity

Source

Reviews

Builds



Pierre-Etienne Poirot

Sunday 9:29 PM

Merge branch '[STASH-2442](#)-inexistent-repo-and-authentication' into 1.0

View full commit

Stash / stash

MODIFY

scm-common/src/main/java/com/atlassian/stash/web/ManagedRepositoryServlet.java

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 Mazkvoi

Support

Ajay Sridhar
Armen Khachatryan
Daniel Rohan
Douglas Fabretti
Felipe Kraemer
Gurleen Anand
Renan Battaglin
Rene Verschoor
Zed Yap

Others**Product management**

Jens Schumacher
Sten Pittet

Design and user experience

Jake Causby
Matt Bond

Product marketing

Giancarlo Lionetti
Jeff Park

Technical writing

Paul Watson

Operations

James Fleming

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.

Type	Key	Summary	Assignee	Reporter	Priority	Status	Resolution	Created	Updated
	STASH-2523	NOTICE file in root of distribution is confusing	Matthew Watson [Atlassian]	Matthew Watson [Atlassian]	↑	 Closed	Fixed	May 14, 2012	May 16, 2012
	STASH-2509	MD5 hash of email for Gravatar needs to use a lower-case version of email addresses	Jason Hinch [Atlassian]	Mary Wholey	↓	 Closed	Fixed	May 09, 2012	May 16, 2012
	STASH-2505	Outgoing Authentication panel in linked product fails to appear	Jason Hinch [Atlassian]	David Pinn [Atlassian]	↑	 Closed	Fixed	May 08, 2012	Oct 12, 2012
	STASH-2494	Unable to create	Jason Hinch [Atlassian]	Jason Hinch [Atlassian]	↑	 Closed	Fixed	May 04, 2012	Oct 11, 2012

Trusted
Applica
tion in
Postgre
s

an] an]

 **Authenticate** to retrieve your issues

4 issues









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.

Type	Key	Summ ary	Assign ee	Report er	Priorit y	Status	Resolu tion	Create d	Update d
	STASH -2501	Source Tree plugin fails to load on decorat or pages	Seb Ruiz [Atlassi an]	Seb Ruiz [Atlassi an]	↓	 Clos ed	Fixed	May 07, 2012	Feb 27, 2013
	STASH -2500	Improv e scalabil ity by queuei ng of scm hosting operati ons under high load	Michael Heems kerk [Atlassi an]	Michael Heems kerk [Atlassi an]	↑	 Clos ed	Fixed	May 07, 2012	May 07, 2012
	STASH -2499	Termin ate git proces ses when HTTP clients disconn ect	Michael Heems kerk [Atlassi an]	Michael Heems kerk [Atlassi an]	↑	 Clos ed	Fixed	May 07, 2012	May 07, 2012
	STASH -2498	Display useful error messa ges for clients	Bryan Turner [Atlassi an]	Bryan Turner [Atlassi an]	↑	 Clos ed	Fixed	May 07, 2012	Jul 26, 2013

		when git comma nds fail over HTTP							
	STASH -2478	Invalid Charac terExce ption in JIRA logs when applicat ion link to Stash is persent	Tim Petters en [Atlassi an]	Tim Petters en [Atlassi an]		 Clos ed	Fixed	May 01, 2012	Feb 27, 2013

 [Authenticate](#) to retrieve your issues

5 issues

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!




 [Download](#)













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.

Type	Key	Summ ary	Assign ee	Report er	Priorit y	Status	Resolu tion	Create d	Update d
	STASH -2501	Source Tree plugin fails to load on decorat or pages	Seb Ruiz [Atlassi an]	Seb Ruiz [Atlassi an]		 Clos ed	Fixed	May 07, 2012	Feb 27, 2013

	STASH-2500	Improve scalability by queuing of scm hosting operations under high load	Michael Heems kerk [Atlassian]	Michael Heems kerk [Atlassian]		 Closed	Fixed	May 07, 2012	May 07, 2012
	STASH-2499	Terminate git processes when HTTP clients disconnect	Michael Heems kerk [Atlassian]	Michael Heems kerk [Atlassian]		 Closed	Fixed	May 07, 2012	May 07, 2012
	STASH-2498	Display useful error messages for clients when git commands fail over HTTP	Bryan Turner [Atlassian]	Bryan Turner [Atlassian]		 Closed	Fixed	May 07, 2012	Jul 26, 2013
	STASH-2478	Invalid CharacterException in JIRA logs when application link to Stash is present	Tim Pettersen [Atlassian]	Tim Pettersen [Atlassian]		 Closed	Fixed	May 01, 2012	Feb 27, 2013

 [Authenticate](#) to retrieve your issues

5 issues

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 Stash 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

- [Stash security advisory 2012-09-04](#)

Stash security advisory 2012-09-04

This advisory discloses a security vulnerability that we have found in Stash and fixed in Stash 1.1.2.

Customers who have downloaded and installed Stash should upgrade their existing Stash installations to fix this vulnerability.

Atlassian is committed to improving product security. The vulnerability listed in this advisory has been discovered by Atlassian, unless noted otherwise. The reporter may also have requested that we do not credit them.

If you have questions or concerns regarding this advisory, please raise a support request at <http://support.atlassian.com/>.

In this advisory:

- [XSS Vulnerability](#)

XSS Vulnerability

Severity



Atlassian rates the severity level of this vulnerability as **High**, according to the scale published in [Severity Levels for Security Issues](#). The scale allows us to rank the severity as critical, high, medium or low.

This is an independent assessment and you should evaluate its applicability to your own IT environment. This vulnerability is **not** of Critical severity.

Description

We have identified and fixed a persistent cross-site scripting (XSS) vulnerability that affects Stash instances, including publicly available instances (that is, Internet-facing servers). XSS vulnerabilities allow an attacker to embed their own JavaScript into a Stash page.

You can read more about XSS attacks at cgisecurity.com, [The Web Application Security Consortium](http://www.wapsec.org) and other places on the web.

This vulnerability affects all supported versions of Stash, and has been fixed in Stash 1.1.2. This issue can be tracked here:  **STASH-2676** - Persistent Cross Site Scripting Vulnerability ( **Closed**)

Risk Mitigation

We strongly recommend upgrading your Stash installation to fix this vulnerability. Please see the 'Fix' section below.

Fix

Upgrade

The vulnerability and fix version are described in the 'Description' section above.

We recommend that you upgrade to the latest version of Stash, if possible. For a full description of the latest version of Stash, see the [release notes](#). You can download the latest version of Stash from the [download centre](#).

Patches are not available for this vulnerability.

Git resources

Get Git

Mac: <http://code.google.com/p/git-osx-installer/downloads/list?can=3>

Linux: http://book.git-scm.com/2_installing_git.html

Ubuntu Linux: <https://launchpad.net/~git-core/+archive/ppa>

Windows: [Full installer for official Git for Windows](#)

On this page:

- [Get Git](#)
- [Learning Git](#)
- [Getting started](#)
- [Git cheat sheets and other resources](#)
- [Git .mailmap](#)

Learning Git

[Git Tutorials and Training](#)

[Basic Git commands](#)

Getting started

One "gotcha" when starting with Git is the way in which it pushes branches by default. On older versions of Git, pushing without arguments would push *all* branches that have the same name both locally and remotely. This can result in unexpected behaviour if you have old branches that complain when the remote branch is updated. It can even be quite dangerous if you do a force push and it reverts changes on the server. You can see the current value by running:

```
git config push.default
```

If this value is blank or 'matching', it is our recommendation that you reconfigure it to use 'upstream'.

```
git config --global push.default upstream
```

There has been some [discussion](#) around changing the default behaviour of Git.

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://zrusin.blogspot.com.au/2007/09/git-cheat-sheet.html>

<http://ndpsoftware.com/git-cheatsheet.html#loc=workspace;>

<http://blog.fournova.com/2011/06/git-cheat-sheet/>

<http://jan-krueger.net/development/git-cheat-sheet-extended-edition>

Git .mailmap

The Git `.mailmap` feature is useful locally, and in Stash repositories, to map multiple commit identities to the one Stash user – this can be used to tidy up your Git histories.

Incidentally, `.mailmap` can also help to manage Stash licenses - as described in [this KB article](#).

The [Git documentation](#) for `.mailmap` has configuration details (see the "MAPPING AUTHORS" section).

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 integrate with any other Atlassian Tools? Crowd? Bitbucket? SourceTree?](#)
 - [Q: Will Stash be available for Atlassian OnDemand?](#)
- **Licensing**
 - [Q: Does my Stash license have to match the number of users in my external directory \(LDAP, Active Directory, Crowd or JIRA\)?](#)

Related pages:

- [Stash Knowledge Base Home](#)
- [Support policies](#)

Child pages:

- [How do I change the external database password](#)
- [Stash home directory](#)
- [Raising a request with Atlassian Support](#)
- [Support policies](#)
- [Building Stash from source](#)
- [Contributing to the Stash documentation](#)

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.

```
git config http.postBuffer 524288000
```

See [Git push fails with 'fatal: The remote end hung up unexpectedly'](#).

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](#) - Include Mercurial (Hg) support in Stash ( [Open](#))

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.

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: Does my Stash license have to match the number of users in my external directory (LDAP, Active Directory, Crowd or JIRA)?

A: No. You can control which users in your external directory have access to Stash, so that the license limit is not exceeded. A user is by [definition](#) any account that has permission to log into the Stash application. If you synchronise Stash with an external user directory, you can grant access to Stash to a subset of users, so as to stay below your license limit. The [Global permissions](#) page explains in detail how to manage login rights for users and groups in Stash.

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](#).


Stash home directory

What 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.

 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.

 Where possible, you should choose a location for your Stash home directory that will *never* need to be moved. Some home contents are location-sensitive, so moving the home directory may corrupt them. Stash attempts to update contents when it detects that the home directory has moved, but the safest approach is to

avoid the issue altogether by leaving the home directory in the same location.

What is in the Stash home directory?

Your Stash home directory contains the following directories and files:

Path	Description
<code>caches</code>	Contains cache and index files. It should be safe for these files to be deleted between application restarts; however, these files must not be modified or removed externally while Stash is running.
<code>config</code>	Contains application configuration.
<code>data</code>	Contains the Git repositories, project avatars, and the embedded HSQL database if an external database is not configured.
<code>export</code>	Contains dump files produced during database migrations.
<code>lib</code>	As of Stash 2.1, can contain third-party jars such as the MySQL JDBC driver.
<code>lib/native</code>	As of Stash 2.1, can contain <i>native libraries</i> , such as Tomcat's APR-based native library.
<code>log</code>	Contains log files for Stash.
<code>plugins</code>	Contains plugin related data (such as externally uploaded plugins) for Stash.
<code>tmp</code>	Contains temporary files created by the system. Its contents can safely be deleted when Stash is <i>not running</i> .
<code>stash-config.properties</code>	Allows configuring various aspects of how Stash behaves, such as its database connection pool size and the location of the Git binary to use. This file will be created automatically during a database migration. It can be created manually otherwise. See Stash config properties for further information.

Securing the Stash home directory

The internal database files, the migration dump files and `stash-config.properties` 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, and that this user has read, write and execute permissions, by setting file system permissions appropriately for your operating system.

About the repositories

As noted above, `data` contains the Git repositories being managed by Stash, where "managed by Stash" are

the operative words. The repositories are for *Stash* to interact with, and they are configured and managed accordingly. They are *not* a mechanism for configuring Stash behaviour. We *strongly* recommend that customers never modify them, nor interact with them directly. They are *intentionally* structured in a way which does not lend itself well to direct interaction.

Being Git repositories, there are certainly standard aspects to how the repositories on disk are stored and how they function. However, the exact way they are configured *can and does* change between Stash releases. Stash makes *no effort* to preserve unexpected configuration changes which have been applied by customers, and such changes may cause failures at runtime or during upgrades. If there is an aspect of Stash's behaviour you wish to configure, please open a feature request on jira.atlassian.com rather than trying to modify the repositories directly.



Repositories are *location sensitive*. Moving your Stash home directory will result in the system being locked (briefly) on startup while Stash updates the repositories on disk. Assuming the updates are applied successfully, the system will then unlock itself for normal usage.

Where possible, please choose a Stash home location which will not need to be changed later.

Raising a request with Atlassian Support

If you encounter any problems when setting up or using Stash, please let us know — we're here to help!

You may want to search the following first:

- the [Atlassian Answers site](#) (the Stash forum), where Atlassian staff and Stash users can answer your questions.
- the [Stash Knowledge Base](#).

If you've found a bug in Stash, or want to request a feature or improvement, raise a ticket in the Stash project of our [public issue tracker](#). Try searching for similar issues - voting for an existing issue is quicker, and avoids duplicates.

If you still need assistance, please raise a support request, either from within Stash or on the Atlassian Support site, as described in the following sections.

Providing as much information as possible about your Stash installation with your initial request will help our Support Engineers to give you a faster and more complete response.

On this page:

- [Raising a Support Request from within Stash](#)
- [Raising a Support request yourself at Atlassian Support](#)
- [Information you should provide](#)

Raising a Support Request from within Stash

This method depends on having a [mail server configured for Stash](#) that supports large zip file attachments.

1. Log in to Stash (as a [System Administrator](#)) and go to the [admin area](#).
2. Click **Atlassian Support Tools** (under 'Support') then **Support Request**.
3. Provide as much information as possible in the Description, including steps to replicate the problem, and any error messages that are appearing on the console or in the [logs](#). For performance issues, please include [profiling logs](#). See the section below about [information you should provide](#).
4. Click **Send**.

This will produce a zip file containing the information categories selected from the list and will email this to Atlassian Support. You will receive an email advising you of details of the Support Request that was automatically created, and you will receive emailed updates about progress on your issue. You can also see the status of your request directly by visiting the [Atlassian Support System](#).

Raising a Support request yourself at Atlassian Support

1. Log in to Stash (as a [System Administrator](#)) and go to the [admin area](#).
2. Click **Atlassian Support Tools** (under 'Support') then **Support Zip**.
3. Select information categories to include in the zip file.
4. Click **Create**.

The zip file is created in the [home directory](#) of the Stash server, for example `<STASH_HOME>\export\Stash_support_2013-11-17-20-49-18.zip`.

When you now go to [Atlassian Support](#) and create a Support Request, you can attach the Support Zip file to the request.

Please provide as much information as possible in the request, including steps to replicate the problem, and any error messages that are appearing on the console or in the [logs](#). For performance issues, please include [profilin g logs](#). See the section below about [information you should provide](#).

Information you should provide

In addition to the logs and configuration information that you can include in the Support Request zip file, the following information can help to give you a faster response:

Environment details

- Stash version
- Java version (for example OpenJDK 1.7.0 JDK)
- Git and Perl versions
- Operating system (for example, Windows 7, Mac OS X 10.6.8)
- Database type (for example, MySQL) and version
- Browsers and versions
- Network topology - is Stash running behind a reverse proxy? Is that secured using HTTPS (SSL)?

Configuration

- Java settings, including JVM_MINIMUM_MEMORY, JVM_MAXIMUM_MEMORY

Logs

You may need to adjust the logging level, or enable profiling in Stash, in order to get more detailed logs. See [Stash debug logging](#).

- Debug logs – Stash debug logs can be found in `<STASH_HOME>/log`.
- Profiling logs – Stash profiling logs can help with analyzing performance issues and can be found in `<STASH_HOME>/log`.

Performance factors

- Number of concurrent Git clones
- Number of users
- The size of the `.git` directory
- CPU spec, number of cores, whether hyperthreading is enabled
- RAM and cache sizes

Integrations

- Other Atlassian applications (and their versions)
- Which build servers are integrated with Stash, if any?
- Are Application Links configured?

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 Issue

If you find a security issue in the product, open an issue on <https://jira.atlassian.com> in the relevant project.

- Set the **security level** of the bug to 'Reporters and Developers'.
- Set the priority of the bug to 'Blocker'.
- Provide as much information on reproducing the bug as possible.

All communication about the security issue should be performed through JIRA, so that Atlassian can keep track of the issue and get a patch out as soon as possible.

If you cannot find the right project to file your issue in, email the details to security@atlassian.com.



When reporting a security vulnerability, please keep in mind the following:

We need a technical description that allows us to assess exploitability and impact of the issue.

- Provide steps to reproduce the issue, including any URLs or code involved.
- If you are reporting a cross-site scripting (XSS), your exploit should at least pop up an alert in the browser. It is much better if the XSS exploit shows user's authentication cookie.
- For a cross-site request forgery (CSRF), use a proper CSRF case when a third party causes the logged in victim to perform an action.
- For a SQL injection, we want to see the exploit extracting database data, not just producing an error message.
- HTTP request / response captures or simply packet captures are also very useful to us.

Please refrain from sending us links to non-Atlassian web sites, or reports in PDF / DOC / EXE files. Image files are ok. Make sure the bug is exploitable by someone other than the user himself (e.g. "self-XSS").

Without this information it is not possible to assess your report and it is unlikely to be addressed.

We are not looking for the reports listing generic "best practice" issues such as:

- Specific cookies being not marked as Secure or HTTPOnly
- Presence or absence of HTTP headers (X-Frame-Options, HSTS, CSP, nosniff and so on)
- Clickjacking
- Mixed HTTP and HTTPS content
- Auto-complete enabled or disabled
- SSL-related issues

Further reading

See [Atlassian Support Offerings](#) for more support-related information.

New features policy

Summary

- We encourage and display customer comments and votes openly in our issue tracking system, <http://jira.atlassian.com>.
- We do not publish roadmaps.
- Product Managers review our most popular voted issues on a regular basis.
- We schedule features based on a variety of factors.
- Our [Atlassian Bug Fixing Policy](#) is distinct from 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:

- **Customer contact:** We get the chance to meet customers and hear their successes and challenges at Atlassian Summit, Atlassian Unite, developer conferences, and road shows.
- **Customer interviews:** All product managers at Atlassian do customer interviews. Our interviews are not simply to capture a list of features, but to understand our customers' goals and plans.
- **Community forums:** There are large volumes of posts on [answers](#), of votes and comments on [jira.atlassian.com](#), and of conversations on community forums like groups on LinkedIn.
- **Customer Support:** Our support team provides clear insights into the issues that are challenging for customers, and which are generating the most calls to support
- **Atlassian Experts:** Our [Experts](#) provide insights into real-world customer deployments, especially for customers at scale.
- **Evaluator Feedback:** When someone new tries our products, we want to know what they liked and disliked and often reach out to them for more detail.
- **In product feedback:** The [JIRA Issue Collectors](#) that we embed our products for evaluators and our Early Access Program give us a constant pulse on how users are experiencing our product.
- **Usage data:** Are customers using the features we have developed?
- **Product strategy:** Our long-term strategic vision for the product.

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 system, <http://jira.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).
- AND
- 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 **critical severity** 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.
- We will send a copy of all posted 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 'Communications Centre' 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.

If you want to track non-critical severity security vulnerabilities, you need to monitor the issue trackers for the relevant products on <http://jira.atlassian.com>. For example, <https://jira.atlassian.com/browse/JRA> for JIRA and <https://jira.atlassian.com/browse/CONF> for Confluence. Security issues in trackers will be marked with a "security" label. All security issues will be listed in the release notes of the release where they have been fixed, similar to other bugs.

One of the ways to monitor updates to security issues is subscribing to the results of a [sample search](#) via email or RSS.

Further reading

See [Atlassian Support Offerings](#) for more support-related information.

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. This policy excludes OnDemand and Bitbucket, since these services are always 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](https://first.org) web site.

CVSS scores are mapped into the following severity ratings:

- Critical
- High
- Medium
- Low

An approximate mapping guideline is as follows:

CVSS score range	Severity in advisory
0 – 2.9	Low
3 – 5.9	Medium
6.0 – 7.9	High
8.0 – 10.0	Critical

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 most of 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 some of 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 some of 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](#) for more support-related information.

Building Stash from source

This page has moved!

To our Development Hub at <https://developer.atlassian.com/display/STASHDEV/Building+from+Source+Code>.

But you really wanted to [build a plugin anyway](#), right?

Contributing to the Stash documentation

Would you like to share your Stash hints, tips and techniques with us and with other Stash users? We welcome your contributions.

Blogging your technical tips and guides

Have you written a blog post describing a specific configuration of Stash or a neat trick that you have discovered? Let us know, and we will link to your blog from our documentation.

Contributing documentation in other languages

Have you written a guide to Stash in a language other than English, or translated one of our guides? Let us know, and we will link to your guide from our documentation.

On this page:

- [Blogging your technical tips and guides](#)
- [Contributing documentation in other languages](#)
- [Updating the documentation itself](#)
 - [Getting permission to update the documentation](#)
 - [Our style guide](#)
 - [How we manage community updates](#)

Updating the documentation itself

Have you found a mistake in the documentation, or do you have a small addition that would be so easy to add yourself rather than asking us to do it? You can update the documentation page directly

Getting permission to update the documentation

Please submit the [Atlassian Contributor License Agreement](#).

Our style guide

Please read our short [guidelines for authors](#).

How we manage community updates

Here is a quick guide to how we manage community contributions to our documentation and the copyright that applies to the documentation:

- **Monitoring by technical writers.** The Atlassian technical writers monitor the updates to the documentation spaces, using RSS feeds and watching the spaces. If someone makes an update that needs some attention from us, we will make the necessary changes.
- **Wiki permissions.** We use wiki permissions to determine who can edit the documentation spaces. We ask people to sign the [Atlassian Contributor License Agreement](#) (ACLA) and submit it to us. That allows us to verify that the applicant is a real person. Then we give them permission to update the documentation.
- **Copyright.** The Atlassian documentation is published under a Creative Commons CC BY license. Specifically, we use a [Creative Commons Attribution 2.5 Australia License](#). This means that anyone can copy, distribute and adapt our documentation provided they acknowledge the source of the documentation. The CC BY license is shown in the footer of every page, so that anyone who contributes to our documentation knows that their contribution falls under the same copyright.