

Integrating with Apache

This page describes how to integrate an [Apache web server](#) with JIRA (via [mod_proxy](#)), such that Apache forwards requests on to JIRA, and responses back to the user. This is useful if you already have Apache serving web pages on port 80 (eg. <http://mycompany.com>), and wish to integrate JIRA as just another URL (eg. <http://mycompany.com/jira>).

Note:

This documentation describes a straightforward implementation of `mod_proxy`. If you require a more complex solution, refer to [Apache HTTP Server Version Documentation](#) and, if necessary, consult with someone in your organization who is knowledgeable in the configuration of Apache.

1. Step 1: Configure JIRA's application server

Here we assume you are using the JIRA Standalone distribution, which comes with Tomcat 5.5. First, we need to edit Tomcat's `conf/server.xml` file, and set the context path:

```
<Server port="8005" shutdown="SHUTDOWN">
    ...
    <Context path="/jira" docBase="${catalina.home}/atlassian-jira" reloadable="false"
        <Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"
            ...
        />
    />
```

Here we have set the context path to `/jira`, assuming JIRA will be running on <http://mycompany.com/jira/>.

Restart Tomcat, and ensure you can still access JIRA normally (eg. at <http://localhost:8080/jira/>).

Note: if you want Tomcat responsible for all URLs, specify a blank context path with `path=""` -- *not* `path="/"`.

Turn JIRA's [GZip compression](#) **OFF** (since there will be no benefit from GZip compression once proxying is implemented).

2. Step 2: Configure Apache

Assuming an Apache 2 installation, the following needs to be done:

2.1. Enable `mod_proxy` and `mod_proxy_http`

The exact steps will be specific to your operating system. Refer to the Apache documentation for your operating system. On Debian/Ubuntu it is done as follows:

```
teacup:/etc/apache2# a2enmod proxy_http
Enabling proxy as a dependency
Module proxy installed; run /etc/init.d/apache2 force-reload to enable.
Module proxy_http installed; run /etc/init.d/apache2 force-reload to enable.
teacup:/etc/apache2#
```

2.2. Configure `mod_proxy`

Here we create a config snippet for JIRA, in `sites-available/jira-mod_proxy`:

```
teacup:/etc/apache2# cd sites-available
teacup:/etc/apache2/sites-available# cat > jira-mod_proxy

<Proxy *>
Order deny,allow
Allow from all
</Proxy>
```

```
ProxyRequests          Off
ProxyPreserveHost     On
ProxyPass              /jira          http://localhost:8080/jira
ProxyPassReverse       /jira          http://localhost:8080/jira
```

```
teacup:/etc/apache2/sites-available#
teacup:/etc/apache2/sites-available# a2ensite jira-mod_proxy
Site jira-mod_proxy installed; run /etc/init.d/apache2 reload to enable.
```

```
teacup:/etc/apache2/sites-available# /etc/init.d/apache2 reload
Reloading apache 2.0 configuration...
teacup:/etc/apache2/sites-available#
```

JIRA should now be integrated with Apache. You should be able to view JIRA at `http://localhost/jira` (i.e. on port 80).

Notes:

- The path `'/jira'` **must** be the same as the context path in Tomcat's `conf/server.xml`
- The **ProxyPreserveHost** directive allows Tomcat to know its public hostname and port. Without this, JIRA would redirect the public URL (eg. `http://mycompany.com/jira/`) to `http://localhost:8080/jira/secure/Dashboard.jspa`.

Note:

If the links for Printable Version, RSS feeds, Word export and Excel export have incorrect URLs, starting with `localhost:8080/jira` instead of `http://mycompany.com/jira`, ensure that **ProxyPreserveHost** is set to **On**.

ProxyPreserveHost is only available on Apache 2. For Apache 1.1-1.3.x, you should instead specify **proxyName** and **proxyPort** attributes in Tomcat as follows:

```
<Server port="8005" shutdown="SHUTDOWN">

  <Service name="Catalina">

    <Connector port="8080"
      maxHttpHeaderSize="8192" maxThreads="150" minSpareThreads="25" maxSpareThreads="
      enableLookups="false" redirectPort="8443" acceptCount="100" connectionTimeout="
      proxyName="mycompany.com" proxyPort="80" />

    <Engine name="Catalina" defaultHost="localhost">
      <Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true">

        <Context path="/jira" docBase="${catalina.home}/atlassian-jira" reloadable="
          <Resource name="jdbc/JiraDS" auth="Container" type="javax.sql.DataSource"
            ....
```

If you are using Apache 1.x, make sure you **don't** use caching (`CacheRoot` directive).

Warning:

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 2 distributions.

3. Configuring SSL

If you want to use https (eg. `https://mycompany.com/jira/`), then:

- Configure Tomcat to serve HTTP traffic on port 8443. There are instructions [here](#). The end result should be that `https://localhost:8443/jira/` works.
- In the Apache config (`/etc/apache2/sites-available/jira-mod_proxy`), ensure you have **SSLProxyEngine on** specified, and redirect `/jira` to `https://localhost:8443/jira:`

```
<Proxy *>
Order deny,allow
Allow from all
</Proxy>

SSLProxyEngine on
ProxyRequests Off
ProxyPreserveHost On
ProxyPass /jira https://localhost:8443/jira
ProxyPassReverse /jira https://localhost:8443/jira
```

- Please ensure that the ProxyPass and ProxyPassReverse directives do not include a trailing '/'. There have been reports that this may cause problems in JIRA 3.7 and above when serving static resources (javascript and css).

4. 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.

If you are on Macintosh OS X, please disable **webperfcache**, which proxies port 80 by default. A user reported this as the likely cause of JIRA session problems, in the form of users' identities becoming mixed up:

```
The OSX Servers enable webperfcache by default for Virtual Hosts, which for static
would be great, but for dynamic sites (which ALL of ours are) it is Evil and cause
issues. Of note recently was the jira session issue. Also see :-
http://developer.apple.com/documentation/Darwin/Reference/ManPages/man8/webperfca
http://www.wodeveloper.com/omniLists/webobjects-admin/2005/January/msg00009.html
Unfortunately even if you disable webperfcache for a site, if there is a single si
then all sites will still proxy through webperfcache with resulting session proble
```

In general, if you are having problems:

1. Ensure that JIRA works as expected when running directly from Tomcat on `http://localhost:8080/jira`
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.

5. For more details..

For more advanced mod_webapp configurations (eg. SSL), see [this mod_proxy guide](#).