

# Configuring AJP13 Using mod\_jk

## Configuring AJP13 Using mod\_jk or mod\_proxy\_ajp

The apache web server is frequently used as a server in front of a servlet container. While there are no real technical reasons to front Jetty with apache, sometimes this is needed for software load balancing, or to fit with a corporate infrastructure, or simply to stick with a known deployment structure.

There are 3 main alternative for connection Apache to Jetty:

1. Using apache [mod\\_proxy](#) and an normal Jetty HTTP connector.
2. Using apache [mod\\_proxy\\_ajp](#) and the Jetty AJP connector.
3. Using apache [mod\\_jk](#) and the Jetty AJP connector.

Using the HTTP Connectors is greatly preferred, as Jetty performs significantly better with HTTP and the AJP protocol is poorly documented and there are many version irregularities. If AJP is to be used, the then `mod_proxy_ajp` module is preferred over `mod_jk`. Previously, the load balancing capabilities of `mod_jk` meant that it had to be used (tolerated), but with apache 2.2, [mod\\_proxy\\_balancer](#) is available and load balance over HTTP and AJP connectors.

## Using HTTP

**THIS IS THE RECOMMENDED MECHANISM TO CONNECT APACHE AND JETTY**

To configure apache to use `mod_proxy`, `mod_proxy_http` and/or `mod_proxy_balancer` with HTTP see [Configuring mod\\_proxy](#).

## Using AJP

**AJP is NOT recommended. Use HTTP and mod\_proxy instead (see above)**

*AJP is not recommended for a number of reasons:*

- *historically mod\_jk has had intermittent maintenance and bad versioning practises. This make it difficult to select a known good version that is fully compatible with the AJP connector you are running.*
- *The mod\_proxy plugin is more actively maintained and the mod\_proxy\_balancer supports a richer set of options for load balancing.*
- *Jetty is optimized to deal with the text based HTTP protocol and the servlet API also exposes the text nature of HTTP to the application. There are no measurable benefits of using apache to convert text HTTP to the binary AJP protocol, only for jetty to have to convert back. Some tests have shown 15% more throughput with mod\_proxy than with mod\_ajp*
- *With cometd style applications, neither mod\_jk nor mod\_proxy scale well. However, mod\_proxy does make greater use of connections, so it is a better choice for moderate comet load. For full comet scaling, either jetty should be directly exposed to the internet or an async load balancer like nginx should be used.*
- *Note however that AJP is still full supported by the jetty team and we will strive to fix any issues found.*

## The Jetty AJP connector.

To use AJP with either `mod_jk` or `mod_proxy_ajp`, Jetty needs to be configured with an AJP13 connector. This can be configured by adding `etc/jetty-ajp.xml` to the command line. Alternately an existing `jetty.xml` file may be modified with:

```
<Call name="addConnector">
  <Arg>
    <New
      class="org.mortbay.jetty.ajp.Ajp13SocketConnector">
        <Set name="port">8009</Set>
      </New>
    </Arg>
  </Call>
```

The full options for the `Ajp13SocketConnector` are available in the [javadoc](#).

## mod\_proxy\_ajp

With apache 2.2 [mod\\_proxy\\_ajp](#) is an extension of the [mod\\_proxy](#) module and may also be used in conjunction with the [mod\\_proxy\\_balancer](#) module. Prior to 2.2, `mod_proxy` did not support AJP.

### Compatibility

Apache	Win32	Linux(ubuntu)
Apache 1.3	no mod_proxy_ajp bundled	no mod_proxy_ajp bundled
Apache 2.0 (2.0.59)	no mod_proxy_ajp bundled	no mod_proxy_ajp bundled
Apache 2.2	✓	✓

### Configuration

The configuration of `mod_proxy_ajp` is identical to the [Configuration of mod\\_proxy](#), except that `ajp://` may be used as a protocol instead of `http://` when specifying destinations (workers) in `ProxyPass` and `BalancerMember` elements.

Apache 2.2 normally bundles `mod_proxy`, `mod_proxy_ajp` and `mod_proxy_balancer`, so they often do not need to be installed separately. If they are separately bundled by your operation system (eg as RPMs or debians) ensure that they are installed.

The apache configuration structure can vary greatly with operating system distros and there may be some template

configurations for mod\_proxy. If not, add the entry below in your httpd.conf apache configuration file located in <apache-root>/conf/ directory

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module
modules/mod_proxy_ajp.so
LoadModule proxy_balancer_module
modules/mod_proxy_balancer.so

# always keep the host header
ProxyPreserveHost On

# map to cluster
ProxyPass /test balancer://my_cluster/test
stickysession=JSESSIONID nofailover=On
ProxyPass /demo balancer://my_cluster/demo
stickysession=JSESSIONID nofailover=On

# define the balancer, with http and/ or ajp
connections
<Proxy balancer://my_cluster>
    BalancerMember ajp://yourjettyhost1:8009
    BalancerMember ajp://yourjettyhost2:8009
</Proxy>
```

Where:

- **LoadModule** - tells your apache server to load a module library and where it is located.
- **ProxyPreserveHost On** - keeps the original Host Header. **THIS IS HIGHLY RECOMMENDED FOR ALL PROXY CONFIGURATIONS**
- **ProxyPass** - Maps a path to a proxied destination. The destination may be a `http://` or `ajp://` URL to directly map to a single server, or it may be a `balancer://` URL to map to a cluster.
- **Proxy balancer://** - defines the nodes (workers) in the cluster. Each member may be a `{http://}` or `ajp://` U

RL or another balancer : // URL for cascaded load balancing configuration.

## mod\_jk

It is NOT recommended to use mod\_jk

### Compatibility

Apache	mod_jk	Win32	Linux(ubuntu)
Apache 1.3		No HTTPD Binary Available	
	mod_jk-1.2.14		Not yet tested
	mod_jk-1.2.15		Not yet tested
	mod_jk-1.2.18		Not yet tested
	mod_jk-1.2.19		Not yet tested
Apache 2.0 (2.0.59)			
	mod_jk-1.2.14	✓	
	mod_jk-1.2.15	✓	
	mod_jk-1.2.18	✓	
	mod_jk-1.2.19	✓	
Apache 2.2			
	mod_jk-1.2.14	No Binary Available	
	mod_jk-1.2.15	No Binary Available	
	mod_jk-1.2.18	✓	
	mod_jk-1.2.19	✓	

### Configuring Apache HTTPD server with mod\_jk

1. put mod\_jk.so into your <apache-root>/modules/ directory
2. you can download mod\_jk.so here <http://www.opensourcecommunity.ph/apache/tomcat/tomcat-connectors/jk/binaries/>
3. add the entry below in your httpd.conf apache configuration file located in <apache-root>/conf/ directory.

```

<IfModule !mod_jk.c>

    LoadModule jk_module    modules/mod_jk.so

</IfModule>

<IfModule mod_jk.c>

    JkWorkersFile "conf/worker.properties"

    JkLogFile "logs/mod_jk.log"

    JkLogLevel info

    JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "

    JkOptions +ForwardKeySize +ForwardURICompat

</IfModule>

```

Where:

4. **LoadModule jk\_module modules/mod\_jk.so** tells your apache server to load the mod\_jk library and where it is located.
5. **JkWorkersFile conf/worker.properties** tells mod\_jk where your worker.properties is located.
6. **JkLogFile logs/mod\_jk.log** tells mod\_jk where to write mod\_jk related Logs.
7. After adding the mod\_jk configuration you may add a **VirtualHost** Entry in the same file (httpd.conf) as long as its located below your mod\_jk configuration entry:

```

<VirtualHost host:*>

    ServerName yourserver

    ServerAdmin user@yourserver

    ## you may add further entries concerning log-files, log-level,
    URL-rewriting, ...

    ## pass requests through to jetty worker

    JkMount /* jetty

</VirtualHost>

```

8. Add a worker file **worker.properties** in your <apache-root>/conf/
9. add the entries below, and make sure to specify your ip-address or hostname in **worker.jetty.host** property entry to where your jetty application is running

```
worker.list=jetty
```

```
worker.jetty.port=8009
```

```
worker.jetty.host=<server name or ip where your jetty will be running>
```

```
worker.jetty.type=ajp13
```

```
worker.jetty.lbfactor=1
```