mod_cluster Documentation

Overview

mod_cluster is an httpd-based load balancer. Like mod_jk and mod_proxy, mod_cluster uses a communication channel to forward requests from httpd to one of a set of application server nodes. Unlike mod_jk and mod_proxy, mod_cluster leverages an additional connection between the application server nodes and httpd. The application server nodes use this connection to transmit server-side load balance factors and lifecycle events back to httpd via a custom set of HTTP methods, affectionately called the Mod-Cluster Management Protocol (MCMP). This additional feedback channel allows mod_cluster to offer a level of intelligence and granularity not found in other load balancing solutions.

Within httpd, mod_cluster is implemented as a set of modules for httpd with mod_proxy enabled. Much of the logic comes from mod_proxy, e.g. mod_proxy_ajp provides all the AJP logic needed by mod_cluster.

1.1. Platforms

JBoss already prepares *binary packages* [http://www.jboss.org/mod_cluster/downloads.html] with httpd and mod_cluster so you can quickly try mod_cluster on the following platforms:

- Linux x86, x64, ia64
- Solaris x86, SPARC
- Windows x86, x64, ia64
- HP-UX PA-RISC, ia64

1.2. Advantages

mod_cluster boasts the following advantages over other httpd-based load balancers:

Dynamic configuration of httpd workers

> Traditional httpd-based load balancers require explicit configuration of the workers available to a proxy. In mod_cluster, the bulk of the proxy's configuration resides on the application servers. The set of proxies to which an application server will communicate is determined either by a static list or using dynamic discovery via the *advertise* mechanism. The application server relays lifecycle events (e.g. server startup/shutdown) to the proxies allowing them to effectively auto-configure themselves. Notably, the graceful shutdown of a server will not result in a failover response by a proxy, as is the case with traditional httpd-based load balancers.

Server-side load balance factor calculation

> In contrast with traditional httpd-based load balancers, mod_cluster uses load balance factors calculated and provided by the application servers, rather than computing these in the proxy.

Consequently, mod_cluster offers a more robust and accurate set of load metrics than is available from the proxy. See the chapter entitled *Server-Side Load Metrics* for details.

Fine grained web-app lifecycle control
: Traditional httpd-based load balancers do not handle web application undeployments particularly well. From the proxy's perspective requests to an undeployed web application are indistinguishable from a request for an non-existent resource, and will result in 404 errors. In mod_cluster, each server forwards any web application context lifecycle events (e.g. web-app deploy/undeploy) to the proxy informing it to start/stop routing requests for a given context to that server.

AJP is optional
: Unlike mod_jk, mod_cluster does not require AJP. httpd connections to application server nodes can use HTTP, HTTPS, or AJP.

The original concepts are described in a *wiki* [http://www.jboss.org/community/docs/DOC-11431].

1.3. Requirements

- httpd-2.2.8+
- JBoss AS 5.0.0+ or JBossWeb 2.1.1+

Note

httpd-2.2.8+ is already in the bundles, so if you use the bundle you don't need to download Apache httpd.

1.4. Limitations

mod_cluster uses shared memory to keep the nodes description, the shared memory is created at the start of httpd and the structure of each item is fixed. The following can't be changed by configuration directives.

- Max Alias length 40 characters (Host: hostname header, Alias in <Host/>).
- Max context length 40 (for example myapp.war deploys in /myapp /myapp is the context).
- Max balancer name length 40 (balancer property in mbean).
- Max JVMRoute string length 80 (JVMRoute in <Engine/>).
- Max load balancing group name length 20 (domain property in mbean).
- Max hostname length for a node 64 (address in the <Connector/>).
- Max port length for a node 7 (8009 is 4 characters, port in the <Connector/>).

- Max scheme length for a node 6 (possible values are http, https, ajp, liked with the protocol of <Connector/>).
- Max cookie name 30 (the header cookie name for sessionid default value: JSESSIONID from org.apache.catalina.Globals.SESSION_COOKIE_NAME).
- Max path name 30 (the parameter name for the sessionid default value: jsessionid from org.apache.catalina.Globals.SESSION_PARAMETER_NAME).
- Max length for a sessionid 120 (something like BE81FAA969BF64C8EC2B6600457EAAAA.node01).

1.5. Downloads

Download the latest mod_cluster release *here* [http://www.jboss.org/mod_cluster/downloads/latest].

The release is comprised of the following artifacts:

- httpd binaries for common platforms
- JBoss AS/JBossWeb/Tomcat binary distribution

Alternatively, you can build from source using the subversion repository:

http://anonsvn.jboss.org/repos/mod_cluster/tags/release/ [http://anonsvn.jboss.org/repos/mod_cluster/tags/]

- Building *httpd modules*
- Building *server-side components*

1.6. Configuration

If you want to skip the details and just set up a minimal working installation of mod_cluster, see the *Quick Start Guide*.

- Configuring *httpd*
- Configuring *JBoss AS*
- Configuring *JBoss Web or Tomcat*

1.7. Migration

Migrating from mod_jk or mod_proxy is fairly straight forward. In general, much of the configuration previously found in httpd.conf is now defined in the application server nodes.

- Migrating from *mod_jk*
- Migrating from *mod_proxy*

1.8. SSL support

Both the request connections between httpd and the application server nodes, and the feedback channel between the nodes and httpd can be secured. The former is achieved via the *mod_proxy_https module* and a corresponding ssl-enabled HTTP connector in JBoss Web. The latter requires the *mod_ssl module* and *explicit configuration in JBoss AS/Web*.

1.9. Load Balancing Demo Application

The mod_cluster binary distribution for JBoss AS/JBossWeb/Tomcat includes a *demo application* that helps demonstrate how different server-side scenarios affect the routing of client requests by the load balancer. The demo application is located in the mod_cluster distribution's demo directory.

Strong cryptography warning

mod_cluster contains mod_ssl, therefore the following warning (copied from *OpenSSL* [http://www.openssl.org/]) applies:

PLEASE REMEMBER THAT EXPORT/IMPORT AND/OR USE OF STRONG CRYPTOGRAPHY SOFTWARE, PROVIDING CRYPTOGRAPHY HOOKS OR EVEN JUST COMMUNICATING TECHNICAL DETAILS ABOUT CRYPTOGRAPHY SOFTWARE IS ILLEGAL IN SOME PARTS OF THE WORLD. SO, WHEN YOU IMPORT THIS PACKAGE TO YOUR COUNTRY, RE-DISTRIBUTE IT FROM THERE OR EVEN JUST EMAIL TECHNICAL SUGGESTIONS OR EVEN SOURCE PATCHES TO THE AUTHOR OR OTHER PEOPLE YOU ARE STRONGLY ADVISED TO PAY CLOSE ATTENTION TO ANY EXPORT/IMPORT AND/OR USE LAWS WHICH APPLY TO YOU. THE AUTHORS OF OPENSSL ARE NOT LIABLE FOR ANY VIOLATIONS YOU MAKE HERE. SO BE CAREFUL, IT IS YOUR RESPONSIBILITY.

Quick Start Guide

The following are the steps to set up a minimal working installation of mod_cluster on a single httpd server and a single back end server, either JBoss AS, JBossWeb or Tomcat. The steps can be repeated to add as many httpd servers or back end servers to your cluster as is desired.

The steps shown here are not intended to demonstrate how to set up a production install of mod_cluster; for example *using SSL to secure access* to the httpd-side mod_manager component is not covered. See the *httpd-side* and *java-side* configuration documentation for the full set of configuration options.

2.1. Download mod_cluster components

Download the latest *httpd and java release bundles* [http://www.jboss.org/mod_cluster/downloads/latest.html]. If there is no pre-built httpd bundle appropriate for your OS or system architecture, you can *build the binary from source*.

2.2. Install the httpd binary

2.2.1. Install the whole httpd

The httpd-side bundles are gzipped tars and include a full httpd install. As they contain already an Apache httpd install you don't need to download Apache httpd. Just extract them in root, e.g.

```
cd /
tar xvf mod-cluster-1.0.0-linux2-x86-ssl.tar.gz
```

That will give you a full httpd install in your `/opt/jboss` directory.

2.2.2. Install only the modules

If you already have a working httpd install that you would prefer to use, you'll need to download the bundle named mod_cluster httpd dynamic libraries corresponding to you plaform, extract the modules and copy them directory to your httpd install's module directory.

```
cd /tmp
tar xvf mod_cluster-1.0.0.CR2-linux2-x86-so.tar.gz
```

And then you have to copy the files below to you module directory:

- mod_proxy.so

- mod_proxy_ajp.so
- mod_slotmem.so
- mod_manager.so
- mod_proxy_cluster.so
- mod_advertise.so

2.2.3. Install in home directory

Since 1.1.0.CR2 a script `opt/jboss/httpd/sbin/installhome.sh` allows reconfiguration of the bundle installation so that it can run in user's home directory. To do that just extract the bundle in your home directory and run the script. Once that done, httpd will run on port 8000 and will accept MCMP messages on localhost:6666 and offer `/mod_cluster_manager` on the same host and port.

2.2.4. Install in Windows

Unzip the bundle corresponding to your architecture.

Change to the bin directory of the subfolder httpd-2.2 where you unzip the bundle and run the installconf.bat shell script.

```
cd httpd-2.2\bin
      installconf.bat
```

You may run httpd directly by using:

```
httpd.exe
```

or install Apache httpd as a service:

```
httpd.exe -k install
```

and start the service via net start or using httpd.exe:

```
net start Apache2.2
```

```
httpd.exe -k start
```

Note that in the windows bundles have a flat directory structure, so you have *httpd-2.2/modules/* instead of `opt/jboss/httpd/lib/httpd/modules`.

2.3. Configure httpd

Since 1.1.0.CR2 httpd.conf is preconfigured with the Quick Start values. You should adapt the default values to your configuration with older mod_cluster we will have to add the following to httpd.conf. If you extracted the download bundle to root as shown above and are using that extract as your httpd install, httpd.conf is located in `/opt/jboss/httpd/httpd/conf`.

```
LoadModule proxy_module /opt/jboss/httpd/lib/httpd/modules/mod_proxy.so
LoadModule proxy_ajp_module /opt/jboss/httpd/lib/httpd/modules/mod_proxy_ajp.so
LoadModule slotmem_module /opt/jboss/httpd/lib/httpd/modules/mod_slotmem.so
LoadModule manager_module /opt/jboss/httpd/lib/httpd/modules/mod_manager.so
LoadModule proxy_cluster_module /opt/jboss/httpd/lib/httpd/modules/mod_proxy_cluster.so
LoadModule advertise_module /opt/jboss/httpd/lib/httpd/modules/mod_advertise.so

Listen 10.33.144.3:6666
<VirtualHost 10.33.144.3:6666>

  <Directory />
    Order deny,allow
    Deny from all
    Allow from 10.33.144.
  </Directory>

  KeepAliveTimeout 60
  MaxKeepAliveRequests 0

  ManagerBalancerName mycluster
  AdvertiseFrequency 5

</VirtualHost>
```

If you are using your own install of httpd, httpd.conf is found in your install's conf directory. The content to add to httpd.conf is slightly different from the above (different path to the various .so files):

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule slotmem_module modules/mod_slotmem.so
LoadModule manager_module modules/mod_manager.so
LoadModule proxy_cluster_module modules/mod_proxy_cluster.so
LoadModule advertise_module modules/mod_advertise.so

Listen 10.33.144.3:6666
<VirtualHost 10.33.144.3:6666>

  <Directory />
    Order deny,allow
    Deny from all
    Allow from 10.33.144.
  </Directory>

  KeepAliveTimeout 60
  MaxKeepAliveRequests 0

  ManagerBalancerName mycluster
  AdvertiseFrequency 5

</VirtualHost>
```

2.4. Install the server-side binaries

First, extract the java-side binary to a temporary directory. The following assumes it was extracted to /tmp/mod-cluster

Your next step depends on whether your target server is JBoss AS or JBoss Web/Tomcat.

2.4.1. Installing in JBoss AS 6.x

You don't need to do anything to install the java-side binaries in AS 6.x; it's part of the AS distribution's default, standard and all configurations.

2.4.2. Installing in JBoss AS 5.x

Assuming $JBOSS_HOME indicates the root of your JBoss AS install and that you want to use mod_cluster in the AS's all config:

```
cp -r /tmp/mod-cluster/mod-cluster.sar $JBOSS_HOME/server/all/deploy
```

2.4.3. Installing in JBoss Web or Tomcat

Assuming $CATALINA_HOME indicates the root of your JBoss Web or Tomcat install:

```
cp -r /tmp/mod-cluster/JBossWeb-Tomcat/lib/*
```

```
    $CATALINA_HOME/lib/
```

2.4.4. Installing in JBoss AS 4.2.x or 4.3.x

Assuming $JBOSS_HOME indicates the root of your JBoss AS install and that you want to use mod_cluster in the AS's all config:

```
cp -r /tmp/mod-cluster/JBossWeb-Tomcat/lib/mod-cluster.jar $JBOSS_HOME/server/all/deploy/
jboss-web.deployer
```

2.5. Configuring the server-side

2.5.1. Configuring mod_cluster with JBoss AS 5.x+

No post-installation configuration necessary!

2.5.2. Configuring mod_cluster with standalone JBoss Web or Tomcat

Edit the `$CATALINA_HOME/conf/server.xml` file, adding the following next to the other <Listener/> elements:

```
<Listener className="org.jboss.modcluster.catalina.ModClusterListener" advertise="true"/>
```

2.5.3. Integrate mod_cluster with JBoss AS 4.2.x and 4.3.x

Edit the `$JBOSS_HOME/server/all/deploy/jboss-web.deployer/server.xml` file, adding the following next to the other <Listener/> elements:

```
<Listener className="org.jboss.modcluster.catalina.ModClusterListener" advertise="true"/>
```

2.6. Start httpd

To start httpd do the following:

```
/opt/jboss/httpd/sbin/apachectl start
```

2.7. Start the back-end server

2.7.1. Starting JBoss AS

```
cd $JBOSS_HOME/bin
./run.sh -c all
```

2.7.2. Starting JBossWeb or Tomcat

```
cd $CATALINA_HOME
./startup.sh
```

2.8. Set up more back-end servers

Repeat the back-end server install and configuration steps for each server in your cluster.

2.9. Experiment with the Load Balancing Demo Application

The *load balancing demo application* is a good way to learn about mod_cluster's capabilities.

httpd configuration

3.1. Apache httpd configuration

You need to load the modules that are needed for mod_cluster for example:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule slotmem_module modules/mod_slotmem.so
LoadModule manager_module modules/mod_manager.so
LoadModule proxy_cluster_module modules/mod_proxy_cluster.so
LoadModule advertise_module modules/mod_advertise.so
```

mod_proxy and mod_proxy_ajp are standard httpd modules. mod_slotmem is a shared slotmem memory provider. mod_manager is the module that reads information from JBoss AS/JBossWeb/Tomcat and updates the shared memory information. mod_proxy_cluster is the module that contains the balancer for mod_proxy. mod_advertise is an additional module that allows httpd to advertise via multicast packets the IP and port where the mod_cluster is listening. This multi-module architecture allows the modules to easily be changed depending on what the customer wants to do. For example when using http instead of AJP, only

```
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
```

needs to be changed to:

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

3.2. mod_proxy configuration

mod_proxy directives like ProxyIOBufferSize could be used to configure mod_cluster. There is no need to use ProxyPass directives because mod_cluster automatically configures which URLs have to be forwarded to JBossWEB.

3.3. mod_slotmem configuration

The actual version doesn't require any configuration directives.

3.4. mod_proxy_cluster

3.4.1. CreateBalancers

CreateBalancers: Define how the balancer are created in the httpd VirtualHosts, this is to allow directives like:

```
ProxyPass / balancer://mycluster1/
```

0: Create in all VirtualHosts defined in httpd.

1: Don't create balancers (requires at least one ProxyPass/ProxyPassMatch to define the balancer names).

2: Create only the main server.

Default: 2

Note: When using 1 don't forget to configure the balancer in the ProxyPass directive, because the default is empty stickysession and nofailover=Off and the values received via the MCMP CONFIG message are ignored.

3.4.2. UseAlias

UseAlias: Check that the Alias corresponds to the ServerName (See *Host Name Aliases* [http://labs.jboss.com/file-access/default/members/jbossweb/freezone/docs/latest/config/host.html])

0: Don't (ignore Aliases)

1: Check it

Default: 0 Ignore the Alias information from the nodes.

3.4.3. LBstatusRecalTime

LBstatusRecalTime: Time interval in seconds for loadbalancing logic to recalculate the status of a node

Default: 5 seconds

The actual formula to recalculate the status of a node is:

```
status = lbstatus + (elected - oldelected) * 1000)/lbfactor;
```

lbfactor is received for the node via STATUS messages.lbstatus is recalculated every LBstatusRecalTime seconds using the formula:

```
lbstatus = (elected - oldelected) * 1000)/lbfactor;
```

elected is the number of time the worker was elected.oldelected is elected last time the lbstatus was recalculated.The node with the lowest status is selected. Nodes with lbfactor # 0 are skipped by the both calculation logic.

3.4.4. ProxyPassMatch/ProxyPass

ProxyPassMatch/ProxyPass: ProxyPassMatch and ProxyPass are mod_proxy directives that when using ! (instead the back-end url) prevent to reverse-proxy in the path. This could be used allow httpd to serve static information like images.

```
ProxyPassMatch ^(/.*\.gif)$ !
```

The above for example will allow httpd to server directly the .gif files.

3.5. mod_manager

The Context of a mod_manger directive is VirtualHost except mentioned otherwise. `server config` means that it must be outside a VirtualHost configuration. If not an error message will be displayed and httpd won't start.

3.5.1. MemManagerFile

MemManagerFile: That is the base name for the names mod_manager will use to store configuration, generate keys for shared memory or lock files. That must be an absolute path name; the directories will created if needed. It is highly recommended that those files are placed on a local drive and not an NFS share. (Context: server config)

Default: $server_root/logs/

3.5.2. Maxcontext

Maxcontext: That is the number max of contexts supported by mod_cluster. (Context: server config)

Default: 100

3.5.3. Maxnode

Maxnode: That is the number max nodes supported by mod_cluster. (Context: server config)

Default: 20

3.5.4. Maxhost

Maxhost: That is the number max host (Aliases) supported by mod_cluster. That is also the max number of balancers. (Context: server config)

Default: 10

3.5.5. Maxsessionid

Maxsessionid: That is the number of active sessionid we store to give number of active sessions in the mod_cluster-manager handler. A session is unactive when mod_cluster doesn't receive any information from the session in 5 minutes. (Context: server config)

Default: 0 (the logic is not activated).

3.5.6. ManagerBalancerName

ManagerBalancerName: That is the name of balancer to use when the JBoss AS/JBossWeb/ Tomcat doesn't provide a balancer name.

Default: mycluster

3.5.7. PersistSlots

PersistSlots: Tell mod_slotmem to persist the nodes, Alias and Context in files. (Context: server config)

Default: Off

3.5.8. CheckNonce

CheckNonce: Switch check of nonce when using mod_cluster-manager handler on | off Since 1.1.0.CR1

Default: on Nonce checked

3.5.9. AllowDisplay

AllowDisplay: Switch additional display on mod_cluster-manager main page on | off Since 1.1.0.GA

Default: off Only version displayed

3.5.10. AllowCmd

AllowCmd: Allow commands using mod_cluster-manager URL on | off Since 1.1.0.GA

Default: on Commmands allowed

3.5.11. ReduceDisplay

ReduceDisplay - Reduce the information the main mod_cluster-manager page to allow more nodes in the page. on | off

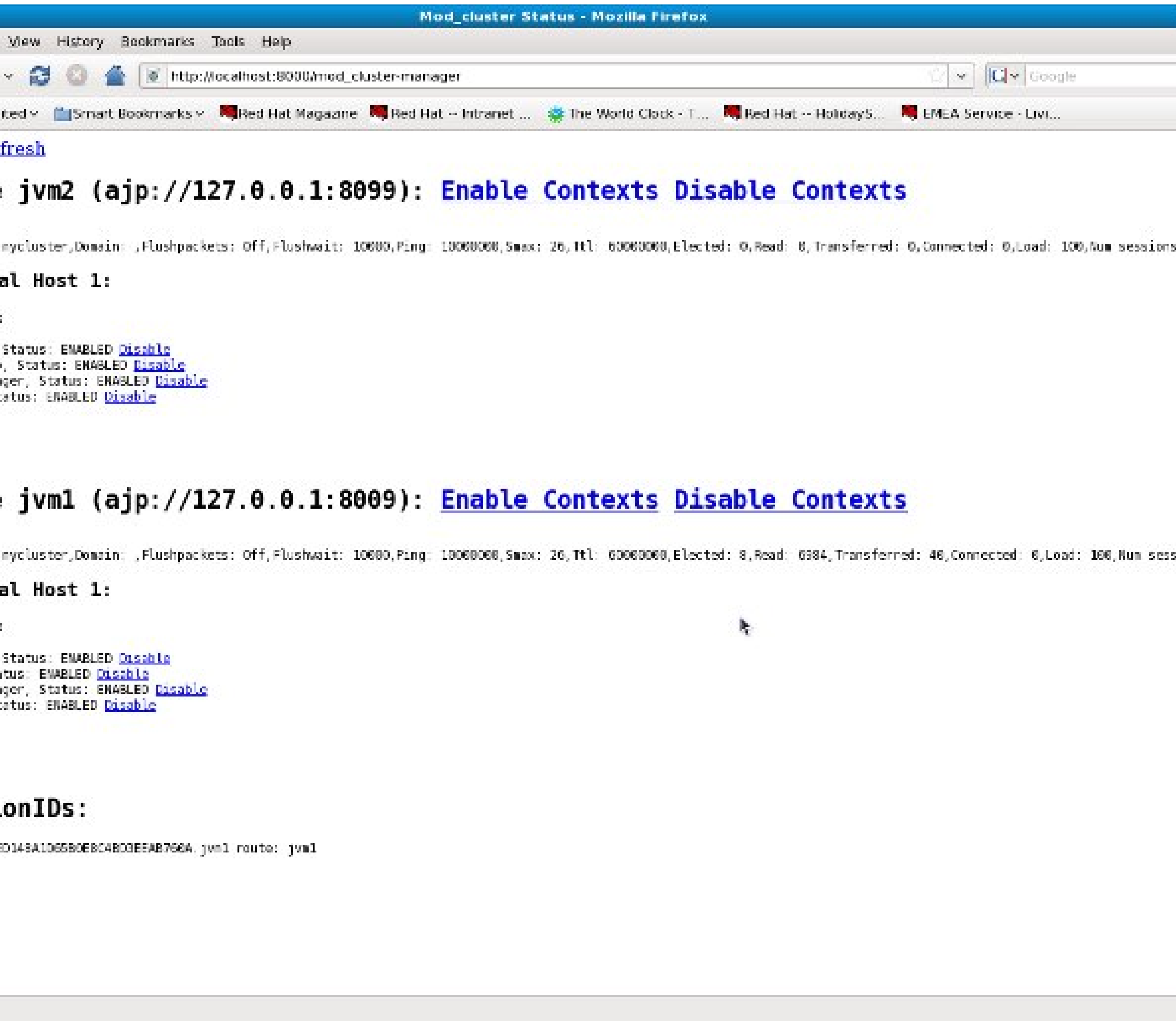
Default: off Full information displayed

3.5.12. SetHandler mod_cluster-manager

SetHandler mod_cluster-manager: That is the handler to display the node mod_cluster sees from the cluster. It displays the information about the nodes like INFO and additionaly counts the number of active sessions.

```
<Location /mod_cluster-manager>
SetHandler mod_cluster-manager
Order deny,allow
Deny from all
Allow from 127.0.0.1
</Location>
```

When accessing the location you define in httpd.conf you get something like:


Mod_cluster Status - Mozilla Firefox
View History Bookmarks Tools Help
http://localhost:8000/mod_cluster-manager
Google
ted Smart Bookmarks Red Hat Magazine Red Hat -- Intranet ... The World Clock - T... Red Hat -- HolidayS... EMEA Service - Livi...

```
fresh

 jvm2 (ajp://127.0.0.1:8099): Enable Contexts Disable Contexts

nycluster,Domain: ,Flushpackets: Off,Flushwait: 10000,Ping: 10000000,Smax: 26,Ttl: 60000000,Elected: 0,Read: 0,Transferred: 0,Connected: 0,Load: 100,Num sessions

al Host 1:

:

 Status: ENABLED Disable
, Status: ENABLED Disable
ger, Status: ENABLED Disable
tatus: ENABLED Disable

 jvm1 (ajp://127.0.0.1:8009): Enable Contexts Disable Contexts

nycluster,Domain: ,Flushpackets: Off,Flushwait: 10000,Ping: 10000000,Smax: 26,Ttl: 60000000,Elected: 8,Read: 6384,Transferred: 40,Connected: 0,Load: 100,Num sess

al Host 1:

:

 Status: ENABLED Disable
tus: ENABLED Disable
ger, Status: ENABLED Disable
tatus: ENABLED Disable

onIDs:

D143A1D65B0E8C4B03EEAB766A.jvm1 route: jvm1
```

Note that:

Transferred: Corresponds to the POST data send to the back-end server.

Connected: Corresponds to the number of requests been processed when the mod_cluster status page was requested.

sessions: Corresponds to the number of sessions mod_cluster report as active (on which there was a request during the past 5 minutes). That field is not present when Maxsessionid is zero.

3.6. mod_advertise

mod_advertise uses multicast packets to advertise the VirtualHost where it is configured that must be the same VirtualHost where mod_manager is defined. Of course at least one mod_advertise must be in the VirtualHost to allow mod_cluster to find the right IP and port to give to the ClusterListener.

3.6.1. ServerAdvertise

ServerAdvertise On: Use the advertise mechanism to tell the JBoss AS/JBossWeb/Tomcat to whom it should send the cluster information.

ServerAdvertise On http://hostname:port: Tell the hostname and port to use. Only needed if the VirtualHost is not defined correctly, if the VirtualHost is a *Name-based Virtual Host* [http://httpd.apache.org/docs/2.2/vhosts/name-based.html] or when VirtualHost is not used.

ServerAdvertise Off: Don't use the advertise mechanism.

Default: Off. (Any Advertise directive in a VirtualHost sets it to On in the VirtualHost)

3.6.2. AdvertiseGroup

AdvertiseGroup IP:port: That is the multicast address to use (something like 232.0.0.2:8888 for example). IP should correspond to AdvertiseGroupAddress and port to AdvertisePort in the JBoss AS/JBossWeb/Tomcat configuration. Note that if JBoss AS is used and the -u startup switch is included in the AS startup command, the default AdvertiseGroupAddress is the value passed via the -u. If port is missing the default port will be used: 23364.

Default: 224.0.1.105:23364.

3.6.3. AdvertiseFrequency

AdvertiseFrequency seconds[.miliseconds]: Time between the multicast messages advertising the IP and port.

Default: 10 Ten seconds.

3.6.4. AdvertiseSecurityKey

AdvertiseSecurityKey value: key string to identify the mod_cluster in JBossWEB.

Default: No default value. Information not sent.

3.6.5. AdvertiseManagerUrl

AdvertiseManagerUrl value: Not used in this version (It is sent in the X-Manager-Url: value header). That is the URL that JBoss AS/JBossWeb/Tomcat should use to send information to mod_cluster

Default: No default value. Information not sent.

3.6.6. AdvertiseBindAddress

AdvertiseBindAddress IP:port: That is the address and port httpd is bind to send the multicast messages. This allow to specify an address on multi IP address boxes.

Default: 0.0.0.0:23364

3.7. Minimal Example

```
LoadModule proxy_module modules/mod_proxy.so
 LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
 LoadModule slotmem_module modules/mod_slotmem.so
 LoadModule manager_module modules/mod_manager.so
 LoadModule proxy_cluster_module modules/mod_proxy_cluster.so
 LoadModule advertise_module modules/mod_advertise.so

 Listen 10.33.144.3:6666
 <VirtualHost 10.33.144.3:6666>

   <Location />
      Order deny,allow
      Deny from all
      Allow from 10.33.144.
   </Location>

 KeepAliveTimeout 60
 MaxKeepAliveRequests 0

 ManagerBalancerName mycluster
 ServerAdvertise On

 </VirtualHost>
```

Building httpd modules

4.1. Build a patched httpd from it sources

To build httpd-2.2.x from its sources see *ASF httpd doc* [http://httpd.apache.org/docs/2.2/install.html]

If needed, patch the httpd-2.2.x sources with (The patch prevents long waiting time when the node IP can't be resolved that should not happen so you can skip the patch part if you don't want to rebuild httpd). *mod_proxy_ajp.patch* [http://anonsvn.jboss.org/repos/mod_cluster/trunk/native/mod_proxy_cluster/mod_proxy_ajp.patch]

```
(cd modules/proxy
  patch -p0 < $location/mod_proxy_ajp.patch
 )
```

Configure httpd with something like:

```
./configure  --prefix=apache_installation_directory \
             --with-mpm=worker \
             --enable-mods-shared=most \
             --enable-maintainer-mode \
             --with-expat=builtin \
             --enable-ssl \
             --enable-proxy \
             --enable-proxy-http \
             --enable-proxy-ajp \
             --disable-proxy-balancer
```

Rebuild (make) and reinstall (make install) after that.

4.2. Build the 4 modules of mod_cluster

You need an httpd installation with mod_proxy (--enable-proxy) and ajp protocol (--enable-proxy-ajp) enabled and with dso enabled (--enable-so)

Download the mod_cluster sources:

```
svn co http://anonsvn.jboss.org/repos/mod_cluster/trunk/ mod_cluster
```

Build the mod_cluster modules components, for each subdirectory advertise, mod_manager, mod_proxy_cluster and mod_slotmem do something like:

```
sh buildconf
 ./configure --with-apxs=apxs_file_location
 make
 cp *.so apache_installation_directory/modules
```

Where apache_installation_directory is the location of an installed version of httpd-2-2.x.

NOTE: You can ignore the libtool message on most platform:

```
libtool: install: warning: remember to run `libtool --finish apache_installation_directory/modules'
```

Once that is done use *Apache httpd configuration* to configure mod_cluster.

4.3. Build the mod_proxy module

It is only needed for httpd-2.2.x where x < 11. Process like the other mod_cluster modules.

Installing httpd modules

Several bundles are available at *http://www.jboss.org/mod_cluster/downloads.html*.

In case you can't find a prepared package of mod_cluser in the download area, it is possible to build mod_cluster for the sources. You need a distribution of httpd (at least 2.2.8) or (better) a source tarball of httpd and the sources of mod_cluster. *Building* explains how to build mod_cluster for it sources.

5.1. Configuration

A minimal configuration is needed in httpd (See *httpd.conf*). A listener must be a added in in JBossWEB conf/server.xml (See *Configuring JBoss AS/Web*).

5.2. Installing and using the bundles

The bundles are tar.gz on POSIX platforms just extract them in root something like:

```
cd /
tar xvf mod-cluster-1.0.0-linux2-x86-ssl.tar.gz
```

The httpd.conf is located in /opt/jboss/httpd/httpd/conf to quick test just add something like:

```
Listen 10.33.144.3:6666
 <VirtualHost 10.33.144.3:6666>

   <Directory />
      Order deny,allow
      Deny from all
      Allow from 10.33.144.
   </Directory>

 KeepAliveTimeout 60
 MaxKeepAliveRequests 0

 ManagerBalancerName mycluster
 AdvertiseFrequency 5

 </VirtualHost>
```

To start httpd do the following:

```
/opt/jboss/httpd/sbin/apachectl start
```

NOTE: Make sure to use SSL before going in production.

Server-side Configuration

6.1. JBoss AS

mod_cluster is supported in AS7 via the modcluster subsystem See *AS7*.

In other AS version mod_cluster's configuration resides within the following file:

`$JBOSS_HOME/server/$PROFILE/deploy/mod_cluster.sar/META-INF/mod_cluster-jboss-beans.xml` file.

The entry point for mod_cluster's server-side configuration is the `ModClusterListener` bean, which delegates web container (i.e. JBoss Web) specific events to a container agnostic event handler.

In general, the `ModClusterListener` bean defines:

1. A `ContainerEventHandler` in which to handle events from the web container. There are two available implementations, the choice of which dictates the mode in which mod_cluster will operate: *clustered* or *non-clustered*.
2. A reference to the JBoss mbean server.

e.g.

```
<bean name="ModClusterListener"
 class="org.jboss.modcluster.catalina.CatalinaEventHandlerAdapter">
  <constructor>
    <parameter class="org.jboss.modcluster.ContainerEventHandler">
      <inject bean="ModClusterService"/><!-- Non-clustered mode -->
      <!--inject bean="HAModClusterService"/--><!-- Clustered mode -->
    </parameter>
    <parameter class="javax.management.MBeanServer">
      <inject bean="JMXKernel" property="mbeanServer"/>
    </parameter>
  </constructor>
</bean>
```

6.1.1. Non-clustered mode

In non-clustered mode, each JBoss AS node communicates with the load balancer directly, and do not communicate with each other. Non-clustered mode is configured via the `ModClusterService` bean.

In general, the `ModClusterService` bean defines:

1. An object containing mod_cluster's *configuration properties*.
2. An object responsible for calculating the load balance factor for this node. This is described in detail in the chapter entitled *Server-Side Load Metrics*.

e.g.

```
<bean name="ModClusterService" class="org.jboss.modcluster.ModClusterService" mode="On
 Demand">



annotation>
  <constructor>
    <parameter class="org.jboss.modcluster.config.ModClusterConfig">
      <inject bean="ModClusterConfig"/>
    </parameter>
    <parameter class="org.jboss.modcluster.load.LoadBalanceFactorProvider">
      <inject bean="DynamicLoadBalanceFactorProvider"/>
    </parameter>
  </constructor>
</bean>
```

6.1.2. Clustered mode

In clustered mode, a single JBoss node (the HA singleton master) communicates with the load balancer on behalf of the other nodes in the cluster. Clustered mode is configured via the `HAModClusterService` bean. This mode offers the following advantages over non-clustered mode:

1. The state of the load balancer will be kept in sync on each node in the cluster.
2. mod_cluster will proactively notify the load balancer of view changes (i.e. crashed members), allowing the load balancer to gracefully reconfigure itself thus avoiding costly failover processing.
3. Clustered-mode allows load balancing group management, adding the ability to enable, disable, or gracefully stop all nodes sharing the same load balancing group. Once a given session is associated with a specific node, subsequent requests for the same session will always prefer to be routed to a node with the same load balancing group. Grouping nodes in this way is useful for limiting the number of nodes to which a given session must replicate to support high-availability. When used in conjunction with sessions affinity, a load balancing group effectively narrows the set of preferred failover nodes to members of the same load balancing group. A load balancing group is conceptually similar to a *domain* in mod_jk.

In general, HAModClusterService defines:

1. An object containing mod_cluster's *configuration properties*.
2. An object responsible for calculating the load balance factor for this node. This is described in detail in the chapter entitled *Server-Side Load Metrics*.
3. An `HAPartition`, the JBoss clustering group communication building block. The default HAPartition is defined in: `$JBOSS_HOME/server/$PROFILE/deploy/cluster/hapartition-jboss-beans.xml`
4. A policy for determining which group member should be designated as the singleton master.

e.g.

```
<bean name="HAModClusterService" class="org.jboss.modcluster.ha.HAModClusterService"
 mode="On Demand">



annotation>
  <constructor>
    <parameter class="org.jboss.modcluster.config.ha.HAModClusterConfig">
      <inject bean="ModClusterConfig"/>
    </parameter>
    <parameter class="org.jboss.modcluster.load.LoadBalanceFactorProvider">
      <inject bean="DynamicLoadBalanceFactorProvider"/>
    </parameter>
    <parameter class="org.jboss.ha.framework.interfaces.HAPartition">
      <inject bean="HAPartition"/>
    </parameter>
    <parameter class="org.jboss.ha.framework.interfaces.HASingletonElectionPolicy">
      <bean class="org.jboss.ha.singleton.HASingletonElectionPolicySimple"/>
    </parameter>
  </constructor>
</bean>
```

6.1.3. Configuration Properties

The `ModClusterConfig` bean enumerates the configuration properties used by mod_cluster. For the complete list of configuration properties and their default values, see the chapter entitled *Server-side Configuration Properties*.

e.g.

```
<bean  name="ModClusterConfig"  class="org.jboss.modcluster.config.ha.HAModClusterConfig"
 mode="On Demand">
  <!-- Specify configuration properties here -->
</bean>
```

6.1.4. Connectors

Like mod_jk and mod_proxy_balancer, mod_cluster requires a connector in your server.xml to which to forward web requests. Unlike mod_jk and mod_proxy_balancer, mod_cluster is not confined to AJP, but can use HTTP as well. While AJP is generally faster, an HTTP connector can optionally be secured via SSL. If multiple possible connectors are defined in your server.xml, mod_cluster uses the following algorithm to choose between them:

1. If an native (APR) AJP connector is available, use it.
2. If an AJP connector is available, use it.
3. Otherwise, choose the HTTP connector with the highest max threads.

6.1.5. Node Identity

Like mod_jk and mod_proxy_balancer, mod_cluster identifies each node via a unique *jvm route* [http://docs.jboss.org/jbossweb/2.1.x/config/engine.html]. By default, mod_cluster uses the following algorithm to assign the jvm route for a given node:

1. Use the value from `server.xml`, `<Engine jvmRoute="..."/>`, if defined.
2. Generate a jvm route using the configured ???. The default implementation does the following:
   a. Use the value of the `jboss.mod_cluster.jvmRoute` system property, if defined.
   b. Generate a UUID.

While UUIDs are ideal for production systems, in a development or testing environment, it is useful to know which node served a given request just by looking at the jvm route. In this case, you can utilize the `org.jboss.modcluster.SimpleJvmRouteFactory`. The factory generates jvm routes of the form:

bind-address:*port*:*engine-name*

6.2. JBoss Web & Tomcat

mod_cluster's entire configuration for JBoss Web or Tomcat resides entirely within `$CATALINA_HOME/conf/server.xml`.

This limits the adds the following constraints to mod_cluster's feature set:

- Only non-clustered mode is supported
- *Only one load metric* can be used to calculate a load factor.

6.2.1. Lifecycle Listener

The entry point for JBoss Web and Tomcat configuration is the ModClusterListener. All mod_cluster *configuration properties* are defined as attributes of the `<Listener/>` element. For the complete list of configuration properties and their default values, see the chapter entitled *Server-side Configuration Properties*.

e.g.

```
<Listener className="org.jboss.modcluster.catalina.ModClusterListener" advertise="true"/>
```

6.2.2. Additional Tomcat dependencies

mod_cluster uses jboss-logging, which exists in JBoss Web, but not in Tomcat. Consequently, to use mod_cluster with Tomcat, it is necessary to add *jboss-logging-spi.jar* [http://repository.jboss.org/nexus/content/groups/public-jboss/org/jboss/logging/jboss-logging-spi/] to $CATALINA_HOME/lib.

6.3. Migrating from 1.0.x

In mod_cluster 1.0.x, you were required to make several manual configuration changes to the jbossweb service before mod_cluster would be usable. mod_cluster 1.1.x eliminates much of this hassle - and is designed to be fully functional out of the box.

6.3.1. Dependency with JBoss Web

In 1.0.x, mod_cluster needed to be deployed *before* JBoss Web. This ensured that mod_cluster was available to handle web application deployment lifecycle events during triggered during server startup. To achieve this, you had to add an explicit dependency on mod_cluster to jbossweb's jboss-beans.xml config file.

mod_cluster 1.1.x reverses this dependency, such that mod_cluster now depends on JBoss Web. Consequently, you no longer need to make any changes to JBoss Web's jboss-beans.xml file.

As an additional consequence of this dependency reversal, mod_cluster can now be hot-deployed or undeployed to a running server without consequence to the JBoss Web service.

6.3.2. server.xml

6.3.2.1. Lifecycle listener

mod_cluster 1.0.x required you to add a lifecycle <Listener/> element to server.xml.

6.3.2.1.1. JBoss AS

The *ModClusterListener* bean now registers itself with the JBoss Web server upon deploying the mod_cluster service; and deregisters itself upon undeploy. Consequently, you no longer need to manually add a <Listener/> to server.xml.

6.3.2.1.2. JBoss Web & Tomcat

You still need to use a lifecycle <Listener/> element in server.xml, but it's location has changed. In 1.1.x, the ModClusterListener class was refactored into the catalina container SPI implementation package, and the old location was deprecated.

6.3.2.2. JVM Route

In mod_cluster 1.0.x, you needed to add a unique jvmRoute to the <Engine/> element within server.xml. In 1.1.x, this is now optional. The auto-assignment of a jvm route is described in the *previous section*.

6.3.2.3. Connector bind address

In mod_cluster 1.0.x, you had the option of manually setting the connector bind address, to restrict the network interface on which mod_cluster would listen for proxied requests from the load balancer. In mod_cluster 1.1.x, this no longer necessary. mod_cluster will set the bind address automatically, based on the interface on which mod_cluster's internal connection was established.

6.3.3. mod_cluster-jboss-beans.xml

The JBoss microcontainer configuration for mod_cluster 1.0.x is *not* compatible with the configuration for 1.1.x. If you had customized your mod_cluster 1.0.x configuration, please start with the default configuration contained in the 1.1.x mod_cluster.sar and reapply any changes.

6.3.3.1. Configuration changes

The following configuration properties were renamed between 1.0.x and 1.1.x:

Deprecated property name	Replacement property name
domain	loadBalancingGroup
masterPerDomain	masterPerLoadBalancingGroup
sslKeyStorePass	sslKeyStorePassword

AS7 modcluster subsystem Configuration

7.1. ModCluster Subsystem in JBoss AS7

The mod_cluster integration is done via the modcluster subsystem. In AS7 only 1.1.x is supported. Addition HA is not supported in AS version 7.0.x but in it will be in version 7.1.x and after.

7.2. ModCluster Subsystem minimal configuration

The minimal configuration is having the modcluster `schemaLocation` in the `schemaLocation` list:

```
urn:jboss:domain:modcluster:1.0 jboss-mod-cluster.xsd
```

and the `extension module` in the `extensions` list:

```
<extension module="org.jboss.as.modcluster"/>
```

and `subsystem` declaration like:

```
<subsystem xmlns="urn:jboss:domain:modcluster:1.0"/>
```

With that configuration modcluster will listen for advertise on `224.0.1.105:23364` use the `simple-load-provider` with a load factor of 1.

7.3. ModCluster Subsystem configuration

7.3.1. mod-cluster-config Attributes

The attributs correspond to the *properties*

Attribute	Property	Default
proxy-list	proxyList	*None*
proxy-url	proxyURL	*None*
advertise	advertise	*true*
advertise-security-key	advertiseSecurityKey	*None*
excluded-contexts	excludedContexts	*None*

Attribute	Property	Default
auto-enable-contexts	autoEnableContexts	*true*
stop-context-timeout	stopContextTimeout	*10 seconds (in seconds)*
socket-timeout	nodeTimeout	*20 seconds (in milli seconds)*

SSL configuration part needs to be added here too

7.3.2. simple-load-provider Attributes

The simple load provider always send the same load factor. That is the default one. Example:

```
<subsystem xmlns="urn:jboss:domain:modcluster:1.0">
  <mod-cluster-config>
    <simple-load-provider load="1"/>
  </mod-cluster-config>
</subsystem>
```

Attribute	Property	Default
factor	LoadBalancerFactor	*1*

7.3.3. dynamic-load-provider Attributes

The dynamic load provide allows to have `load-metric` as well as `custom-load-metric` defined. For example:

```
<subsystem xmlns="urn:jboss:domain:modcluster:1.0">
  <mod-cluster-config advertise-socket="mod_cluster">
    <dynamic-load-provider history="10" decay="2">
       <load-metric type="cpu" weight="2" capacity="1"/>
       <load-metric type="sessions" weight="1" capacity="512"/>
       <custom-load-metric class="mypackage.myclass" weight="1" capacity="512">
          <property name="myproperty" value="myvalue" />
          <property name="otherproperty" value="othervalue" />
       </custom-load-metric>
    </dynamic-load-provider>
  </mod-cluster-config>
</subsystem>
```

Attribute	Property	Default
history	history	512
decay	decayFactor	512

7.3.4. load-metric Configuration

The load-metric are the "classes" collecting information to allow computation of the load factor sent to httpd

Attribute	Property	Default
type	A Server-Side Load Metrics	*Mandatory*
weight	weight	*9*
capacity	capacity	*512*

7.3.4.1. Supported load metric types

type	Corresponding Server-Side Load Metric
cpu	*AverageSystem*LoadMetric
mem	*SystemMemoryUsage*LoadMetric
heap	*HeapMemoryUsage*LoadMetric
sessions	*ActiveSessions*LoadMetric
requests	*ActiveSessions*LoadMetric
send-traffic	*SendTraffic*LoadMetric
receive-traffic	*ReceiveTraffic*LoadMetric
busyness	*BusyConnectors*LoadMetric
connection-pool	*ConnectionPoolUsage*LoadMetric

7.3.5. custom-load-metric Configuration

The custom-load-metric are for user defined "classes" collecting information. They are like the load-metric except `type` is replaced by `class`:

Attribute	Property	Default
class	Name of your class	*Mandatory*

Building Server-Side Components

8.1. Requirements

Building mod_cluster's server-side components from source requires the following tools:

- JDK 5.0+
- Maven 2.0+

8.2. Building

Steps to build:

1. Download the mod_cluster sources

```
svn co http://anonsvn.jboss.org/repos/mod_cluster/trunk/ mod_cluster
```

2. Use maven "dist" profile to build:

```
cd mod_cluster
mvn -P dist package
```

Note

Some unit tests require UDP port 23365. Make sure your local firewall allows the port.

8.3. Build Artifacts

The build produces the following output in the target directory:

mod-cluster.sar
: Exploded format sar to copy to the deploy dir in your JBoss AS install

JBossWeb-Tomcat/lib directory
: Jar files to copy to the lib directory in your JBossWeb or Tomcat install to support use of mod_cluster

demo directory
: The *load balancing demo* application

mod-cluster-XXX.tar.gz
: The full distribution tarball; includes the above elements

mod-cluster-XXX.tar.gz

Server-side Configuration Properties

The tables below enumerate the configuration properties available to an application server node. The location for these properties depends on *how mod_cluster is configured*.

9.1. Proxy Discovery Configuration

The list of proxies from which an application expects to receive AJP connections is either defined statically, via the addresses defined in the *proxyList* configuration property; or discovered dynamically via the advertise mechanism. Using a special mod_advertise module, proxies can advertise their existence by periodically broadcasting a multicast message containing its address/ port. This functionality is enabled via the *advertise* configuration property. If configured to listen, a server can learn of the proxy's existence, then notify that proxy of its own existence, and update its configuration accordingly. This frees both the proxy *and* the server from having to define static, environment-specific configuration values.

Attribute	Default	Scope	Description
proxyList	*None*	Configuration	Defines a comma delimited list of httpd proxies with which this node will initially communicate. Value should be of the form: *address1*:*port1*,*address2*:*port2* Using the default configuration, this property can by manipulated the jboss.mod_cluster.proxyList system property.
excludedContexts	ROOT, admin-console, invoker, bossws, jmx-console, juddi, web-console	Configuration	List of contexts to exclude from httpd registration, of the form: *host1*:*context1*,*host2*:*context2*,*host3*:*conte* If no host is indicated, it is assumed to be the default host of the server (e.g. localhost). "ROOT" indicates the root context. Using the

Attribute	Default	Scope	Description
			default configuration, this property can by manipulated via the jboss.mod_cluster.excludedContexts system property.
autoEnableContexts	true	Configuration	If false the contexts are registered disabled in httpd, they need to be enabled via the enable() mbean method or via mod_cluster_manager.
stopContextTimeout	10	Configuration	The amount of time, measure in units specified by *stopContextTimeoutUnit*, for which to wait for clean shutdown of a context (completion of pending requests for a distributable context; or destruction/ expiration of active sessions for a non-distributable context).
stopContextTimeoutUnit	TimeUnit.SECONDS	Configuration	The unit of time for use with *stopContextTimeout*
sessionDrainingStrategy	org.jboss.modcluster.SessionDrainingStrategyEnum.DEFAULT	Configuration	Indicates the session draining strategy used during undeployment of a web application. There are three possible values: DEFAULT Drain sessions before web application undeploy only

Attribute	Default	Scope	Description
			if the web application is non-disributable. ALWAYS Always drain sessions before web application undeploy, even for distributable web applications. NEVER Do not drain sessions before web application undeploy, even for non-distributable web application.
proxyURL	*None*	Apache HTTPD	If defined, this value will be prepended to the URL of MCMP commands.
socketTimeout	20000	Configuration	Number of milliseconds to wait for a response from an httpd proxy to MCMP commands before timing out, and flagging the proxy as in error.
advertise	true, if *proxyList* is undefined, false otherwise	Configuration	If enabled, httpd proxies will be auto-discovered via multicast announcements. This can be used either in concert or in place of a static *proxyList*.
advertiseGroupAddress	224.0.1.105	Apache HTTPD	UDP address on which to listen for

Attribute	Default	Scope	Description
			httpd proxy multicast advertisements
advertisePort	23364	Apache HTTPD	UDP port on which to listen for httpd proxy multicast advertisements
advertiseSecurityKey	*None*	Apache HTTPD	If specified, httpd proxy advertisements checksums will be verified using this value as a salt
advertiseThreadFactory	Executors.defaultThreadFactory()	Configuration	The thread factory used to create the background advertisement listener.
jvmRouteFactory	new SystemPropertyJvmRouteFactory(new UUIDJvmRouteFactory(), "jboss.mod_cluster.jvmRoute")	Configuration	Defines the strategy for determing the jvm route of a node, if none was specified in server.xml. The default factory first consults the `jboss.mod_cluster.jvmRoute` system property. If this system property is undefined, the jvm route is assiged a UUID.

9.2. Proxy Configuration

The following configuration values are sent to proxies during server startup, when a proxy is detected via the advertise mechanism, or during the resetting of a proxy's configuration during error recovery.

Attribute	Default	Scope	Description
stickySession	true	Balancer	Indicates whether subsequent requests for a given session should be routed to

Attribute	Default	Scope	Description
			the same node, if possible.
stickySessionRemove	false	Balancer	Indicates whether the httpd proxy should remove session stickiness in the event that the balancer is unable to route a request to the node to which it is stuck. This property is ignored if *stickySession* is false.
stickySessionForce	true	Balancer	Indicates whether the httpd proxy should return an error in the event that the balancer is unable to route a request to the node to which it is stuck. This property is ignored if *stickySession* is false.
workerTimeout	-1	Balancer	Number of seconds to wait for a worker to become available to handle a request. When all the workers of a balancer are usable, mod_cluster will retry after a while (workerTimeout/100) to find an usable worker. That is timeout in the balancer mod_proxy documentation. A value of -1 indicates that the httpd will not wait for a worker to be available and will return an error if none is available.

Attribute	Default	Scope	Description
maxAttempts	1	Balancer	Number of times an httpd proxy will attempt to send a given request to a worker before giving up. The minimum value is 1, meaning try only once. (Note that mod_proxy default is also 1: no retry).
flushPackets	false	Node	Enables/disables packet flushing
flushWait	-1	Node	Time to wait before flushing packets. A value of -1 means wait forever.
ping	10	Node	Time (in seconds) in which to wait for a pong answer to a ping
smax	Determined by httpd configuration	Node	Soft maximum idle connection count (that is the smax in worker mod_proxy documentation). The maximum value depends on the httpd thread configuration (ThreadsPerChild or 1).
ttl	60	Node	Time to live (in seconds) for idle connections above smax
nodeTimeout	-1	Node	Timeout (in seconds) for proxy connections to a node. That is the time mod_cluster will wait for the back-end response before returning error. That corresponds to

Attribute	Default	Scope	Description
			timeout in the worker mod_proxy documentation. A value of -1 indicates no timeout. Note that mod_cluster always uses a cping/cpong before forwarding a request and the connectiontimeout value used by mod_cluster is the ping value.
balancer	mycluster	Node	The balancer name
loadBalancingGroup	*None*	Node	If specified, load will be balanced among jvmRoutes withing the same load balancing group. A loadBalancingGroup is conceptually equivalent to a mod_jk domain directive. This is primarily used in conjunction with partitioned session replication (e.g. buddy replication).

Note

When nodeTimeout is not defined the ProxyTimeout directive Proxy is used. If ProxyTimeout is not defined the server timeout (Timeout) is used (default 300 seconds). nodeTimeout, ProxyTimeout or Timeout is set at the socket level.

9.3. SSL Configuration

The communication channel between application servers and httpd proxies uses HTTP by default. This channel can be secured using HTTPS by setting the *ssl* property to true.

Note

This HTTP/HTTPS channel should not be confused with the processing of HTTP/HTTPS client requests.

Attribute	Default	Description
ssl	false	Should connection to proxy use a secure socket layer
sslCiphers	*The default JSSE cipher suites*	Overrides the cipher suites used to init an SSL socket ignoring any unsupported ciphers
sslProtocol	TLS	Overrides the default SSL socket protocol.
sslCertificateEncodingAlgorithm	*The default JSSE key manager algorithm*	The algorithm of the key manager factory
sslKeyStore	`System.getProperty("user.home") + "/.keystore"`	The location of the key store containing client certificates
sslKeyStorePassword	changeit	The password granting access to the key store
sslKeyStoreType	JKS	The type of key store
sslKeyStoreProvider	*The default JSSE security provider*	The key store provider
sslTrustAlgorithm	*The default JSSE trust manager algorithm*	The algorithm of the trust manager factory
sslKeyAlias		The alias of the key holding the client certificates in the key store
sslCrlFile		Certificate revocation list
sslTrustMaxCertLength	5	The maximum length of a certificate held in the trust store
sslTrustStore	`System.getProperty("javax.net.ssl.trustStorePassword")`	The location of the file containing the trust store
sslTrustStorePassword	`System.getProperty("javax.net.ssl.trustStore")`	The password granting access to the trust store.
sslTrustStoreType	`System.getProperty("javax.net.ssl.trustStoreType")`	The trust store type
sslTrustStoreProvider	`System.getProperty("javax.net.ssl.trustStoreProvider")`	The trust store provider

9.4. HA Configuration

Additional configuration properties when mod_cluster is configured in clustered mode.

Attribute	Default	Description
masterPerLoadBalancingGroup	false	If the *loadBalancingGroup* directive is used, should HA partition use a singleton master per loadBalancingGroup.

9.5. Load Configuration for JBoss Web and Tomcat

Additional configuration properties used when mod_cluster is configured in JBoss Web standalone or Tomcat.

Attribute	Default	Description
loadMetricClass	org.jboss.modcluster.load.metric.impl.BusyConnectorsLoadMetric	Class name of an object implementing org.jboss.load.metric.LoadMetric
loadMetricCapacity	1	The capacity of the load metric defined via the loadMetricClass property
loadHistory	9	The number of historic load values to consider in the load balance factor computation.
loadDecayFactor	2	The factor by which a historic load values should degrade in significance.

Server-Side Load Metrics

A major feature of mod_cluster is the ability to use server-side load metrics to determine how best to balance requests.

The `DynamicLoadBalanceFactorProvider` bean computes the load balance factor of a node from a defined set of load metrics.

```
<bean name="DynamicLoadBalanceFactorProvider" class="org.jboss.modcluster.load.impl.DynamicLoadBalanceFa

ceFactorProviderMBean.class)</
annotation>
  <constructor>
    <parameter>
      <set elementClass="org.jboss.modcluster.load.metric.LoadMetric">
        <inject bean="BusyConnectorsLoadMetric"/>
        <inject bean="HeapMemoryUsageLoadMetric"/>
      </set>
    </parameter>
  </constructor>
  <property name="history">9</property>
  <property name="decayFactor">2</property>
</bean>
```

Load metrics can be configured with an associated weight and capacity.

The weight (default is 1) indicates the significance of a metric with respect to the other metrics. For example, a metric of weight 2 will have twice the impact on the overall load factor than a metric of weight 1.

The capacity of a metric serves 2 functions:

- To normalize the load values from each metric. In some load metrics, capacity is already reflected in the load values. The capacity of a metric should be configured such that 0 <= (load / capacity) >= 1.
- To favor some nodes over others. By setting the metric capacities to different values on each node, proxies will effectively favor nodes with higher capacities, since they will return smaller load values. This adds an interesting level of granularity to node weighting. Consider a cluster of two nodes, one with more memory, and a second with a faster CPU; and two metrics, one memory-based and the other CPU-based. In the memory-based metric, the first node would be given a higher load capacity than the second node. In a CPU-based metric, the second node would be given a higher load capacity than the first node.

Each load metric contributes a value to the overall load factor of a node. The load factors from each metric are aggregated according to their weights.

In general, the load factor contribution of given metric is: (load / capacity) * weight / total weight.

The DynamicLoadBalanceFactorProvider applies a time decay function to the loads returned by each metric. The aggregate load, with respect to previous load values, can be expressed by the following formula:

$L = (L_0 + L_1/D + L_2/D^2 + L_3/D^3 + ... + L_H/D^H) * (1 + D + D^2 + D^3 + ... D^H)$

... or more concisely as:

$L = (\#^H_{i=0} L_i/D^i) * (\#^H_{i=0} D^i)$

... where D = decayFactor, and H = history.

Setting history = 0 effectively disables the time decay function and only the current load for each metric will be considered in the load balance factor computation.

The mod_cluster load balancer expects the load factor to be an integer between 0 and 100, where 0 indicates max load and 100 indicates zero load. Therefore, the final load factor sent to the load balancer

$L_{Final} = 100 - (L * 100)$

While you are free to write your own load metrics, the following LoadMetrics are available out of the box:

10.1. Web Container metrics

10.1.1. `ActiveSessionsLoadMetric`

- Requires an explicit capacity
- Uses `SessionLoadMetricSource` to query session managers
- Analogous to method=S in mod_jk

e.g.

```
<bean name="ActiveSessionsLoadMetric" class="org.jboss.modcluster.load.metric.impl.ActiveSessionsLoadMetric"
class)</
annotation>
  <constructor>
    <parameter><inject bean="SessionLoadMetricSource"/></parameter>
  </constructor>
  <property name="capacity">1000</property>
</bean>
<bean name="SessionLoadMetricSource" class="org.jboss.modcluster.load.metric.impl.SessionLoadMetricSource"
  <constructor>
    <parameter class="javax.management.MBeanServer">
      <inject bean="JMXKernel" property="mbeanServer"/>
    </parameter>
  </constructor>
</bean>
```

10.1.2. `BusyConnectorsLoadMetric`

- Returns the percentage of connector threads from the thread pool that are busy servicing requests
- Uses `ThreadPoolLoadMetricSource` to query connector thread
- Analogous to method=B in mod_jk

e.g.

```
<bean name="BusyConnectorsLoadMetric" class="org.jboss.modcluster.load.metric.impl.BusyConnectorsLoadMetr

.class)</

annotation>
  <constructor>
    <parameter><inject bean="ThreadPoolLoadMetricSource"/></parameter>
  </constructor>
</bean>
<bean name="ThreadPoolLoadMetricSource" class="org.jboss.modcluster.load.metric.impl.ThreadPoolLoadMetricS
  <constructor>
    <parameter class="javax.management.MBeanServer">
      <inject bean="JMXKernel" property="mbeanServer"/>
    </parameter>
  </constructor>
</bean>
```

10.1.3. `ReceiveTrafficLoadMetric`

- Returns the incoming request POST traffic in KB/sec (the application needs to read POST data)
- Requires an explicit capacity

- Uses `RequestProcessorLoadMetricSource` to query request processors
- Analogous to method=T in mod_jk

e.g.

```
<bean name="ReceiveTrafficLoadMetric" class="org.jboss.modcluster.load.metric.impl.ReceiveTrafficLoadMetric" m

ass)</

annotation>
  <constructor>

 <parameter class="org.jboss.modcluster.load.metric.impl.RequestProcessorLoadMetricSource">
      <inject bean="RequestProcessorLoadMetricSource"/>
    </parameter>
  </constructor>
  <property name="capacity">1024</property>
</bean>
<bean name="RequestProcessorLoadMetricSource" class="org.jboss.modcluster.load.metric.impl.RequestProcess
  <constructor>
    <parameter class="javax.management.MBeanServer">
      <inject bean="JMXKernel" property="mbeanServer"/>
    </parameter>
  </constructor>
</bean>
```

10.1.4. `SendTrafficLoadMetric`

- Returns the outgoing request traffic in KB/sec
- Requires an explicit capacity
- Uses `RequestProcessorLoadMetricSource` to query request processors
- Analogous to method=T in mod_jk

```
<bean name="SendTrafficLoadMetric" class="org.jboss.modcluster.load.metric.impl.SendTrafficLoadMetric" mode=

s)</

annotation>
  <constructor>

 <parameter class="org.jboss.modcluster.load.metric.impl.RequestProcessorLoadMetricSource">
      <inject bean="RequestProcessorLoadMetricSource"/>
```

```
    </parameter>
  </constructor>
  <property name="capacity">512</property>
</bean>
```

10.1.5. `RequestCountLoadMetric`

- Returns the number of requests/sec
- Requires an explicit capacity
- Uses `RequestProcessorLoadMetricSource` to query request processors
- Analogous to method=R in mod_jk

e.g.

```
<bean name="RequestCountLoadMetric" class="org.jboss.modcluster.load.metric.impl.RequestCountLoadMetric" m
ass)</
annotation>
  <constructor>
 <parameter class="org.jboss.modcluster.load.metric.impl.RequestProcessorLoadMetricSource">
      <inject bean="RequestProcessorLoadMetricSource"/>
    </parameter>
  </constructor>
  <property name="capacity">1000</property>
</bean>
```

10.2. System/JVM metrics

10.2.1. `AverageSystemLoadMetric`

- Returns CPU load
- Requires Java 1.6+
- Uses `OperatingSystemLoadMetricSource` to generically read attributes

e.g.

```
<bean name="AverageSystemLoadMetric" class="org.jboss.modcluster.load.metric.impl.AverageSystemLoadMetric
```

class)</

```
annotation>
  <constructor>
    <parameter><inject bean="OperatingSystemLoadMetricSource"/></parameter>
  </constructor>
</bean>
<bean name="OperatingSystemLoadMetricSource" class="org.jboss.modcluster.load.metric.impl.OperatingSystemL
</bean>
```

10.2.2. `SystemMemoryUsageLoadMetric`

- Returns system memory usage
- Requires com.sun.management.OperatingSystemMXBean (available in Sun's JDK or OpenJDK)
- Uses OperatingSystemLoadMetricSource to generically read attributes

e.g.

Bean.class)</

```
<bean name="SystemMemoryUsageLoadMetric" class="org.jboss.modcluster.load.metric.impl.SystemMemoryUsag


annotation>
  <constructor>
    <parameter><inject bean="OperatingSystemLoadMetricSource"/></parameter>
  </constructor>
</bean>
```

10.2.3. `HeapMemoryUsageLoadMetric`

- Returns the heap memory usage as a percentage of max heap size

e.g.

ean.class)</

```
<bean name="HeapMemoryUsageLoadMetric" class="org.jboss.modcluster.load.metric.impl.HeapMemoryUsageLo


annotation>
</bean>
```

10.3. Other metrics

10.3.1. `ConnectionPoolUsageLoadMetric`

- Returns the percentage of connections from a connection pool that are in use
- Uses ConnectionPoolLoadMetricSource to query JCA connection pools

e.g.

```
<bean name="ConnectionPoolUsageMetric" class="org.jboss.modcluster.load.metric.impl.ConnectionPoolUsageLoa

lBean.class)</

annotation>
  <constructor>
    <parameter><inject bean="ConnectionPoolLoadMetricSource"/></parameter>
  </constructor>
</bean>
<bean name="ConnectionPoolLoadMetricSource" class="org.jboss.modcluster.load.metric.impl.ConnectionPoolLoa
  <constructor>
    <parameter class="javax.management.MBeanServer">
      <inject bean="JMXKernel" property="mbeanServer"/>
    </parameter>
  </constructor>
</bean>
```

Installing Server-Side Components

First, extract the server-side binary to a temporary directory. The following assumes it was extracted to /tmp/mod_cluster

Your next step depends on whether your target server is JBoss AS or JBossWeb/Tomcat.

11.1. Installing in JBoss AS 6.0.0.M1 and up

You don't need to do anything to install the java-side binaries in AS 6.x; it's part of the AS distribution's default, standard and all configurations.

11.2. Installing in JBoss AS 5.x

Assuming $JBOSS_HOME indicates the root of your JBoss AS install and that you want to use mod_cluster in the AS's all config:

```
cp -r /tmp/mod_cluster/mod_cluster.sar $JBOSS_HOME/server/all/deploy
```

11.3. Installing in JBoss Web or Tomcat

Assuming $CATALINA_HOME indicates the root of your JBossWeb or Tomcat install:

```
cp -r /tmp/mod_cluster/JBossWeb-Tomcat/lib/* $CATALINA_HOME/lib/
```

11.4. Installing in JBoss AS 4.2.x or 4.3.x

Assuming $JBOSS_HOME indicates the root of your JBoss AS install and that you want to use mod_cluster in the AS's all config:

```
cp -r /tmp/mod_cluster/JBossWeb-Tomcat/lib/mod_cluster.jar $JBOSS_HOME/server/all/deploy/
jboss-web.deployer
```

Using SSL in mod_cluster

There are 2 connections between the cluster and the front-end. Both could be encrypted. That chapter describes how to encrypt both connections.

12.1. Using SSL between JBossWEB and httpd

As the ClusterListener allows to configure httpd it is adviced to use SSL for that connection. The most easy is to use a virtual host that will only be used to receive information from JBossWEB. Both side need configuration.

12.1.1. Apache httpd configuration part

mod_ssl [http://httpd.apache.org/docs/2.2/mod/mod_ssl.html] of httpd is using to do that. See in one example how easy the configuration is:

```
Listen 6666
<VirtualHost 10.33.144.3:6666>
   SSLEngine on
   SSLCipherSuite AES128-SHA:ALL:!ADH:!LOW:!MD5:!SSLV2:!NULL
   SSLCertificateFile conf/server.crt
   SSLCertificateKeyFile conf/server.key
   SSLCACertificateFile conf/server-ca.crt
   SSLVerifyClient require
   SSLVerifyDepth  10
</VirtualHost>
```

The conf/server.crt file is the PEM-encoded Certificate file for the VirtualHost it must be signed by a Certificate Authority (CA) whose certificate is stored in the sslTrustStore of the ClusterListener parameter.

The conf/server.key file is the file containing the private key.

The conf/server-ca.crt file is the file containing the certicate of the CA that have signed the client certificate JBossWEB is using. That is the CA that have signed the certificate corresponding to the sslKeyAlias stored in the sslKeyStore of the ClusterListener parameters.

12.1.2. ClusterListener configuration part

There is a *wiki* [http://www.jboss.org/community/docs/DOC-9300] describing the SSL parameters of the ClusterListener. See in one example how easy the configuration is:

```
<Listener className="org.jboss.web.cluster.ClusterListener"
```

```
          ssl="true"
          sslKeyStorePass="changeit"
          sslKeyStore="/home/jfclere/CERTS/CA/test.p12"
          sslKeyStoreType="PKCS12"
          sslTrustStore="/home/jfclere/CERTS/CA/ca.p12"
          sslTrustStoreType="PKCS12" sslTrustStorePassword="changeit"
          />
```

The sslKeyStore file contains the private key and the signed certificate of the client certificate JBossWEB uses to connect to httpd. The certificate must be signed by a Cerficate Authority (CA) who certificate is in the conf/server-ca.crt file of the httpd

The sslTrustStore file contains the CA certificate of the CA that signed the certificate contained in conf/server.crt file.

12.1.3. mod-cluster-jboss-beans configuration part

The mod-cluster-jboss-beans.xml in $JBOSS_HOME/server/*profile*/deploy/mod-cluster.sar/META-INF in the ClusterConfig you are using you should have something like:

```
      <property name="ssl">true</property>
      <property name="sslKeyStorePass">changeit</property>
      <property name="sslKeyStore">/home/jfclere/CERTS/test.p12</property>
      <property name="sslKeyStoreType">pkcs12</property>
      <property name="sslTrustStore">/home/jfclere/CERTS/ca.p12</property>
      <property name="sslTrustStoreType">pkcs12</property>
      <property name="sslTrustStorePassword">changeit</property>
```

12.1.4. How the diferent files were created

The files were created using OpenSSL utilities see *OpenSSL* [http://www.openssl.org/] CA.pl (/etc/pki/tls/misc/CA for example) has been used to create the test Certificate authority, the certicate requests and private keys as well as signing the certicate requests.

12.1.4.1. Create the CA

1. Create a work directory and work for there:

```
mkdir -p CERTS/Server
cd CERTS/Server
```

2. Create a new CA:

```
/etc/pki/tls/misc/CA -newca
```

That creates a directory for example ../../CA that contains a cacert.pem file which content have to be added to the conf/server-ca.crt described above.

12.1.4.2. Create the server certificate

1. Create a new request:

   ```
   /etc/pki/tls/misc/CA -newreq
   ```

   That creates 2 files named newreq.pem and newkey.pem. newkey.pem is the file conf/server.key described above.

2. Sign the request:

   ```
    /etc/pki/tls/misc/CA -signreq
   ```

   That creates a file named newcert.pem. newcert.pem is the file conf/server.crt described above. At that point you have created the SSL stuff needed for the VirtualHost in httpd. You should use a browser to test it after importing in the browser the content of the cacert.pem file.

12.1.4.3. Create the client certificate

1. Create a work directory and work for there:

   ```
   mkdir -p CERTS/Client
   cd CERTS/Client
   ```

2. Create request and key for the JBossWEB part.

   ```
   /etc/pki/tls/misc/CA -newreq
   ```

   That creates 2 files: Request is in newreq.pem, private key is in newkey.pem

3. Sign the request.

```
/etc/pki/tls/misc/CA -signreq
```

That creates a file: newcert.pem

4. Don't use a passphrase when creating the client certicate or remove it before exporting:

```
openssl rsa -in newkey.pem -out key.txt.pem
mv key.txt.pem newkey.pem
```

5. Export the client certificate and key into a p12 file.

```
openssl pkcs12 -export -inkey newkey.pem -in newcert.pem -out test.p12
```

That is the sslKeyStore file described above (/home/jfclere/CERTS/CA/test.p12)

12.2. Using SSL between httpd and JBossWEB

Using https allows to encrypt communications betwen httpd and JBossWEB. But due to the ressources it needs that no advised to use it in high load configuration.

(See *Encrypting connection between httpd and TC* [http://www.jboss.org/community/docs/DOC-9701] for detailed instructions).

httpd is configured to be a client for AS/TC so it should provide a certificate AS/TC will accept and have a private key to encrypt the data, it also needs a CA certificate to valid the certificate AS/TC will use for the connection.

```
SSLProxyEngine On
SSLProxyVerify require
SSLProxyCACertificateFile conf/cacert.pem
SSLProxyMachineCertificateFile conf/proxy.pem
```

conf/proxy.pem should contain both key and certificate. The certificate must be trusted by Tomcat via the CA in truststoreFile of <connector/>.

conf/cacert.pem must contain the certificat of the CA that signed the AS/TC certificate. The correspond key and certificate are the pair specificed by keyAlias and truststoreFile of the <connector/>. Of course the <connector/> must be the https one (normally on port 8443).

12.2.1. How the diferent files were created

The files were created using OpenSSL utilities see *OpenSSL* [http://www.openssl.org/] CA.pl (/etc/pki/tls/misc/CA for example) has been used to create the test Certificate authority, the certicate requests and private keys as well as signing the certicate requests.

12.2.1.1. Create the CA

(See *above*)

12.2.1.2. Create the server certificate

(See *above*)

The certificate and key need to be imported into the java keystore using keytool

make sure you don't use a passphare for the key (don't forget to clean the file when done)

1. Convert the key and certificate to p12 file:

   ```
   openssl pkcs12 -export -inkey key.pem -in newcert.pem -out test.p12
   ```

   make sure you use the keystore password as Export passphrase.

2. Import the contents of the p12 file in the keystore:

   ```
   keytool -importkeystore -srckeystore test.p12 -srcstoretype PKCS12
   ```

3. Import the CA certificate in the java trustore: (Fedora13 example).

   ```
   keytool -import -trustcacerts -alias "caname" \
   -file  ../../CA/cacert.pem -keystore /etc/pki/java/cacerts
   ```

4. Edit server.xml to have a <connector/> similar to:

   ```

   <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
              keyAlias="1"
              truststoreFile="/etc/pki/java/cacerts"
              maxThreads="150" scheme="https" secure="true"
              clientAuth="true" sslProtocol="TLS" />
   ```

5. Start TC/AS and use openssl s_client to test the connection:

```
openssl s_client -CAfile /home/jfclere/CA/cacert.pem -cert newcert.pem -key newkey.pem \
-host localhost -port 8443
```

There shouldn't be any error and you should be able to see your CA in the "Acceptable client certificate CA names".

12.3. Forwarding SSL browser informations when using http/https between httpd and JBossWEB

When using http or https beween httpd and JBossWEB you need to use the SSLValve and export the SSL variable as header in the request in httpd. If you are using AJP, mod_proxy_ajp will read the SSL variables and forward them to JBossWEB automaticaly.

(See *Forwarding SSL environment when using http/https proxy* [http://www.jboss.org/community/docs/DOC-11988] for detailed instructions).

The SSL variable used by mod_proxy_ajp are the following:

1. "HTTPS" SSL indicateur.
2. "SSL_CLIENT_CERT" Chain of client certificates.
3. "SSL_CIPHER" Cipher used.
4. "SSL_SESSION_ID" Id of the session.
5. "SSL_CIPHER_USEKEYSIZE" Size of the key used.

Migration from mod_jk

Mod_cluster only support Apache httpd, there are no plan to support IIS or Iplanet.

The migration from mod_jk to mod_cluster is not very complex. Only very few worker properties can't be mapped to mod_cluster parameters.

Here is the table of worker properties and how to transfer them in the ClusterListener parameters.

mod_jk worker property	ClusterListener parameter	Remarks
host	-	It is read from the <Connector/> Address information
port	-	It is read from the <Connector/> Port information
type	-	It is read from the <Connector/> Protocol information
route	-	It is read from the <Engine/> JVMRoute information
domain	domain	That is not supported in this version
redirect	-	The nodes with loadfactor = 0 are standby nodes they will be used no other nodes are available
socket_timeout	nodeTimeout	Default 10 seconds
socket_keepalive	-	KEEP_ALIVE os is always on in mod_cluster
connection_pool_size	-	The max size is calculated to be AP_MPMQ_MAX_THREADS+1 (max)
connection_pool_minsize	smax	The defaut is max
connection_pool_timeout	ttl	Time to live when over smax connections. The defaut is 60 seconds
-	workerTimeout	Max time to wait for a free worker default 1 second
retries	maxAttempts	Max retries before returning an error Default: 3
recovery_options	-	mod_cluster behave like mod_jk with value 7

mod_jk worker property	ClusterListener parameter	Remarks
fail_on_status	-	Not supported
max_packet_size	iobuffersize/receivebuffersize	Not supported in this version. Use ProxyIOBufferSize
max_reply_timeouts	-	Not supported
recovert_time	-	The ClusterListener will tell (via a STATUS message) mod_cluster that the node is up again
activation	-	mod_cluster receives this information via ENABLE/ DISABLE/STOP messages
distance	-	mod_cluster handles this via the loadfactor logic
mount	-	The context "mounted" automaticly via the ENABLE-APP messages. ProxyPass could be used too
secret	-	Not supported
connect_timeout	-	Not supported. Use ProxyTimeout or server TimeOut (Default 300 seconds)
prepost_timeout	ping	Default 10 seconds
reply_timeout	-	Not supported. Use ProxyTimeout or server TimeOut? directive (Default 300 seconds)

Migration from mod_proxy

As mod_cluster is a sophisticated balancer migration from mod_proxy to mod_cluster is strait forward. mod_cluster replaces a reverse proxy with loadbalancing. A reversed proxy is configured like:

```
ProxyRequests Off
<Proxy *>
Order deny,allow
Allow from all
</Proxy>
ProxyPass /foo http://foo.example.com/bar
ProxyPassReverse /foo http://foo.example.com/bar
```

All the general proxy parameters could be used in mod_cluster they work like in mod_proxy, only the balancers and the workers definitions are slightly different.

14.1. Workers

Mod_proxy Parameter	ClusterListener parameter	Remarks
min	-	Not supported in this version
max	-	mod_cluster uses mod_proxy default value
smax	smax	Same as mod_proxy
ttl	ttl	Same as mod_proxy
acquire	workerTimeout	Same as mod_proxy acquire but in seconds
disablereuse	-	mod_cluster will disable the node in case of error and the ClusterListener will for the reuse via the STATUS message
flushPackets	flushPackets	Same as mod_proxy
flushwait	flushwait	Same as mod_proxy
keepalive	-	Always on: OS KEEP_ALIVE is always used. Use connectionTimeout in the <Connector> if needed
lbset	-	Not supported

Mod_proxy Parameter	ClusterListener parameter	Remarks
ping	ping	Same as mod_proxy Default value 10 seconds
lbfactor	-	The load factor is received by mod_cluster from a calculated value in the ClusterListener
redirect	-	Not supported lbfactor sent to 0 makes a standby node
retry	-	ClusterListener will test when the node is back online
route	JVMRoute	In fact JBossWEB via the JVMRoute in the Engine will add it
status	-	mod_cluster has a finer status handling: by context via the ENABLE/STOP/DISABLE/ REMOVE application messages. hot-standby is done by lbfactor = 0 and Error by lbfactor = 1 both values are sent in STATUS message by the ClusterListener
timeout	nodeTimeout	Default wait for ever (http:// httpd.apache.org/docs/2.2/ mod/mod_proxy.html is wrong there)
ttl	ttl	Default 60 seconds

14.2. Balancers

Mod_proxy Parameter	ClusterListener parameter	Remarks
lbmethod	-	There is only one load balancing method in mod_cluster "cluster_byrequests"
maxattempts	maxAttempts	Default 3
nofailover	stickySessionForce	Same as in mod_proxy
stickysession	StickySessionCookie/ StickySessionPath	The 2 parameters in the ClusterListener are combined in one that behaves like in mod_proxy

Mod_proxy Parameter	ClusterListener parameter	Remarks
timeout	workerTimeout	Default 1 seconds

Load Balancing Demo Application

15.1. Overview

The mod_cluster distribution includes a demo application that helps demonstrate how different server-side scenarios affect the routing of client requests by the load balancer. The demo application is located in the mod_cluster distribution's demo directory.

The demo application consists of two components:

1. The first component is a war file that needs to be deployed in JBossWeb/Tomcat/JBoss AS. The war includes a number of servlets.

2. The second component is a GUI application that allows a user to launch a pool of threads that repeatedly make requests to the load balancer. The requests are ultimately routed to the demo war's primary servlet. The application tracks which servers are handling the requests and displays this information in a chart.

   The application can also send separate requests to the demo war's load generation servlets, allowing the user to see how different load conditions affect the balancing of requests.

Note that the demo application does not actually depend on mod_cluster in any way. Its only dependency is on JBossWeb/Tomcat. [1] Consequently, the demo can be used to demonstrate the effect of different server-side scenarios on the routing decisions made by any load balancer, including mod_jk, mod_proxy or the various hardware load balancers.

Note also that this demo application is not intended to be used as a load testing tool; i.e. something that can demonstrate the maximum load a cluster configuration can handle. Using it as such has a good chance of showing you the maximum load the client can generate rather than the maximum load your cluster can handle.

15.2. Basic Usage

To run the demo application:

1. Unpack the mod_cluster distribution on your filesystem. Here we assume it has been unzipped to ~/mod_cluster or C:\mod_cluster.

2. Install mod_cluster into your httpd server as described at *Installation of the httpd part*

3. Install mod_cluster into your JBossAS/JBossWeb/Tomcat servers as described at *Installation on the Java side*

4. Start httpd and your JBossAS/JBossWeb/Tomcat servers

[1]The demo's "Datasource Use" load generation scenario requires the use of JBoss Application Server.

5. Deploy the load-demo.war found in the distribution's demo/server folder to your JBossAS/JBossWeb/Tomcat servers.

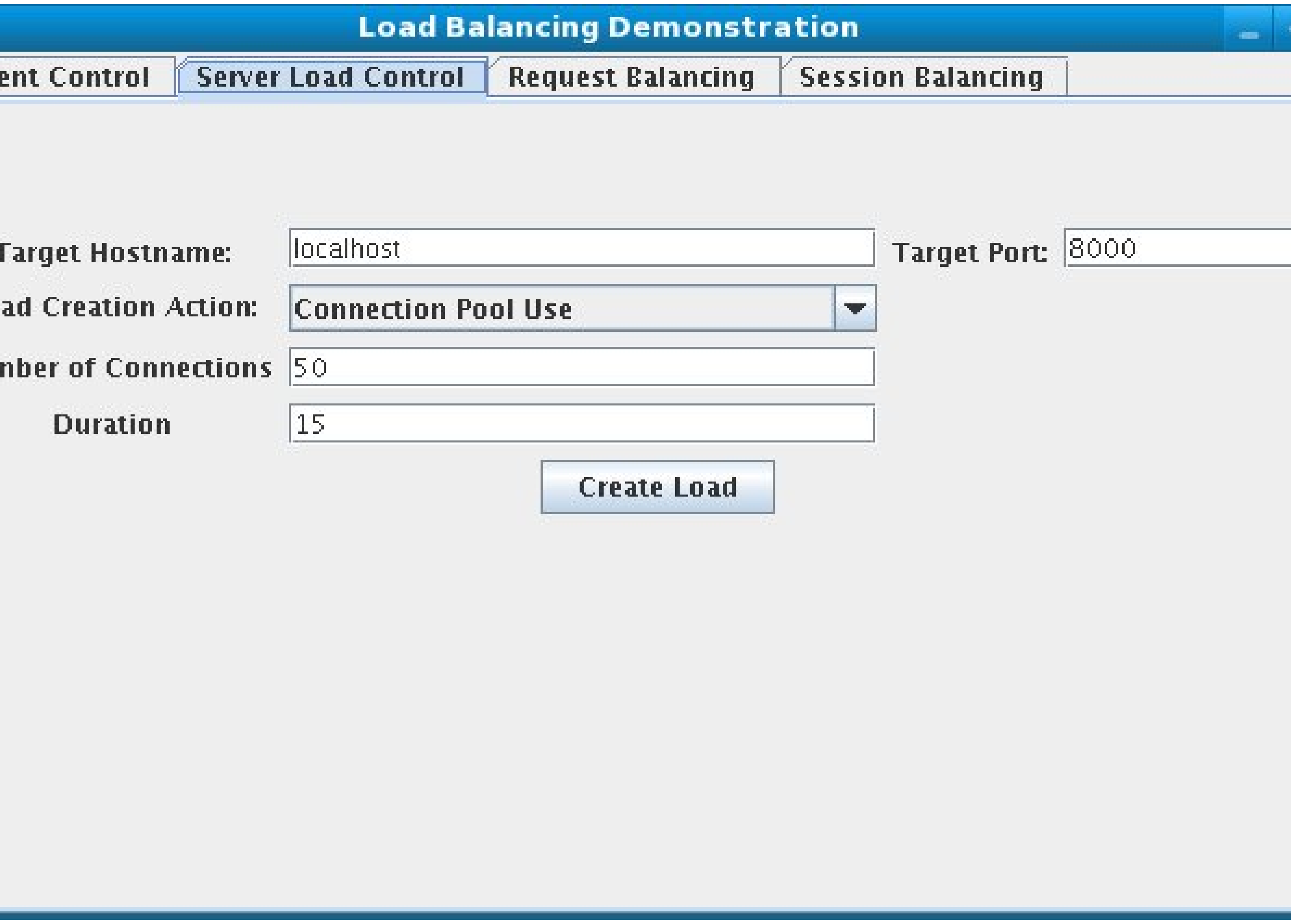
6. Start the demo application:

   a. On *nix:

```
cd ~/mod_cluster/demo/client
./run-demo.sh
```

   b. On Windows:

```
C:\>cd mod_cluster\demo\client
C:\mod_cluster\demo\client>run-demo
```



Target Hostname: localhost
Target Port: 8000
ad Creation Action: Connection Pool Use
nber of Connections 50
Duration 15

7. Configure the hostname and address of the httpd server, the number of client threads, etc and click the "Start" button. See *Client Driver Configuration Options* for details on the configuration options.

8. Switch to the "Request Balancing" tab to see how many requests are going to each of your JBossAS/JBossWeb/Tomcat servers.

9. Switch to the "Session Balancing" tab to see how many active sessions [2] are being hosted by each of your JBossAS/JBossWeb/Tomcat servers.

10. Stop some of your JBossAS/JBossWeb/Tomcat servers and/or undeploy the load-demo.war from some of the servers and see the effect this has on load balancing.

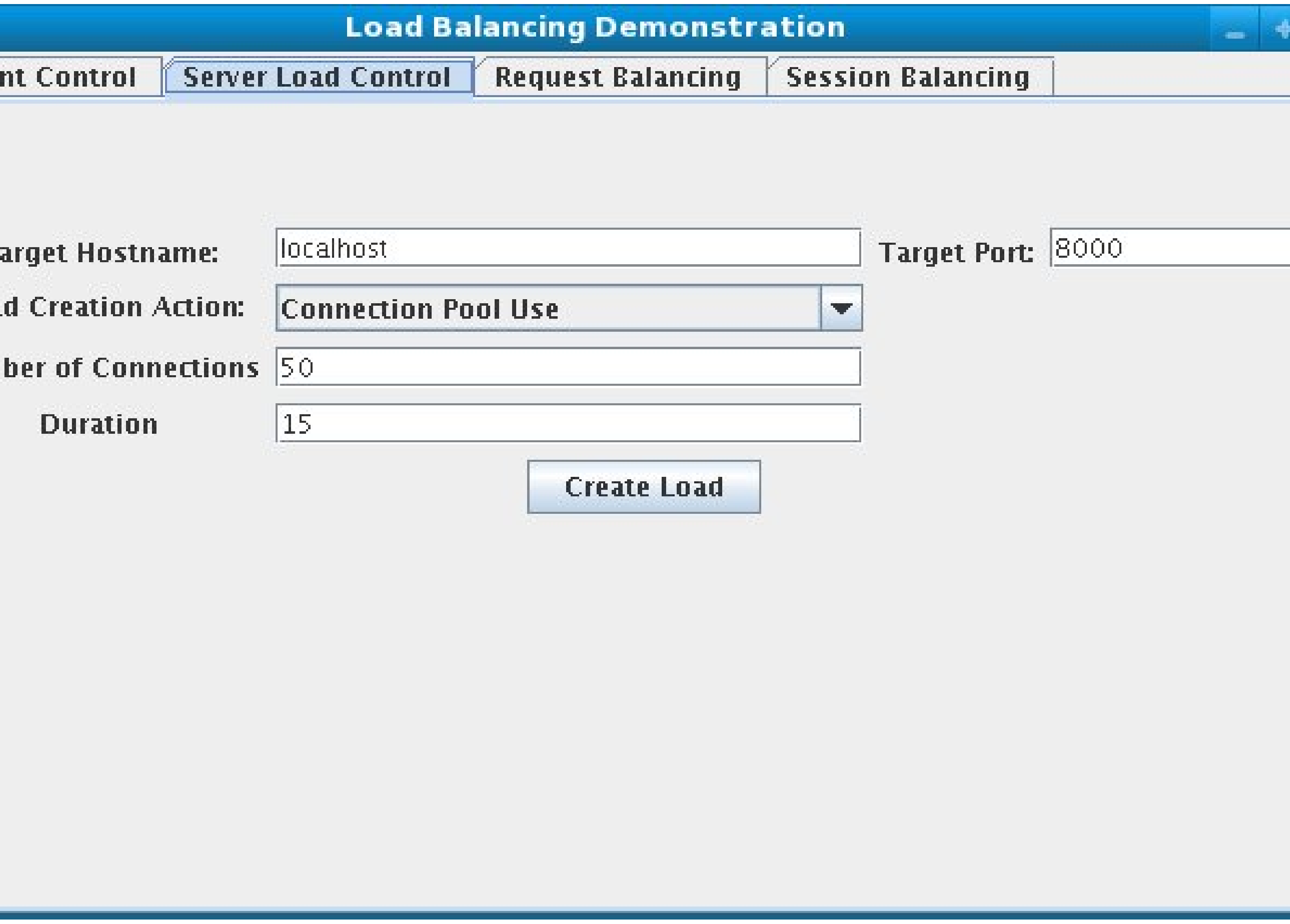
11. Restart some of your JBossAS/JBossWeb/Tomcat servers and/or re-deploy the load-demo.war to some of the servers and see the effect this has on load balancing.

12. Experiment with adding artificial load to one or more servers to see what effect that has on load balancing. See *Load Generation Scenarios* for details.

Most of the various panels in application interface also present information on the current status on any client threads. "Total Clients" is the number of client threads created since the last time the "Start" button was pushed. "Live Clients" is the number of threads currently running. "Failed Clients" is the number of clients that terminated abnormally; i.e. made a request that resulted in something other than an HTTP 200 response.

15.3. Client Driver Configuration Options

The configuration of the client is driver is done via the application's "Clent Control" tab.



arget Hostname: localhost Target Port: 8000

d Creation Action: Connection Pool Use

ber of Connections 50

Duration 15

The panel includes the following options:

1. Proxy Hostname: Hostname of the load balancer or the IP address on which it is listening for requests [3]
2. Proxy Port: Port on which the load balancer is listening for requests [4]
3. Context Path: Portion of the request URL that specifies the request is for the load-demo.war
4. Session Life: Number of seconds a client thread should use a session before invalidating or abandoning it. Generally it is good to keep this to a small value; otherwise the use of session stickiness will prevent changes in server load from affecting the load balancer's routing decisions. With sticky sessions enabled (strongly recommended), it is the creation of a new session that allows the load balancer to try to balance load.
5. Invalidate: Controls what the client thread should do when it stops using a session because Session Life has passed. If checked, the driver will send a request that results in the session

being invalidated. If unchecked, the session will just be abandoned, and will continue to exist on the server until Session Timeout seconds have passed. In the future this will likely be changed to a percentage input, so X% can be invalidated, the rest abandoned.

6. Session Timeout: Number of seconds a session can remain unused before the server is free to expire it. Unchecking Invalidate and setting a high value relative to Session Life allows a significant number of unused sessions to accumulate on the server.

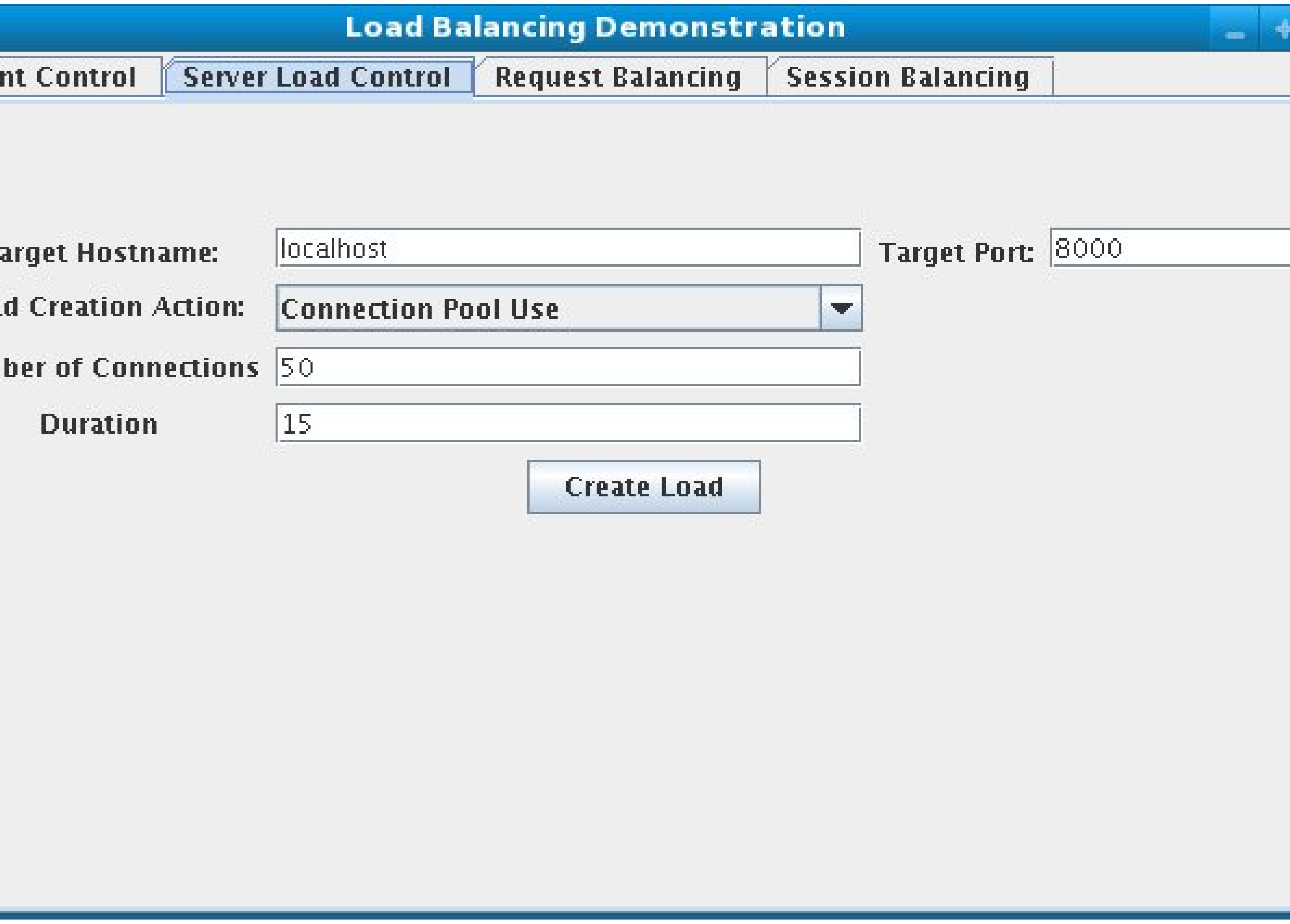
7. Num Threads: Number of client threads to launch. Each thread repeatedly makes requests until the "Stop" button is pushed or a request receives a response other than HTTP 200.

8. Sleep Time: Number of ms the client threads should sleep between requests.

9. Startup Time: Number of seconds over which the application should stagger the start of the client threads. Staggering the start advised as it avoids the unnatural situation where for the life of the demonstation all sessions start at about the same time and then are invalidated or abandoned at the same time. Staggering the start allows the load balancer to continually see new sessions and decide how to route them.

15.4. Load Generation Scenarios

You can use the application's GUI to instruct individual servers to artificially generate various types of load, and then track how that load affects request and session balancing. Load generation is controlled via the application's "Server Load Control" tab.



arget Hostname: localhost
Target Port: 8000
d Creation Action: Connection Pool Use
ber of Connections 50
Duration 15

The panel includes the following options:

- Target Hostname and Target Port: The hostname or IP address of the server on which you want load generated. There are two strategies for setting these:

  1. You can use the hostname and port of the load balancer, in which case the load balancer will pick a backend server and route the request to it. Note the client application does not maintain a session cookie for these requests, so if you invoke another server load generation request, you shouldn't expect the same server to handle it.

  2. If the JBoss AS/JBossWeb/Tomcat servers are running the HttpConnector as well as the AJP connector, you can specify the address and port on which a particular server's HttpConnector is listening. The standard port is 8080.

- Load Creation Action: Specifies the type of load the target server should generate. See below for details on the available load types.

- Params: Zero or more parameters to pass to the specified load creation servlet. For example, in the screenshot above, Number of Connections and Duration. How many parameters are displayed, their name and their meaning depend on the selected Load Creation Action. The label for each parameter includes a tooltip that explains its use.

  The available Load Creation Actions are as follows:

  Active Sessions
  : Generates server load by causing session creation on the target server.

  Datasource Use
  : Generates server load by taking connections from the java:DefaultDS datasource for a period

  Connection Pool Use
  : Generates server load by tieing up threads in the webserver connections pool for a period

  Heap Memory Pool Use
  : Generates server load by filling 50% of free heap memory for a period

  CPU Use
  : Generates server CPU load by initiating a tight loop in a thread

  Server Receive Traffic
  : Generates server traffic receipt load by POSTing a large byte array to the server once per second for a period

  Server Send Traffic
  : Generates server traffic send load by making a request once per second to which the server responds with a large byte array

  Request Count
  : Generates server load by making numerous requests, increasing the request count on the target server

Change Log

16.1. 1.1.3.Final (12 August 2011)

🔨	request hang with a node is stopped in EC2 (MODCLUSTER-217) (jfclere)
🔨	Extra ENABLE-APP/REMOVE-APP event after failover (MODCLUSTER-220) (jfclere)
🔨	mod_cluster does not work with Tomcat 7 due to API Change in Connector (MODCLUSTER-240) (jfclere)
🔨	kill -HUP httpd process increase the number of open locks (MODCLUSTER-241) (jfclere)

16.2. 1.1.2.Final (21 April 2011)

🔨	mod_cluster failover does not work for a / webappcontext when the / root context exists (MODCLUSTER-188) (jfclere)
🔨	UseAlias broken (MODCLUSTER-212) (jfclere)
🔨	mod_rewrite PT doesn't work (MODCLUSTER-213) (jfclere)
🔨	memory usage growning in httpd (MODCLUSTER-214) (jfclere)
🔨	Second attempt to connect from Jboss to apache module sends incomplete Host Header (MODCLUSTER-216) (jfclere)
🔨	when using tomcat manager webapp to stop/ start an application the start is ignored by mod_cluster (MODCLUSTER-224) (jfclere)
🔨	Unable to override load properties when running in Tomcat 6 (MODCLUSTER-232) (jfclere)
🔨	Documentation contains wrong default loadMetric class name (MODCLUSTER-233) (jfclere)
🔧	

	mod_cluster should issue a warning when Maxcontext is reached and no more context will be taken (MODCLUSTER-223) (jfclere)
☞	add a note explaining where Maxcontext must be put in httpd.conf and issue error if in wrong location (MODCLUSTER-225) (jfclere)

16.3. 1.1.1.Final (31 January 2011)

🔨	NPE when overriding default load metric in ModClusterListener (MODCLUSTER-183) (pferraro)
🔨	mod_cluster failover does not work for a / webappcontext when the / root context exists (MODCLUSTER-188) (jfclere)
🔨	mod_cluster issues an ENABLE-APP too early in the webapp lifecycle (MODCLUSTER-190) (jfclere)
🔨	mod_cluster 1.1.0 docs step 11.1.4.3 is wrong (MODCLUSTER-193) (jfclere)
🔨	Incorrect routing of requests when one context root is the prefix of another (MODCLUSTER-196) (jfclere)
🔨	Can only rewrite from the root context in httpd if there is a root context deployed in JBoss (MODCLUSTER-198) (jfclere)
🔨	the STATUS MCMP message is send before the connector is started (MODCLUSTER-202) (jfclere)
🔨	The windoze bundle don't have the default configuration (MODCLUSTER-205) (jfclere)
🔨	httpd cores after graceful restart after the mod_cluster configuration is added (MODCLUSTER-206) (jfclere)
☞	Make logging on Tomcat use JDK14LoggerPlugin by default (MODCLUSTER-185) (pferraro)
☞	update mod_cluster to use HTTP/1.1 (MODCLUSTER-201) (jfclere)
☞	

	Add support for system properties used in 1.0.x (MODCLUSTER-207) (pferraro)

16.4. 1.1.0.Final (13 August 2010)

🔨	Demo servlets throw InstanceNotFoundException against EAP5 (MODCLUSTER-170) (pferraro)
🔨	mod_advertise: Invalid ServerAdvertise Address too often (MODCLUSTER-172) (jfclere)
🔨	Fix class name of MBeanAttributeRatioLoadMetric in MC config (MODCLUSTER-174) (pferraro)
🔨	Wrong configuration could cause an httpd core (MODCLUSTER-175) (jfclere)
🔨	Advertise not configured error message in log is actually a warning (MODCLUSTER-176) (jfclere)
🧩	Better no servers connected message (MODCLUSTER-165) (jfclere)
🧩	How does the UI deal with 20 or more servers (MODCLUSTER-165) (jfclere)
🧩	mod_cluster should use hostname provided in address instead a IP address (MODCLUSTER-168) (pferraro)
🧩	Read only view of mod_cluster-manager (MODCLUSTER-181) (jfclere)
📋	Demo client throws Bus Error when run with JDK 1.6 on OSX (MODCLUSTER-169) (jfclere)
📋	Use versioned docs (MODCLUSTER-141) (jfclere, pferraro)
🧩	Deprecate use of term "domain" (MODCLUSTER-177) (jfclere, pferraro)

16.5. 1.1.0.CR3 (15 June 2010)

🔨	Rpc failure can lead to failure to deploy a webapp (MODCLUSTER-140) (pferraro)
🔨	

	Quotes in jsessionId causing sticky sessions to fail (MODCLUSTER-146) (jfclere)
🔨	ManagerBalancerName doesn't work (MODCLUSTER-153) (jfclere)
🔨	Parsing of IPv6 loopback address fails (MODCLUSTER-156) (pferraro)
🔨	SystemMemoryUsageLoadMetric returns wrong load metric (MODCLUSTER-157) (pferraro)
🔨	Clean shutdown logic can still inadvertently kill requests for non-distributed contexts. (MODCLUSTER-159) (pferraro)
🔨	NoClassDefFoundError running demo app against AS6 (MODCLUSTER-161) (pferraro)
➕	Allow override of default clean shutdown behavior (MODCLUSTER-139) (pferraro)
➕	Avoid unnecessary open sockets for non-master nodes (MODCLUSTER-158) (pferraro)

16.6. 1.1.0.CR2 (11 May 2010)

➕	Add the lifecycle listener dynamically (MODCLUSTER-20) (pferraro)
➕	Use UUID for auto-generated jvmRoute (MODCLUSTER-142) (pferraro)
➕	improve packaging so that the bundle can run in ~ too (MODCLUSTER-150) (jfclere)
🔨	jboss.mod_cluster.proxyList: invalid hosts cause mod-cluster startup to be delayed (MODCLUSTER-155) (pferraro)
🔨	mod_cluster 1.1.0.CR1 doesn't work with Tomcat (MODCLUSTER-143) (jfclere/ pferraro)
➕	Allow configuration of stopContextTimeout units (MODCLUSTER-138) (pferraro)
➕	Add a solaris10 64 bits sparc in the bundles (MODCLUSTER-137) (jfclere)
🔨	INFO and mod_cluster_manager/ displays milliseconds and DUMP second (MODCLUSTER-128) (jfclere)

➕	Make mod_cluster manager tolerant to F5 page refresh when disabled context (MODCLUSTER-124) (jfclere)

16.7. 1.1.0.CR1 (22 March 2010)

➕	Update httpd to 2.2.25. (MODCLUSTER-134) (jfclere)
🔨	Apache with mod_cluster refuses to start at first, but after 7 retries it starts up (MODCLUSTER-120) (jfclere)
➕	Add getLoad() to load metric mbean interface (MODCLUSTER-130) (pferraro)
➕	Disable "cnone" request parameter to ease remote invocation on mod_cluster-manager (MODCLUSTER-127) (jfclere)
🔨	Microcontainer does not always choose the right constructor when creating ModClusterService (MODCLUSTER-116) (pferraro)
🔨	Microcontainer does not choose the right constructor when creating RequestCountLoadMetric (MODCLUSTER-126) (pferraro)
🔨	Mod_cluster does support more that 3 Alias in <Host/> (MODCLUSTER-121) (jfclere)
➕	Allow toggling of context auto-enable during mod_cluster startup. (MODCLUSTER-125) (pferraro)
🔨	STATUS should retry the worker even if there was an error before. (MODCLUSTER-133) (jfclere)
🔨	Split ModClusterServiceMBean.ping(String) into 3 methods (MODCLUSTER-110) (pferraro/jfclere)
🔨	Use clean shutdown by default, leveraging STOP-APP-RSP for <distributable/> contexts and session draining for non-distributable contexts. mod_cluster shutdown now triggered earlier via Connector JMX notification. (MODCLUSTER-131) (pferraro/jfclere)

➕	move the web site to magnolia (MODCLUSTER-114) (.org team)
🔨	ping and nodeTimeout interact. (MODCLUSTER-132) (jfclere)
➕	update mod_jk to 1.2.30 (MODCLUSTER-138) (jfclere)
🔨	query string is truncated to (MODCLUSTER-118) (jfclere)
🔨	AdvertiseBindAddress does not default to the 23364 port (MODCLUSTER-119) (jfclere)
➕	Skip load balance factor calculation if there are no proxies to receive status message (MODCLUSTER-103) (pferraro)
🔨	Disabling contexts does not work (MODCLUSTER-123) (jfclere)
🔨	advertise doesn't use new AdvertiseSecurityKey on graceful restarts. (MODCLUSTER-129) (jfclere)
🔨	Load-demo.war specifies obsolete servlet in web.xml (MODCLUSTER-113) (pferraro)

16.8. 1.1.0.Beta1 (30 October 2009)

🔨	Interaction with mod_rewrite looks weird for end-users. (MODCLUSTER-86) (jfclere)
➕	admin-console should be in the excludedContexts. (MODCLUSTER-87) (pferraro)
🔨	ClassCastException upon redeploy after mod-cluster-jboss-beans.xml modification. (MODCLUSTER-88) (pferraro)
🔨	Alias from webapps/jboss-web.xml are not handled correctly in mod_cluster. (MODCLUSTER-89) (jfclere)
➕	Display version. (MODCLUSTER-90) (jfclere)
🔨	Connector bind address of 0.0.0.0 propagated to proxy. (MODCLUSTER-91) (pferraro)
➕	Display status of the worker. (MODCLUSTER-92) (jfclere)

➕	Update httpd to lastest version. (MODCLUSTER-93) (jfclere)
🔨	getProxyInfo failed when there are too many nodes. (MODCLUSTER-94) (jfclere)
🔨	mod_cluster-manager display corrupted with jboss starting. (MODCLUSTER-95) (jfclere)
🔨	DISABLE application active as STOPPED. (MODCLUSTER-96) (jfclere)
➕	Httpd should remove workers it can't ping. (MODCLUSTER-97) (jfclere)
🔨	Linux mod_cluster_manager display zero instead values. (MODCLUSTER-98) (jfclere)
🔨	mod_cluster_manager doesn't seem to ENABLE/DISABLE the right context. (MODCLUSTER-99) (jfclere)
🔨	load balancing logic doesn't allow manual demo of load-balancing. (MODCLUSTER-100) (jfclere)
🔨	404 errors when load is increasing. (MODCLUSTER-102) (jfclere)
🔨	Advertise security key verification does not work. (MODCLUSTER-104) (jfclere)
🤝	Allow advertise listener to listen on a specific network interface. (MODCLUSTER-106) (pferraro)
🤝	Allow thread factory injection for advertise listener. (MODCLUSTER-108) (pferraro)
➕	Create SPI and isolate tomcat/jbossweb usage into service provider implementation. (MODCLUSTER-111) (pferraro)

Frequently Asked questions

17.1. What is Advertise

Advertise allows autodiscovery of httpd proxies by the cluster nodes. It is done by sending multicast messages from httpd to the cluster. The httpd specialized module: mod_advertise sends UDP message on a multicast group, both mod_advertise and the cluster listener joined the multicast group and the cluster receives the messages.

Example for mod_advertise message:

```
HTTP/1.0 200 OK
Date: Wed, 08 Apr 2009 12:26:32 GMT
Sequence: 16
Digest: f2d5f806a53effa6c67973d2ddcdd233
Server: 1b60092e-76f3-49fd-9f99-a51c69c89e2d
X-Manager-Address: 127.0.0.1:6666
X-Manager-Url: /bla
X-Manager-Protocol: http
X-Manager-Host: 10.33.144.3
```

The X-Manager-Address header value is used by the cluster logic to send information about the cluster to the proxy. It is the IP and port of the VirtualHost where mod_advertise is configured or URL parameter of the ServerAdvertise directive.

See *Proxy Discovery* in Configuration Properties and *mod_advertise* in Apache httpd configuration.

17.2. What to do if I don't want to use Advertise (multicast):

In the VirtualHost receiving the MCPM of httpd.conf don't use any Advertise directive or use:

```
ServerAdvertise Off
```

In mod_cluster-jboss-beans.xml add the addresses and ports of the VirtualHost to the proxyList property and set advertise to false, for example:

```
<property name="proxyList">10.33.144.3:6666,10.33.144.1:6666</property>
<property name="advertise">false</property>
```

In server.xml (with JBossweb/Tomcat)

```
<Listener className="org.jboss.modcluster.catalina.ModClusterListener" advertise="true"/>
```

17.3. I am using Tomcat 7 / 6 what should I do:

See at the end of *java configuration*. You can't use the mod_cluster clustered mode with Tomcat so you get a loadbalancing logic similar to mod_jk but with a dynamic configuration.

17.4. It is not working what should I do:

Most likely you have a configuration problem. Check the log of the cluster nodes and error_log of httpd.

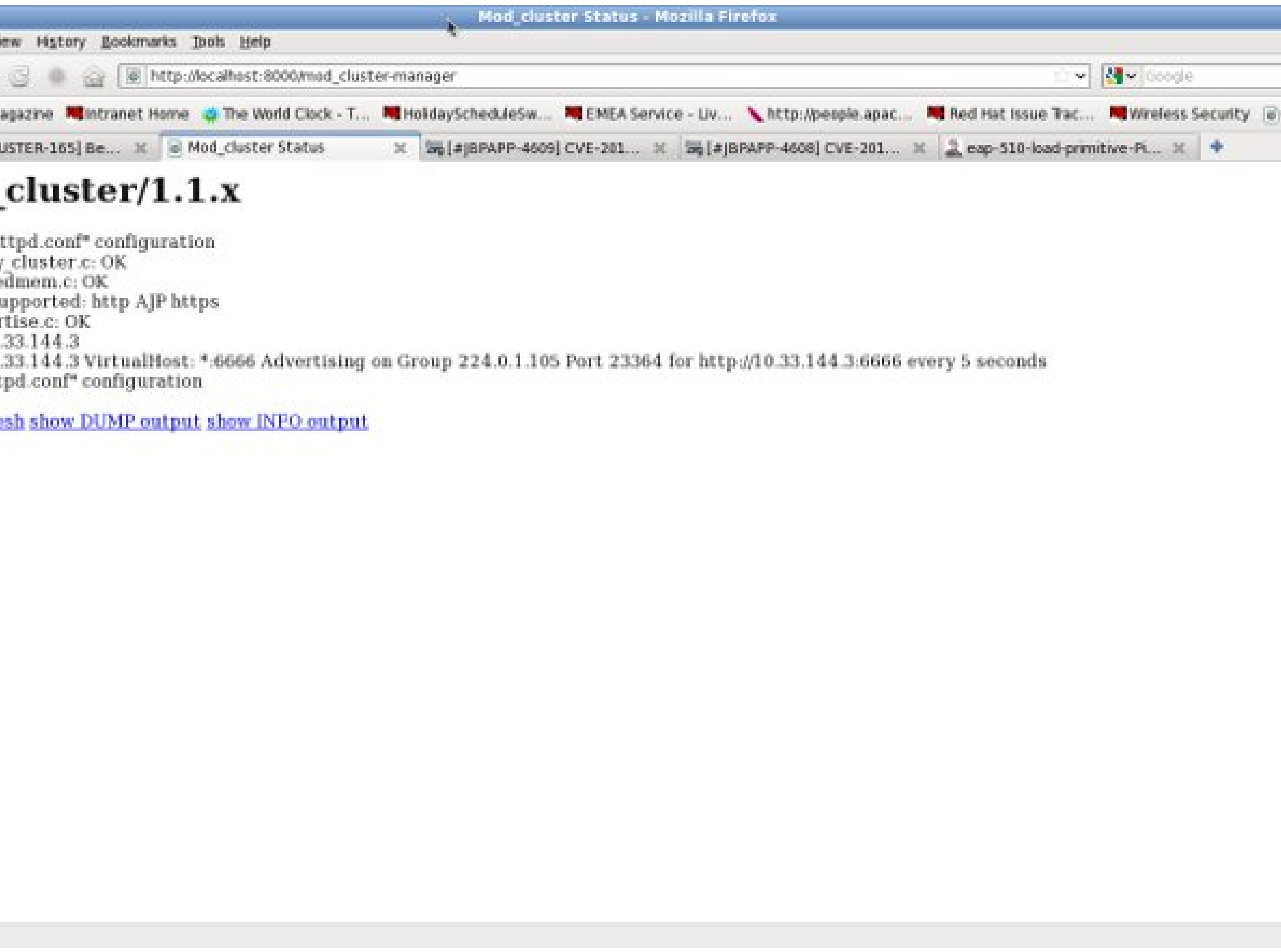
17.4.1. No error

That happens when Advertise is not working: The nodes don't get the *advertise messages* from httpd.

1. Check the modules are loaded and Advertise is started. In httpd.conf activate extended information display, add:

   ```
   AllowDisplay On
   ```

   When accessing to the *mod_cluster_manager* you should get something like:


Mod_cluster Status - Mozilla Firefox
ew History Bookmarks Tools Help
http://localhost:8000/mod_cluster-manager
agazine Intranet Home The World Clock - T... HolidayScheduleSw... EMEA Service - Liv... http://people.apac... Red Hat Issue Trac... Wireless Security
USTER-165] Be... Mod_cluster Status [#JBPAPP-4609] CVE-201... [#JBPAPP-4608] CVE-201... eap-510-load-primitive-R...

```
_cluster/1.1.x
ttpd.conf" configuration
y_cluster.c: OK
edmem.c: OK
upported: http AJP https
rtise.c: OK
.33.144.3
.33.144.3 VirtualHost: *:6666 Advertising on Group 224.0.1.105 Port 23364 for http://10.33.144.3:6666 every 5 seconds
tpd.conf" configuration

esh show DUMP output show INFO output
```

If not, go to the *Minimal Example* and add the missing directive(s).

2. Check that Advertise message are received on the cluster node. A *small Java* [http://anonsvn.jboss.org/repos/mod_cluster/trunk/test/java/Advertize.java] utility could be used to check Advertise. It is in the mod_cluster repository and can be compiled using javac. A compiled version can be found under in /opt/jboss/httpd/tools in the bundles. Run it using java Advertise multicastaddress port. The output should be something like:

```
[jfclere@jfcpc java]$ java Advertize 224.0.1.105 23364
ready waiting...
received: HTTP/1.0 200 OK
Date: Mon, 28 Jun 2010 07:30:31 GMT
Sequence: 1
Digest: df8a4321fa99e5098174634f2fe2f87c
```

```
Server: 1403c3be-837a-4e76-85b1-9dfe5ddb4378
X-Manager-Address: test.example.com:6666
X-Manager-Url: /1403c3be-837a-4e76-85b1-9dfe5ddb4378
X-Manager-Protocol: http
X-Manager-Host: test.example.com
```

3. No Advertise messages

   Check firewall (don't forget the boxes firewall). Advertise uses UDP port 23364 and multicast addresse 224.0.1.105

4. Can't get Advertise messages

   Use ProxyList property. In case Advertise can't work you put the address and port of the VirtualHost used in httpd to receive the MCMP. In server/profile/deploy/mod_cluster.sar/META-INF/mod_cluster-jboss-beans.xml

```
<property name="proxyList">test.example.com:6666</property>
```

   or in server.xml:

```
<Listener                    className="org.jboss.modcluster.catalina.ModClusterListener"
 proxyList="test.example.com:6666"/>
```

17.4.2. Error in server.log or catalina.out

1. "IO error sending command ":

   Check firewall and error_log if there is nothing error_log then it is a firewall problem. If you have something like:

```
18:36:14,533 INFO  [DefaultMCMPHandler] IO error sending command INFO to proxy jfcpc/
10.33.144.3:8888
```

   You can use telnet hostname/address port to check by hands that it is OK for example:

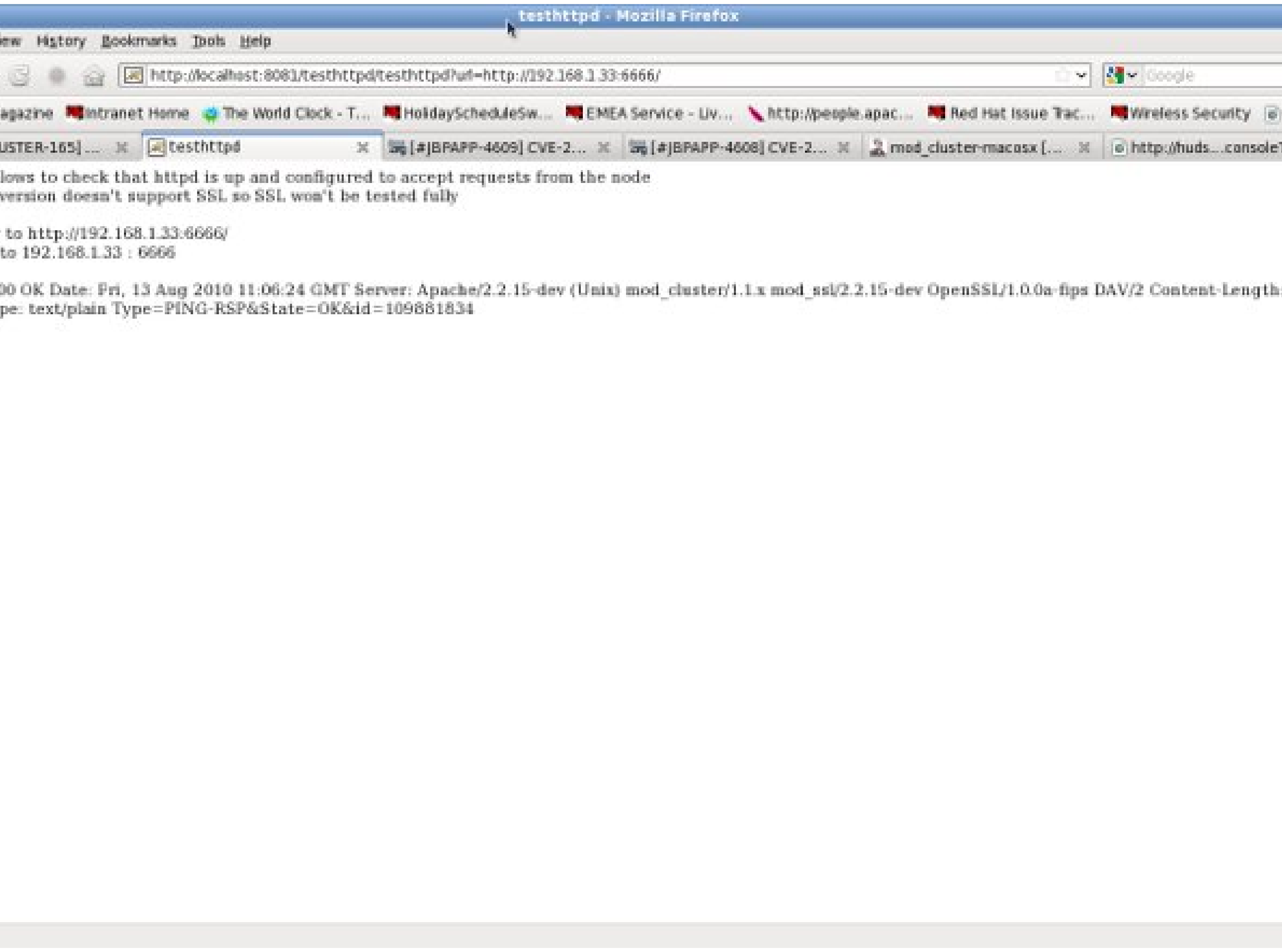
```
[jfclere@jfcpc docs]$ telnet 10.33.144.3 8888
Trying 10.33.144.3...
Connected to jfcpc.
Escape character is '^]'.
GET /
```

```
<html><body><h1>It works!</h1></body></html>Connection closed by foreign host.
```

Check that the address and port are the expected ones you may use ServerAdvertise On http://hostname:port, like:

```
ServerAdvertise On http://localhost:6666
```

Or use the servlet testhttpd in the testhttpd.war of the bundle (in /opt/jboss/httpd/tools). Install/deploy it in in the node and start the AS/Tomcat on the node, than access to the node directly and call /testhttpd/testhttpd



```
lows to check that httpd is up and configured to accept requests from the node
version doesn't support SSL so SSL won't be tested fully

 to http://192.168.1.33:6666/
to 192.168.1.33 : 6666

00 OK Date: Fri, 13 Aug 2010 11:06:24 GMT Server: Apache/2.2.15-dev (Unix) mod_cluster/1.1.x mod_ssl/2.2.15-dev OpenSSL/1.0.0a-fips DAV/2 Content-Length
pe: text/plain Type=PING-RSP&State=OK&id=109881834
```

If you don't get a similar page the output should help to find that is wrong.

17.4.3. Error in error_log

1. "client denied by server configuration":

   The directory in the VirtualHost is not allowed for the client. If you have something like:

```
Mon Jun 28 18:08:47 2010] [error] [client 10.33.144.3] client denied by server configuration: /
```

   You need to have something like:

```
<Directory />
  Order deny,allow
  Deny from all
  Allow from 10.33.144.3
</Directory>
```