HITACHI
Inspire the Next

Pentaho Server in High Availability (HA) End to End

HITACHI
Inspire the Next

Change log (if you want to use it):

Date	Version	Author	Changes

Contents

This page intentionally left blank.

Overview

This document covers some best practices on setting up your Pentaho servers with a clustered High Availability (HA) solution. Due to the clusters, the solutions presented are for non-ETL use.

Some of the topics discussed here include clustering the Pentaho server nodes, configuring a load balancer for High Availability, and Tomcat and Apache configuration for Windows and Linux.

Software	Version(s)
Pentaho	7.x, 8.0

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

Before beginning, use the following information to prepare for the procedures described in the main section of the document.

In this guide, we will be using Apache HTTPD. All configurations will be shown in an Apache web server context.

1. This guide assumes that a load balancer is already installed and running.
2. You will need to make sure that each Pentaho server in your cluster is already configured to use the repository database of your choice. Make sure each server is pointing to the same database instance.
3. Initialize your database using the appropriate steps for your system. Pentaho documentation has instructions for PostgreSQL, MySQL, MS SQL Server, and Oracle databases.
4. After you have initialized and configured your repository, clean up temporary files by locating the `../pentaho-server/tomcat` directory and removing all files and folders from the `temp` and `work` folders.

You now have a configured repository and are ready to move to the next step for clustering.

Use Case: Managing Increased Server Traffic

Janice needs to address and manage increasing data processing and concurrent user connections within her Pentaho server. We recommend setting up a cluster of Pentaho servers with a clustered High Availability solution, which places a load balancer in front of the Pentaho cluster and directs traffic accordingly. Before doing so, Janice will need to make sure that her servers are configured to use the repository database, and make sure that they are prepared to reach the same database instance.

Introduction

Most installations of Pentaho are single-server installations. This solution works well in small- and medium-sized organizations. In large scale deployments, however, a clustered High Availability (HA) solution is needed to address the increase in data processing and concurrent user connections.

HA solutions for Pentaho servers follow the typical load balancing models, where a load balancer (such as Apache HTTPD) sits in front of a cluster of Pentaho servers and forwards traffic using either a round-robin fashion or other methods, such as worker server quotas.

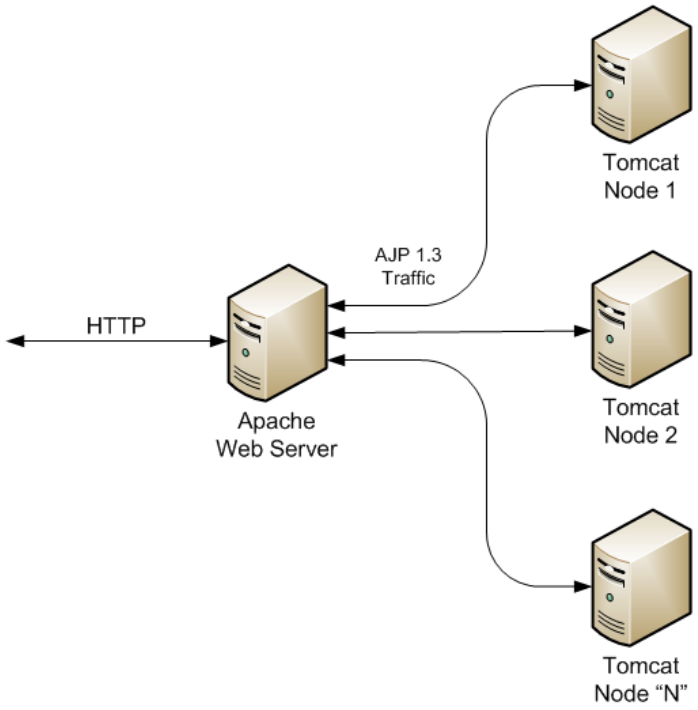

Figure 1: Pentaho Server in HA Diagram

This model allows not only for distributed processing, but also for failover. This way, if one Pentaho server goes down, service is not interrupted, but is instead taken over by a live server.

A single point of failure with this model would be with the load balancer, so further consideration to cluster the load balancer would need to be taken. In this best practice guide, we will only be covering the configuration of a single load balancer.

Clustering the Pentaho Server Nodes

The first step in developing a Pentaho server with HA solution would be to configure the cluster nodes to use the same backend repository.

Figure 2: Cluster Node Configuration Workflow

You can find details on these topics in the following sections:

- Configure Jackrabbit Journal
- Configure Quartz
- Start and Test the Cluster

Configure Jackrabbit Journal

Pentaho uses Apache Jackrabbit as its content repository. Documentation on how to configure a cluster for each database repository type is available in the Cluster the Application Server documentation.

Configure the Jackrabbit Journal for your cluster using these steps:

1. Make sure each node has a unique ID.
2. Locate the `repository.xml` file in the `…/pentaho-server/pentaho-solutions/system/jackrabbit` directory and open it with a text editor.
3. Scroll to the bottom of `repository.xml` and replace the section that begins `<!—Run with a cluster journal -->` with the correct code for your database repository.
4. Save and close `repository.xml`.

Configure Quartz

Pentaho uses Quartz for scheduling. Documentation on how to configure Quartz to work with your cluster is available in the Cluster the Application Server documentation.

We recommend using the Pentaho User Console (PUC) for scheduling.

Configure Quartz using these steps:

1. Locate the `quartz.properties` file in the ...`pentaho-server/pentaho-solutions/system/quartz` directory and open it with any text editor.
2. Find the `org.quartz.scheduler.instanceID = INSTANCE_ID` line and change `INSTANCE_ID` to `AUTO`:

```
org.quartz.scheduler.instanceID = AUTO
```

3. Find the `#_replace_jobstore_properties` section and change the default value of `org.quartz.jobStore.isClustered` to `true`:

```
#_replace_jobstore_properties


org.quartz.jobStore.misfireThreshold = 60000
org.quartz.jobStore.driverDelegateClass =
org.quartz.impl.jdbcjobstore.PostgreSQLDelegate
org.quartz.jobStore.useProperties = false
org.quartz.jobStore.dataSource = myDS
org.quartz.jobStore.tablePrefix = QRTZ5_
org.quartz.jobStore.isClustered = true
```

4. Add this line just after the `org.quartz.jobStore.isClustered = true` line:

```
org.quartz.jobStore.clusterCheckinInterval = 20000
```

More information is available at the Quartz Configuration Reference site.

Start and Test the Cluster

Follow these instructions to start the cluster and verify that it is working properly:

1. Start the solution database.
2. Start the application server on each node.
3. Make sure that the load balancer can ping each node.
4. Access the login screen for each server in your cluster.
5. Browse directly to each node instead of going through the load balancer.

Configuring a Load Balancer for Pentaho Server High Availability

Now that each server in the Pentaho cluster can be accessed individually, it is time to configure the load balancer to manage traffic to each node.

The first section will cover configuring Apache as a load balancer with the Pentaho Server cluster nodes. We provide instructions for configuring Apache for both Windows and Linux, and additional considerations to keep in mind while you are working through these procedures.

Tomcat Configuration

To configure Tomcat:

1. Shut down the Tomcat server on each node in the cluster.
2. Open the `tomcat/conf/server.xml` file in a text editor.
3. Locate the following line `<Engine name="Catalina" defaultHost="localhost">`.
4. The `jvmRoute` value will need to be unique for each node. This value will map to the `BalancerMember` setup in the following section when configuring Apache. Add the `jvmRoute` attribute like this:

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="server1">
```

5. To close connections so they do not become orphaned and cause errors, locate the following line:

```
<Connector URIEncoding="UTF-8" port="8009" protocol="AJP/1.3"
redirectPort="8443" />
```

   Change it to:

```
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443"
           connectionTimeout="10000" keepAliveTimeout="10000" />
```

6. Edit the `pentaho-solutions/system/server.properties` and change the `fully-qualified-server-url param-value` to the load balancer's URL.
7. Start the Tomcat service back up after configuring this file.
8. Repeat this process throughout all nodes within the cluster.

`connectionTimeout` is the number of milliseconds this connector will wait, after accepting a connection, for the request URI line to be presented. The default value for AJP protocol connectors is `-1` (infinite).

`keepAliveTimeout` is the number of milliseconds this connector will wait for another AJP request before closing the connection. The default value is to use the value that has been set for the `connectionTimeout` attribute.

Apache Configuration (Windows Environment)

There are different ways to configure Apache in a Windows environment. One example is presented here:

1. Open the `C:\Program Files (x86)\Apache24\conf\httpd.conf` file in a text editor.
2. The following modules will need to be uncommented if they aren't already:

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

3. At the bottom of the file, add the following code, making sure to reflect the `BalancerMember` hostname with your Pentaho cluster nodes:

```
ProxyPass /pentaho balancer://reportingcluster
ProxyPassReverseCookiePath / /pentaho
ProxyPass /pentaho-style balancer://reportingcluster-style
ProxyPassReverseCookiePath / /pentaho

Timeout 600
ProxyTimeout 600

<Proxy balancer://reportingcluster>
      ProxySet stickysession=JSESSIONID timeout=360 scolonpathdelim=On
failonstatus=502,503
      BalancerMember ajp://192.168.1.8:8009/pentaho connectiontimeout=10
loadfactor=10 route=server1 max=20 ttl=120 retry=300
      BalancerMember ajp://192.168.1.9:8009/pentaho connectiontimeout=10
loadfactor=10 route=server2 max=20 ttl=120 retry=300
</Proxy>

<Proxy balancer://reportingcluster-style>
      ProxySet stickysession=JSESSIONID timeout=360 scolonpathdelim=On
failonstatus=502,503
      BalancerMember ajp://192.168.1.8:8009/pentaho-style
connectiontimeout=10 loadfactor=10 route=server1 max=20 ttl=120 retry=300
      BalancerMember ajp://192.168.1.9:8009/pentaho-style
connectiontimeout=10 loadfactor=10 route=server2 max=20 ttl=120 retry=300
</Proxy>
```

Each proxy balancer has two servers in the cluster. If you have more than two servers in your cluster, you will need to add each of them.

4. After adding the above code, save the file and restart the Apache service.
5. You should now be able to browse to your load balancer and see the Pentaho User Console, for example: `http://loadbalancerhostname/pentaho`

Apache Configuration (Linux Environment)

There are different ways to configure Linux in a Windows environment. One example is presented here:

1. The following modules will need to be loaded:
   `proxy_ajp`
   `proxy_balancer`
   `proxy_http`
   `lbmethod_byrequests`
   You will need to use the proper OS-specific commands to load these modules. For example, with Debian-based distributions, run the command to load the required modules:

```
sudo a2enmod proxy proxy_ajp proxy_balancer proxy_http lbmethod_byrequests
```

For RPM-based distributions, you will need to uncomment these modules in the `httpd.conf` file to enable them, similar to the Windows configuration above. The modules may have already been uncommented by default.

2. Locate and edit the Apache configuration file. For Debian-based distributions, this is located by default in `/etc/apache2/sites-available/000-default.conf`. For RPM-based distributions, this is located by default in `/etc/httpd/conf/httpd.conf`.
3. Within this configuration file, you will need to edit the existing (or add a new) `VirtualHost` directive. Make sure to reflect the `BalancerMember` hostname with your Pentaho cluster nodes:

```
<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        <Location "/pentaho">
                ProxyPass balancer://reportingcluster
                ProxyPassReverseCookiePath / /pentaho
        </Location>
```

```
        <Location "/pentaho-style">
                ProxyPass balancer://reportingcluster-style
                ProxyPassReverseCookiePath / /pentaho
        </Location>
</VirtualHost>

ProxyPass /pentaho balancer://reportingcluster
ProxyPassReverseCookiePath / /pentaho
ProxyPass /pentaho-style balancer:// reportingcluster-style
ProxyPassReverseCookiePath / /pentaho

Timeout 600
ProxyTimeout 600

<Proxy balancer://reportingcluster>
        ProxySet stickysession=JSESSIONID timeout=360 scolonpathdelim=On
failonstatus=502,503
        BalancerMember ajp://192.168.1.8:8009/pentaho connectiontimeout=10
loadfactor=10 route=server1 max=20 ttl=120 retry=300
        BalancerMember ajp://192.168.1.9:8009/pentaho connectiontimeout=10
loadfactor=10 route=server2 max=20 ttl=120 retry=300
</Proxy>

<Proxy balancer://reportingcluster-style>
        ProxySet stickysession=JSESSIONID timeout=360 scolonpathdelim=On
failonstatus=502,503
        BalancerMember ajp://192.168.1.8:8009/pentaho-style
connectiontimeout=10 loadfactor=10 route=server1 max=20 ttl=120 retry=300
        BalancerMember ajp://192.168.1.9:8009/pentaho-style
connectiontimeout=10 loadfactor=10 route=server2 max=20 ttl=120 retry=300
</Proxy>
```

Each proxy balancer has two servers in the cluster. If you have more than two servers in your cluster, you will need to add each of them.

4. After adding the above code, save the file and restart the Apache service.
5. You should now be able to browse to your load balancer and see the Pentaho User Console, for example: `http://loadbalancerhostname/pentaho`

Summary of Apache Configuration

When you configure Apache, such as in the previous examples, some of the actions you take include the following:

Ensure Sticky Sessions are Enabled

Use `stickysession=JSESSIONID` to enable sticky sessions so that your session is tied to a single node. By default, Tomcat uses `JSESSIONID` as the name of its session cookie.

Configure `ProxyPass` and `ProxyPassReverseCookiePath` Directives

Both `ProxyPass` and `ProxyPassReverseCookiePath` are directives of the `mod_proxy` module. This allows Apache to map remote servers to the local server URL-space. In this instance, the remove servers are referencing clusters rather than a single server. You will also notice that two instances of these directives are required for proper Pentaho Server mapping.

1. The first set of directives needs to point to a cluster that uses the address `ajp://{hostname}:8009/pentaho`, which is the main Pentaho web application.
2. The second set of directives needs to point to a cluster that uses the address `ajp://{hostname}:8009/pentaho-style`, which is the web application used for images, stylesheets, and the overall look and feel of the User Console.

Configure `Proxy Balancer`

This is where you configure the actual cluster nodes. In a previous example, `Proxy balancer` was configured in such a way that traffic is loaded evenly across all nodes within the cluster. This is done with the `loadfactor` and `route` parameters. `Route` was configured with the same `jvmRoute` ID that was configured in the previous section on Tomcat Configuration. This is how Apache knows which node in the `Proxy balancer` maps to which Tomcat instance.

Other parameters can be used to specify certain counting algorithms that will distribute traffic based on server quota, as well as other parameters that can control other aspects of the load balancer. Apache's documentation has the list of parameters that can be used and descriptions of counting algorithms.

Due to better optimization and speed, the AJP protocol and corresponding Tomcat port have been used in the cluster node URLs. HTTP could be used instead; however, the cluster node's URL would not then be masked when traffic is forwarded from Apache. AJP will mask the cluster node URL with the load balancer's URL so that the redirect is hidden from the user. Other differences between the two protocols are documented on Apache's site in their FAQ on connectors.

Known Issues

Some known issues and solutions or workarounds are:

User Prompted to Reauthenticate/PUC Image Load Error

A user is redirected by the load balancer to the PUC login screen where they will enter their credentials. After selecting **Log in**, another authentication box will pop up asking for credentials again. Upon reauthenticating, PUC does not load all images for the home screen correctly.

Solution

This could be because the `pentaho-style` web application is not configured properly for the `ProxyPassReverseCookiePath` directive. Rather than use the exact examples above, you may need to change it from this:

```
<Location "/pentaho-style">
ProxyPass balancer://reportingcluster-style ProxyPassReverseCookiePath /
/pentaho
</Location>
```

To this:

```
<Location "/pentaho-style">
ProxyPass balancer://reportingcluster-style ProxyPassReverseCookiePath /
/pentaho-style
</Location>
```

Failover Behavior

The Pentaho Server does not support session failover. The net result of this is that while you can cluster Pentaho Servers for load-balancing, if any node in the cluster goes down, all users on that node will end up without a valid server session, and their next request will be sent to another node in the cluster where they would have to re-log-in (as if they had timed out).

For reference, failover is handled by the `failonstatus` parameter. In the example configuration in this document, server failover will happen when a request receives HTTP error codes 502 and 503.

Related Information

Here are some links to information that you may find helpful while using this best practices document:

Pentaho documentation:

- Cluster the Application Server: Configure Jackrabbit Journal
- Cluster the Application Server: Configure Quartz
- Use MS SQL Server as Your Repository Database (Manual Installation)
- Use MySQL as Your Repository Database (Manual Installation)
- Use Oracle as Your Repository Database (Manual Installation)
- Use PostgreSQL as Your Repository Database (Manual Installation)

External documentation:

- Quartz Configuration Reference: Configure Clustering with JDBC-JobStore
- AJP Connector
- Apache: Connector FAQ
- Apache: Module `mod_proxy`
- Apache: Module `mod_proxy_balancer`

Finalization Checklist

This checklist is designed to be added to any implemented project that uses this collection of best practices, to verify that all items have been considered and reviews have been performed.

Name of the Project:_____

Date of the Review:_____

Name of the Reviewer:_____

Item	Response	Comments
Did you configure Jackrabbit Journal and Quartz before starting and testing the cluster?	YES_____ NO_____	
Did you configure Tomcat and Apache for the load balancer?	YES_____ NO_____	