CESNET Technical Report 6/2013

Running Multiple Shibboleth IdP Instances on a Single Host

Ivan Novakov

Received 10.12.2013

Abstract

The article describes a way how multiple Shibboleth IdP instances may be run on a single host. It gives the required technical background and it expects that the reader has some knowledge of the Shibboleth IdP installation and configuration process.

Keywords: Shibboleth IdP, multiple instances, single host

1 Introduction

Shibboleth[1] is an open-source implementation for identity management and federated identity-based authentication and authorization (or access control) infrastructure based on Security Assertion Markup Language (SAML)[2]. Shibboleth Identity Provider (Shibboleth IdP) provides user authentication and single sign-on to requesting services and applications. Additionally, the IdP can provide a rich set of user-related data to the Service Provider.

This article expects the reader has some experience with Shibboleth and especially with the Shibboleth Identity Provider installation and configuration process. It focuses on specific actions during the installation and configuration related to the topic. It does not cover the whole process. For more detailed information, refer to the official documentation[3].

For various reasons it may be convenient to run multiple Shibboleth IdP instances on a single host. While it is not recommended, there may be cases, when such configuration is justified. While multiple service providers running on a single host is not a problem and the Shibboleth SP explicitly supports it, that is not the case with the identity provider. It is not possible to run multiple identity providers virtually in one Shibboleth Identity Provider instance. Each separate identity provider requires a separately installed instance.

The article describes one possible way, how to achieve this by setting up a single Tomcat instance with a single Apache 2.x instance as a frontend.

1 `http://shibboleth.net/`

2 `https://en.wikipedia.org/wiki/Saml`

3 `https://wiki.shibboleth.net/confluence/display/SHIB2/Installation+and+Configuration`

2 Prerequisites

Shibboleth IdP requires Java 6 or later and Tomcat version 6.0.17 or greater. Tomcat 7 is also supported, when using Apache as a frontend, but the following text assumes, that Tomcat 6 is being used. All 2.x versions of Apache are supported, but it is recommended to use Apache 2.2.

Let's suppose that we need to run two identity provider instances on the same host and each identity provider has to run under its own hostname. The hostnames will be `idp.one.org`and `idp.two.org`. There are two ways how to bind them to a single host.

The first way is to use CNAMEs - each hostname is a CNAME for the target host. While this may be easier to set up and requires a single IP address, there may be issues related to the SSL virtual host configuration. Using a single IP address and CNAMEs requires, that a name based virtual host configuration has to be used. This means, that you will need a single SSL certificate with multiple Subject Alternative Names[4]. In practice, this would not be a problem, if you have only few virtual hosts. But the more virtual hosts you have, the more difficult it gets to maintain.

Another way is to have a virtual IP address with an A DNS record for each instance. It is more flexible to maintain. You don't need name based virtual hosts and you can have a separate SSL certificate for each instance. Additionally, if you decide to move an instance to a separate host later, you can do it right away.

3 Apache configuration

The Apache configuration is straightforward and it doesn't differ too much from the single instance scenario. We need to set up a virtual host for each identity provider instance.

```
<VirtualHost {IP address}:443>
    ServerName idp.one.org
    #...
</VirtualHost>
<VirtualHost {IP address}:443>
    ServerName idp.two.org
    #...
</VirtualHost>
```

Inside each virtual host configuration we have to configure the AJP[5]connection to the Tomcat backend:

```
ProxyRequests Off
ProxyPass /idp/ ajp://localhost:8009/idp/
```

Next, we need to set the right certificate/key pair. If we have name based virtual hosts (the CNAME path), these settings are similar for all virtual hosts. Otherwise we have to set the corresponding certificate/key pair:

[4] https://en.wikipedia.org/wiki/SubjectAltName

[5] https://en.wikipedia.org/wiki/Apache_JServ_Protocol

```
SSLCertificateFile /etc/ssl/certs/{hostname}.crt.pem
SSLCertificateKeyFile /etc/ssl/private/{hostname}.key.pem
```

We also have to specify the server certificate chain - the CA certificates used to sign our server certificate:

```
SSLCertificateChainFile /etc/ssl/certs/ca-bundle.pem
```

It is also a good idea to have separate logs for each virtual host:

```
ErrorLog ${APACHE_LOG_DIR}/{hostname}/error.log
CustomLog ${APACHE_LOG_DIR}/{hostname}/ssl_access.log combined
```

4 Tomcat configuration

In Tomcat we need to set up virtual hosts as well. The Apache frontend communicates with the Tomcat backend through the AJP protocol[6]. For each Apache virtual host we need to configure the corresponding Tomcat virtual host. Apache virtual hosts relay HTTP requests through a single Tomcat endpoint - `localhost:8009`. But each request contains the corresponding *Host*HTTP header, which carries the hostname of the target IdP instance and allows Tomcat to choose the right virtual host. The only file we need to edit is `server.xml`.

First, let's configure the AJP connector:

```
<Connector port="8009" address="127.0.0.1" enableLookups="false"
    tomcatAuthentication="false"
    protocol="AJP/1.3" redirectPort="443" />
```

And we need to add a virtual host for all IdP instances (after the default "localhost" virtual host):

```
<Host name="idp.one.org"  appBase="webapps"
    unpackWARs="true" autoDeploy="true"
    xmlValidation="false" xmlNamespaceAware="false">
</Host>
<Host name="idp.two.org"  appBase="webapps"
    unpackWARs="true" autoDeploy="true"
    xmlValidation="false" xmlNamespaceAware="false">
</Host>
```

Now, when we restart Tomcat, we'll notice that for each virtual host the corresponding directory has been created under the `CATALINA_BASE`directory (usually it's `/etc/tomcat6/Catalina/`):

```
/etc/tomcat6/Catalina/idp.one.org/
/etc/tomcat6/Catalina/idp.two.org/
```

Later, after we install Shibboleth IdP, in each directory we'll place the corresponding context fragment diagram - a piece of configuration, which defines how the application (Shibboleth IdP) should be deployed.

---

[6] `https://en.wikipedia.org/wiki/Apache_JServ_Protocol`

5 Shibboleth IdP Installation

5.1 Overview

Shibboleth IdP doesn't support running virtual instances within a single installation, so we need a separate installation for each IdP instance. The installation process itself is not complicated. You have to download the latest release[7], unpack it and run the installation script `install.sh`.

The installation script asks for some information:

— the hostname of the IdP
— the target directory to install to
— the password for the keystore containing the certificate/key pair

For each IdP instance we need to run a separate installation specifying the corresponding IdP hostname and a separate directory.

Before running the installation script, you may want to customize[8]the design of the login page for each IdP instance. For convenience, you may maintain a separate source directory for each IdP instance in order to make easier future changes.

5.2 Example installation

Let's assume we need two IdP instances with hostnames `idp.one.org` and `idp.two.org`. We'll have our IdP instances installed under the `/opt/idp/` directory. We'll place the source directories under the `/opt/dist/` directory.

First, we need download the latest Shibboleth IdP release and unpack it:

```
$ cd /opt/dist
$ wget http://shibboleth.net/downloads/identity-provider/latest/
     shibboleth-identityprovider-2.x.x-bin.tar.gz
$ tar xvfz shibboleth-identityprovider-2.x.x-bin.tar.gz
```

Now we'll make a copy of the source directory for each IdP instance:

```
$ cp -r shibboleth-identityprovider-2.x.x/ idp.one.org-source/
$ cp -r shibboleth-identityprovider-2.x.x/ idp.two.org-source/
```

For each source directory we first need to customize the login page. It can be done by editing the `src/main/webapp/login.jsp`file. For more information, see the official documentation[9].

Then we can run the installation script `install.sh`. Again, for each instance we need to run a separate installation:

```
$ cd idp.one.org-source/
$ ./install.sh
```

When prompted, we enter hostname `idp.one.org`and target directory `/opt/idp/idp.one.org`. And for the second instance:

---

[7] `http://shibboleth.net/downloads/identity-provider/latest/`

[8] `https://wiki.shibboleth.net/confluence/display/SHIB2/IdPAuthUserPassLoginPage`

[9] `https://wiki.shibboleth.net/confluence/display/SHIB2/IdPAuthUserPassLoginPage`

```
$ cd ../idp.two.org-source/
$ ./install.sh
```

Here we enter hostname `idp.two.org`and target directory `/opt/idp/idp.two.org`.

Now, we have our two instances installed – the first in the `/opt/idp/idp.one.org` directory and the second in the `/opt/idp/idp.two.org` directory. We need to make sure, that Tomcat can write to the `logs/`subdirectory:

```
$ chown tomcat6 /opt/idp/idp.one.org/logs
$ chown tomcat6 /opt/idp/idp.two.org/logs
```

The next step is to deploy the IdP instances under Tomcat. The installation script creates a WAR[10] file ready for deployment. It is placed in the `war/`subdirectory of the target installation directory. So in our case we have two WAR files – `/opt/idp/idp.one.org/war/idp.war`and `/opt/idp/idp.two.org/war/idp.war`.

For each virtual host directory we'll create the corresponding context deployment fragment. Basically, it is a small XML configuration file, specifying the path to the WAR file to be deployed. So, for the first IdP instance we'll create file `/etc/tomcat6/Catalina/idp.one.org/idp.xml`with the following content:

```
<Context docBase="/opt/idp/idp.one.org/war/idp.war"
         privileged="true"
         antiResourceLocking="false"
         antiJARLocking="false"
         unpackWAR="false"
         swallowOutput="true"
         cookies="false" />
```

And for the second IdP instance we'll create file `/etc/tomcat6/Catalina/idp.two.org/idp.xml`:

```
<Context docBase="/opt/idp/idp.two.org/war/idp.war"
         privileged="true"
         antiResourceLocking="false"
         antiJARLocking="false"
         unpackWAR="false"
         swallowOutput="true"
         cookies="false" />
```

Now, our two IdP instances should be deployed and there should be some logs under the `logs/`subdirectory. If that is not the case, try to restart Tomcat and make sure the `logs/`subdirectories are writable by Tomcat.

[10] `https://en.wikipedia.org/wiki/WAR_file_format_%28Sun%29`

6 Shibboleth IdP Configuration

The minimum required configuration for each IdP instance includes:

— setting unique entity ID (`conf/relying-party.xml`)
— specifying metadata source (`conf/relying-party.xml`)
— configuring authentication (`conf/handler.xml`, `conf/relying-party.xml`, `conf/login.config`)
— configuring basic attribute resolution and release (`conf/attribute-resolver.xml`, `conf/attribute-filter.xml`)

Detailed Shibboleth IdP configuration is beyond the scope of this article. For more information, see the original documentation[11].

7 Maintenance suggestions

7.1 Adding another instance

Considering the example installation above, adding a new IdP instance should be fairly easy. Let's add a new instance with hostname `idp.three.org`. First, add a new Apache virtual host:

```
<VirtualHost {IP address}:443>
    ServerName idp.three.org
    #...
</VirtualHost>
```

Then add a new Tomcat virtual host in `/etc/tomcat6/server.xml`:

```
<Host name="idp.three.org"  appBase="webapps"
    unpackWARs="true" autoDeploy="true"
    xmlValidation="false" xmlNamespaceAware="false">
</Host>
```

Restart Tomcat and a new directory `/etc/tomcat6/Catalina/idp.three.org` will be created.

Create a new copy of the Shibboleth IdP source directory for the new instance:

```
$ cd /opt/dist
$ cp -r shibboleth-identityprovider-2.x.x/ idp.three.org-source/
```

Customize the login page by editing the `src/main/webapp/login.jsp` file. Then run the installation script `install.sh`. When prompted, enter hostname `idp.three.org` and target directory `/opt/idp/idp.three.org`.

Deploy the instance to Tomcat by creating a context deployment fragment `/etc/tomcat6/Catalina/idp.three.org/idp.xml`:

[11] `https://wiki.shibboleth.net/confluence/display/SHIB2/IdPConfiguration`

```
<Context docBase="/opt/idp/idp.three.org/war/idp.war"
         privileged="true"
         antiResourceLocking="false"
         antiJARLocking="false"
         unpackWAR="false"
         swallowOutput="true"
         cookies="false" />
```

Configure the instance by setting the entity ID, metadata source, authentication and attribute release.

8 Handling multiple IdP instance configuration

If you have multiple IdP instances which have similar configuration, it may be convenient to implement some sort of configuration generation based on templates. It could be particularly useful in cases when only several ”variables” are different across multiple IdP instances, such as - entity ID, organization name or department unit.

At the same time, you may use a version control system (VCS) to store your configuration and templates. This will allow you to maintain different versions of your configuration and to be able to track changes easily.

9 Issues with this solution

The biggest advantage is the biggest problem at the same time - there is a single host with a single Apache instance and a single Tomcat instance to handle all IdP instances. While the ”single host” problem itself can be solved by implementing a suitable high-availability solution, the ”single Apache/Tomcat” instance is still an issue.

Whenever something affects Apache/Tomcat, it affects all IdP instances at once. For example, if you need to modify the configuration of a single IdP instance you need to restart Tomcat afterwards. As a result:

— all IdP instances suffer from a brief outage while Tomcat is being restarted
— all login sessions from all IdP instances are lost, unless you don’t have implemented an alternative session storage to ensure session persistence

Another major problem could be performance issues due to having too many IdP instances. Each instance runs as a Tomcat application and consumes certain amount of memory and CPU time. You have to make sure, that you have enough hardware resources to run run all your IdP instances.

References

[1] https://en.wikipedia.org/wiki/Shibboleth_(Internet2)

[2] https://en.wikipedia.org/wiki/SubjectAltName

[3] http://shibboleth.net/products/identity-provider.html

[4] https://wiki.shibboleth.net/confluence/display/SHIB2/IdPAuthUserPassLoginPage

[5] http://tomcat.apache.org/tomcat-6.0-doc/virtual-hosting-howto.html