IdPInstall

End of Life Warning

All maintenance for the Shibboleth Identity Provider V2 release branch ceased on July 31, 2016. All deployments should have upgraded to V3 or evaluate alternatives.

This is an installation guide. For an introduction to the identity provider or Shibboleth, please refer to Understanding Shibboleth.

Before You Begin

Before you begin you should collect the following things:

- an SSL certificate that you'll use to secure your IdP's browser-facing HTTP connection
- a source of SAML Metadata for the service providers your IdP will communicate with (this could come from a Federation you've joined, directly from the SP owners, or created and maintained by hand)

The installation process will generate the following information for you:

- the IdP's entity ID
- a keypair and self-signed certificate used for signing (these are not used for browser-facing HTTP connections to your server)
- the IdP's initial metadata
- a basic set of IdP configuration files based on this information

Red Hat/CentOS Users, Stop!

Some older versions of Red Hat Enterprise Linux and CentOS ship with the GNU Java compiler and VM (`gcj`) by default. These are not usable with Shibboleth so you must install another JVM.

Debian OpenJDK6 warning

There have been reports of problems running the Shibboleth IdP on Debian 7 ("Wheezy") with an openjdk-6 package, which could be resolved by switching to openjdk-7 (`openjdk-7-jre-headless` will generally be sufficient).

OpenJDK Warning

We strongly recommend the use of Oracle's "standard" JVM on all platforms. The OpenJDK implementation that ships with many Linux distributions is used by many deployers, but the community has off and on reported various problems that have frequently been traced to the use of OpenJDK, including memory leaks, the Debian issue above, and others. You should expect that reports of unexplained problems may be met with a request to reproduce them on Oracle's JVM.

Performing the Install

The V2 Shibboleth Identity Provider is a standard Java web application based on the Servlet 2.4 specification and should run for the most part in any compatible servlet container. If you are unsure which to choose, most people use Apache Tomcat today, but Jetty is the strongly preferred option and is used by the primary team members in their production environments.

The officially supported containers are Jetty 7+ and Tomcat 6+. Containers for which we have specific installation help are shown in step 1 below, including some that we do not officially support. Material specific to any container is provided as a convenience, and is not a substitute for the container's own documentation.

1. Prepare your Servlet container: Jetty 9, Jetty 7, Apache Tomcat, JBoss Tomcat, Glassfish
    - Linux deployers may want to take a look at IdPLinuxNonRoot.
2. Download the latest Identity Provider software package.
3. Unzip the archive you downloaded: *jar -xf shibboleth-identityprovider-VERSION-bin.zip*
4. Change into the newly created IdP distribution directory, *shibboleth-identityprovider-VERSION*
5. Run either **./install.sh** (on Unix systems) or **install.bat**(on Windows systems).
    - The installation directory given during installation will be known as */IDP_HOME* throughout this document.
6. Deploy the IdP WAR file, located in */IDP_HOME/war/*. See the Servlet container preparation notes for the best approach for doing this.

After the installation script has completed the IdP home directory will have been created, here's a brief description of what you'll find in it:

- **bin/** - This directory contains various tools useful in running, testing, or deploying the IdP
- **conf/** - This directory contains all the configuration files for the IdP
- **credentials/** - This is where the IdP's signing and encryption credential, called *idp.key* and *idp.crt*, is stored
- **lib/** - This directory contains various code libraries used by the tools in *bin/*
- **logs/** - This directory contains the log files for the IdP
- **metadata/** - This is the directory in which the IdP will store its metadata, in a file called *idp-metadata.xml*. It is recommend you store any other retrieved metadata here as well.

- **war/** - This contains the web application archive (war) file that you will deploy into the servlet container

> **Generated IdP metadata is NOT dynamically updated**
>
> The IdP metadata generated at install won't be dynamically updated if you change the Identity Provider's configuration or certificates. It is **your** responsibility to maintain this.

A Quick Test

You can test that the IdP is properly installed and running by accessing the URL: `https://HOSTNAME/idp/profile/Status`. If everything is working correctly you should receive an "ok" page. This doesn't mean that you will be able to log into anything yet as you have not yet configured the IdP to use your organization's infrastructure.

> There is a second status page that provides additional information about the IdP.

Next Steps

After installation you will normally need to perform two steps in order to have a basic setup:

1. Load SAML metadata for the service provider(s) with which you will interact.
2. Configure an authentication mechanism.

After you have finished that the next step is usually to collect and release attributes.

Advanced Installation Topics

> Version 2.3 and later

Changing the lifetime of the self signed certificate

During first installation a self signed certificate with a lifetime of 20 years is generated. This lifetime can be adjusted by setting the environmental variable **`IdPCertLifetime`** to the number of years desired.

Using a customized web.xml

During all installations, if a file called `web.xml` in the `conf` subdirectory of the IdP installation directory exists it is used in preference to the default file. This allows a customized `web.xml` to be carried from release to release.

Regenerating credentials

If you need to regenerate credentials without reinstalling the IdP, see IdPCertRenew.

Add content to the Velocity templates used for POST web page returned to user's browser after authentication (the web page that contains the auto-submit to the appropriate SP endpoint)

> Version 2.4 and later

Starting with version 2.4.0 of the IdP, there is an easier way to add content to the default Velocity templates that generate the POST response web page that is returned to the user's browser, and then auto-submitted to the appropriate SP endpoint. One has always been able to override those Velocity templates altogether, and create your own templates, if you know what you are doing. But this new feature for the IdP makes it much easier to make particular kinds of additions to those pages without needing to completely override the default pages.

A couple of examples of things you could do with this would be to add Javascript to invoke Google Analytics to record stats about logins, or to add a message to the page in case network delays cause this page to be displayed to the user long enough that the user wonders what is happening next.

Details on how to take advantage of this feature, and an example of adding Head and Body content, are found at AddPostPageHeadBodyContent.