
simpleSAMLphp Installation and Configuration

Andreas Åkre Solberg <andreas.solberg@uninett.no>

Fri Sep 14 10:49:49 2007

Table of Contents

The history of simpleSAMLphp	1
Changelog	2
Version 0.4	2
Download and get simpleSAMLphp	2
Getting a working copy of simpleSAMLphp from subversion	2
Installing simpleSAMLphp	3
The simpleSAMLphp installation webpage	3
Making configuration and metadata files	3
Configuring simpleSAMLphp	4
Configuration for LDAP authentication plugin	4
Setting up a SAML 2.0 SP	4
Configuring metadata for a SAML 2.0 SP	4
Test the SAML 2.0 SP example	5
Setting up a Shibboleth 1.3 SP	5
Configuring metadata for Shibboleth 1.3 SP	5
Test the Shibboleth 1.3 SP example	6
Setting up a SAML 2.0 IdP	6
Configuring the SAML 2.0 IdP	6
Adding a SAML IdP signing certificate	7
Test SAML 2.0 IdP	7
Using the built-in SP WAYF functionality	7
Setting up WebSSO bridges	7
Bridging SAML 2.0 <-> SAML 2.0	7
Bridging Shibboleth 1.3 <-> Shibboleth 1.3	8
Bridging Shibboleth 1.3 <-> SAML 2.0	8
Bridging SAML 2.0 <-> Shibboleth 1.3	8
Bridging SAML 2.0 <-> OpenID	8
Bridging Shibboleth 1.3 <-> OpenID	8
Authentication API	8

The history of simpleSAMLphp

simpleSAMLphp is based on code from Sun OpenSSO Extensions [<https://opensso.dev.java.net/public/extensions/>] (formerly known as Lightbulb).

The initial versions of the SAML 2.0 SP part was written by Pat Patterson, Sun [<http://blogs.sun.com/superpat/>].

The functionality has been extended and Andreas Åkre Solberg [<http://claimid.com/erlang>], UNINETT [<http://uninett.no>], has rewritten the library and added support for Shibboleth. The product is used to bridge AAI protocols in the GÉANT project, <http://geant2.net>.

Changelog

Here is changes between simpleSAML versions. Look here if you are upgrading, to see if there are any changes to the config format.

Version 0.4

Released 2007-09-14. Revision X.

- Improved documentation
- Authentication plugin API. Only LDAP authentication plugin is included, but it is now easier to implement your own plugin.
- Added support for SAML 2.0 IdP to work with Google Apps for Education. Tested.
- Initial implementation of SAML 2.0 Single Log-Out functionality both for SP and IdP. Seems to work, but not yet well-tested.
- Added support for bridging SAML 2.0 to SAML 2.0.
- Added some time skew offset to the NotBefore timestamp on the assertion, to allow some time skew between the SP and IdP.
- Fixed Browser/POST page to automatically submit, and have fall back functionality for user agents with no javascript support.
- Fixed some bug with warning traversing Shibboleth 1.3 Assertions.
- Fixed tabindex on the login page of the LDAP authentication module to allow you to tab from username, to password and then to submit.
- Fixed bug on autodiscovering hostname in multihost environments.
- Cleaned out some debug messages, and added a debug option in the configuration file. This debug option let's you turn on the possibility of showing all SAML messages to users in the web browser, and manually submit them.
- Several minor bugfixes.

Download and get simpleSAMLphp

You can go to <http://rnd.feide.no/category/simplesamlphp/> to find the most recent release of simpleSAMLphp. Download the zipped file, and unzip it on your webserver. However I highly recommend running from a subversion checkout instead.

Getting a working copy of simpleSAMLphp from subversion

Warning

Right now the subversion repository is requiring a username / password. I'll update the access control, so that everyone can get read access without authentication. I'll announce it on the rnd blog when it is ready.

If you want a working copy from subversion enter:

```
svn co https://svn.uninett.no/svn/feidernd/simplesamlphp
```

If you know subversion you know how to view logs and review changes to the files. To update the version you have checked out, enter:

```
cd simplesamlphp
svn up
```

Installing simpleSAMLphp

First find an appropriate place for the `simplesamlphp` folder. In example `/var/simplesamlphp`.

Of the folders inside `simplesamlphp`, only the `www` folder needs to be accessible from the web. There are several ways of putting the `simpleSAMLphp` depending on the way web sites are structured on your apache web server. Here is what I believe is the best configuration.

Find the apache configuration file for the virtual hosts that you want to run `simpleSAML` on. The configuration may look like this:

```
<VirtualHost *>
    ServerName service.example.com
    DocumentRoot /var/www/service.example.com

    Alias /simplesamlphp /var/simplesamlphp/www
</VirtualHost>
```

What is special is the `Alias` directive. That directive will give control to `simplesamlphp` to all urls that matches `http(s)://service.example.com/simplesamlphp/*`. `SimpleSAML` will need to have several SAML interfaces available on the web, and all these interfaces are included in the `www` subdirectory of your `simplesamlphp` installation. You can set the alias to whatever you want, but this alias must be set in the `config.php` file of `simpleSAML` as described in the section called “Configuring `simpleSAMLphp`”. Here is an example of how this configuration may look like in `config.php`:

```
$config = array (
    'basedir'      => '/var/simplesamlphp/',
    'baseurl'     => 'http://service.example.com',
    'baseurlpath' => 'simplesamlphp/',
```

The simpleSAMLphp installation webpage

When you have installed `simpleSAMLphp`, you can access the homepage of your installation, which contains some information and a few links to the test services. The url of an installation can be in example:

```
https://service.example.com/simplesamlphp/
```

But it depends on how you set it up with apache.

Making configuration and metadata files

Configuration and metadata files are stored in a template format, you need to copy them to have your local copies. The reason why it is done this way, is that when you upgrade you can do `svn up` in subversion or just copy the whole directory over your installation, without replacing your existing configuration. When

you are updating, you should investigate whether the config format is changed, this should be documented in the changelog.

Here are the steps you need to do to create local configuration files:

```
cd /var/simplesamlphp
cp config/config-template.php config/config.php
cp -r metadata-templates/*.php metadata/
```

Configuring simpleSAMLphp

First configure all the paths in the beginning of the config file, to correspond to your organization of the apache web server, and where you place simpleSAMLphp.

You will need to set the entityid of a default IdP in `default-saml2-idp` or `default-shib13-idp` depending on whether you use shibboleth or SAML 2.0.

There is one parameter `debug` that may be set to true or false. If you set it to true, then all Browser/POST SAML messages will be printed to the web browser, and the user will have to manually submit it.

The `session.duration` parameter says how many seconds that a session should be valid. After this amount of time, the session is not valid anymore.

Configuration for LDAP authentication plugin

If you want to perform local authentication on this server, and you want to use the LDAP authentication plugin, then you need to configure the following parameters:

- `auth.ldap.dnpattern`: What DN should you bind to? Replacing `%username%` with the username the user types in.
- `auth.ldap.hostname`: The hostname of the LDAP server
- `auth.ldap.attributes`: Search parameter to LDAP. What attributes should be extracted? `objectclass=*` gives you all.

Setting up a SAML 2.0 SP

This functionality is relevant if you want to integrate SAML 2.0 authentication on a service of yours, and you know one or more IdPs that you can connect to. You would need metadata for those IdPs.

Configuring metadata for a SAML 2.0 SP

To configure a SAML 2.0 SP, you first need to configure the SP data for all your vhosts. If you run only one host, you need only one entry. This metadata is stored in the `metadata/saml20-sp-hosted.php` file. Here is an example of a metadata:

```
"dev.andreas.feide.no" => array(
  'host'           => 'dev.andreas.feide.no',
  'assertionConsumerServiceURL' => "http://dev.andreas.feide.no/saml2/sp/AssertionConsumerServiceURL",
  'issuer'         => "dev.andreas.feide.no",
  'spNameQualifier' => "dev.andreas.feide.no",
  'ForceAuthn'    => "false",
```

```
"NameIDFormat" => "urn:oasis:names:tc:SAML:2.0:nameid-format:transient"  
) ,
```

Note that you should fill in the host field matching the hostname of your vhost. That way simpleSAMLphp can automatically detect what SP metadata to use based on the `Host` : header sent by the HTTP user agent.

You also need to configure the metadata for the IdP that you want to use. Here is a metadata example for the Feide IdP:

```
"sam.feide.no" => array(  
  "SingleSignOnUrl" => "https://sam.feide.no/amserver/SSORedirect/metaAlias/idp",  
  "SingleLogoutUrl" => "https://sam.feide.no/amserver/IDPSloRedirect/metaAlias/i",  
  "certFingerprint" => "3a:e7:d3:d3:06:ba:57:fd:7f:62:6a:4b:a8:64:b3:4a:53:d9:5d",  
  "base64attributes" => true),
```

The IdP metadata is stored in the `metadata/saml20-idp-remote.php` file. Configure the correct URLs of the endpoints, the hash of the certificate, and whether the IdP is base64 encoding attributes or not. Most IdPs don't use base64, so if you do not connect to Feide you should turn this parameter to `false`. Notice that the key of the array is the entity id of the IdP, in this example: `sam.feide.no`.

Test the SAML 2.0 SP example

Go to the URL of the test page, similar to:

```
http://service.example.com/simplesamlphp/example-simple/saml2-example.php
```

Note

The simpleSAMLphp installation homepage will link you to this example, so you do not need to type in the full url.

You should be redirected to the IdP. Login, and you should be sent back and shown all the attributes sent from the IdP.

Setting up a Shibboleth 1.3 SP

If you want to configure a service with authentication towards an external Shibboleth 1.3 IdP, this section describes you how to proceed.

Configuring metadata for Shibboleth 1.3 SP

Configure Shibboleth 1.3 SP metadata for all your vhosts. If you run only one host, you need only one entry. This metadata is stored in the `metadata/shib13-sp-hosted.php` file. Here is an example:

```
'http://dev.andreas.feide.no' => array(  
  'AssertionConsumerService' => 'http://dev.andreas.feide.no/shib13/sp/AssertionCo',  
  'host' => 'dev.andreas.feide.no'  
) ,
```

Note that you should fill in the host field matching the hostname of your vhost. That way simpleSAMLphp can automatically detect what SP metadata to use based on the `Host` : header sent by the HTTP user agent.

You also need to configure the metadata for the Shibboleth 1.3 IdPs that you want to connect to. Here is an example:

```
'urn:mace:switch.ch:aaitest:dukono.switch.ch' => array(
  'SingleSignOnUrl' => 'https://dukono.switch.ch/shibboleth-idp/SSO',
  'certFingerprint' => 'c7279a9f28f11380509e075441e3dc55fb9ab864'
),
```

Notice that the key of the array is the entity ID.

Test the Shibboleth 1.3 SP example

Go to the URL of the shibboleth test page, similar to:

```
http://service.example.com/example-simple/shib13-example.php
```

You should be redirected to the IdP. Login, and you should be sent back and shown all the attributes sent from the IdP.

Note

simpleSAMLphp does not support the attribute profile that Shibboleth is using by default. To make attributes work, you need to configure the IdP to perform attribute push.

Setting up a SAML 2.0 IdP

If you have a user database and want to offer a SAML 2.0 IdP functionality towards external services, here is how you set it up.

Configuring the SAML 2.0 IdP

Setup idp metadata in saml20-idp-hosted. Then for all the SP the IdP should trust in saml20-sp-remote. Then configure in config.php, ldap DN patterns, ldap host etc. Next add a certificate with openssl.

Example config.php:

```
'auth.ldap.dnpattern' => 'uid=%username%,dc=feide,dc=no,ou=feide,dc=uninett,dc=no',
'auth.ldap.hostname' => 'ldap.uninett.no',
'auth.ldap.attributes' => 'objectclass=*
```

Example IdP Metadata saml20-idp-hosted:

```
'dev2.andreas.feide.no' => array(
  'host' => 'dev2.andreas.feide.no',
  'SingleSignOnUrl' => "http://dev2.andreas.feide.no/saml2/idp/SSOService.php",
  'SingleLogoutUrl' => "http://dev2.andreas.feide.no/saml2/idp/LogoutService.php",
  'privatekey' => 'server.pem',
  'certificate' => 'server.crt',
  'base64attributes' => true,
  'auth' => 'auth/login.php'
)
```

The server.pem and server.crt is an example certificate shipped with the package, and be used for demo purposes, but you must generate your own to use in production services.

You also need to configure metadata for trusted SPs, here is an example:

```
_ "dev.andreas.feide.no" => array(
```

```
'host'          => 'dev.andreas.feide.no',  
  "assertionConsumerServiceURL" => "http://dev.andreas.feide.no/saml2/sp/Assertio  
"issuer"        => "dev.andreas.feide.no",  
"spNameQualifier"    => "dev.andreas.feide.no",  
"ForceAuthn"       => "false",  
"NameIDFormat"     => "urn:oasis:names:tc:SAML:2.0:nameid-format:transient"  
) ,
```

Adding a SAML IdP signing certificate

You should generate a new certificate for your IdP.

Warning

There is a certificate that follows this package that you can use for test purposes, but off course NEVER use this in production as the private key is also included in the package and can be downloaded by anyone.

Here is an examples of openssl commands to generate a new key and a selfsigned certificate to use for signing SAML messages:

```
openssl genrsa -des3 -out server2.key 1024  
openssl rsa -in server2.key -out server2.pem  
openssl req -new -key server.key -out server2.csr  
openssl x509 -req -days 60 -in server2.csr -signkey server2.key -out server2.crt
```

The certificate above will be valid for 60 days.

Note

simpleSAMLphp will only work with RSA and not DSA certificates.

Test SAML 2.0 IdP

To test the SAML 2.0 IdP, it is best to configure two hosts with simpleSAMLphp, and use the SAML 2.0 SP demo example to test the IdP.

Using the built-in SP WAYF functionality

The WAYF is not yet a part of the simpleSAMLphp release. This functionality will be added soon.

Setting up WebSSO bridges

simpleSAMLphp can be used to bridge between two WebSSO protocols. Here is some short descriptions of how to setup the different bridge configurations.

Bridging SAML 2.0 <-> SAML 2.0

In this setup you can bridge between two federations using SAML 2.0.

To approach this, you must configure both saml 2.0 IdP and SP hosted metadata, and in the IdP hosted metadata configure the auth parameter to be the SP initialization endpoint, like this:

```
'auth' => 'saml2/sp/initSSO.php?idpentityid=sam.feide.no'
```

As you can see you specify the IdP in the remote federation as a parameter to the initialization endpoint.

Note

This section of the documentation is only a placeholder. There will be more detailed information added later. For now, ask the author if you want more details of such a setup.

Bridging SAML 2.0 SLO is not implemented. Will be improved soon.

Bridging Shibboleth 1.3 <-> Shibboleth 1.3

Will be supported soon.

Bridging Shibboleth 1.3 <-> SAML 2.0

Will be supported soon.

Bridging SAML 2.0 <-> Shibboleth 1.3

Will be supported soon.

Bridging SAML 2.0 <-> OpenID

Will be supported soon.

Bridging Shibboleth 1.3 <-> OpenID

Will be supported soon.

Authentication API

The authentication plugin should be placed in the auth directory.

The following parameters must be accepted in the incoming URL:

- `RelayState`: This is the URL that the user should be sent back to after authentication within the plugin.
- `RequestID`: This is the ID of an incoming request.

The `initSSO.php` takes in addition the following parameters:

- `idpentityid`: This is the entityid of the IdP to authenticate with. This parameter is optional, if not set the default for this host will be used.
- `spentityid`: This is which SP config to use. This parameter is optional, if not set the default for this host will be used.

In hosted IdP metadata there is a config parameter `auth` that will tell simpleSAML which authentication plugin that can be used.

Tip

The authentication API is pretty basic. The easiest way to understand how it works is to look at one of the existing plugins that is located in the auth directory of your installation.