SimpleSAMLphp Identity Provider QuickStart

This guide will describe how to configure SimpleSAMLphp as an identity provider (IdP). You should previously have installed SimpleSAMLphp as described in the SimpleSAMLphp installation instructions

Enabling the Identity Provider functionality

The first that must be done is to enable the identity provider functionality. This is done by editing `config/config.php` . The option `enable.saml20-idp` controls whether SAML 2.0 IdP support is enabled. Enable it by assigning `true` to them:

```
'enable.saml20-idp' => true,
```

Authentication module

The next step is to configure the way users authenticate on your IdP. Various modules in the `modules/` directory provides methods for authenticating your users. This is an overview of those that are included in the SimpleSAMLphp distribution:

`authcrypt:Hash`
: Username & password authentication with hashed passwords.

`authcrypt:Htpasswd`
: Username & password authentication against .htpasswd file.

`authX509:authX509userCert`
: Authenticate against a LDAP database with a SSL client certificate.

`exampleauth:UserPass`
: Authenticate against a list of usernames and passwords.

`exampleauth:Static`
: Automatically log in as a user with a set of attributes.

`ldap:LDAP`
: Authenticates an user to a LDAP server.

`ldap:LDAPMulti`
: Authenticates an user to one of several LDAP server. The user can choose the LDAP server from a dropdown list.

`sqlauth:SQL`
: Authenticate an user against a database.

`radius:Radius`
: Authenticates an user to a Radius server.

`multiauth:MultiAuth`
: Allow the user to select from a list of authentication sources.

`saml:SP`
: Authenticate against a SAML IdP. Can be used for bridging.

`authYubiKey:YubiKey`
: Authenticate with a YubiKey .

`authtwitter:Twitter`
: Authenticate with your Twitter account using the Twitter OAuth API.

`papi:PAPI`
: Authenticate by means of the PAPI protocol.

In this guide, we will use the `exampleauth:UserPass` authentication module. This module does not have any dependencies, and is therefore simple to set up.

Configuring the authentication module

The `exampleauth:UserPass` authentication module is part of the `exampleauth` module. This module isn't enabled by default, so you will have to enable it. In `config.php` , search for the `module.enable` key and set `exampleauth` to true:

```
    'module.enable' => [
        'exampleauth' => true,
        …
    ],
```

The next step is to create an authentication source with this module. An authentication source is an authentication module with a specific configuration. Each authentication source has a name, which is used to refer to this specific configuration in the IdP configuration. Configuration for authentication sources can be found in `config/authsources.php` .

In this setup, this file should contain a single entry:

```
<?php
$config = [
  'example-userpass' => [
    'exampleauth:UserPass',
    'student:studentpass' => [
      'uid' => ['student'],
      'eduPersonAffiliation' => ['member', 'student'],
    ],
    'employee:employeepass' => [
      'uid' => ['employee'],
      'eduPersonAffiliation' => ['member', 'employee'],
    ],
  ],
];
```

This configuration creates two users - `student` and `employee` , with the passwords `studentpass` and `employeepass` . The username and password are stored in the array index (`student:studentpass` for the `student` -user). The attributes for each user are configured in the array referenced by the index. So for the student user, these are:

```
[
  'uid' => ['student'],
  'eduPersonAffiliation' => ['member', 'student'],
],
```

The attributes will be returned by the IdP when the user logs on.

Creating a self signed certificate

The IdP needs a certificate to sign its SAML assertions with. Here is an example of an `openssl` -command which can be used to generate a new private key key and the corresponding self-signed certificate. The private key and certificate go into the directory defined in the certdir setting (defaults to `cert/`)

This key and certificate can be used to sign SAML messages:

```
openssl req -newkey rsa:3072 -new -x509 -days 3652 -nodes -out example.org.crt -keyout example.org.pem
```

The certificate above will be valid for 10 years.

Note

SimpleSAMLphp will only work with RSA certificates. DSA certificates are not supported.

Configuring the IdP

The SAML 2.0 IdP is configured by the metadata stored in `metadata/saml20-idp-hosted.php` . This is a minimal configuration:

```
<?php
$metadata['__DYNAMIC:1__'] = [
    /*
     * The hostname for this IdP. This makes it possible to run multiple
     * IdPs from the same configuration. '__DEFAULT__' means that this one
     * should be used by default.
     */
    'host' => '__DEFAULT__',

    /*
     * The private key and certificate to use when signing responses.
     * These are stored in the cert-directory.
     */
    'privatekey' => 'example.org.pem',
    'certificate' => 'example.org.crt',

    /*
     * The authentication source which should be used to authenticate the
     * user. This must match one of the entries in config/authsources.php.
     */
    'auth' => 'example-userpass',
];
```

For more information about available options in the idp-hosted metadata files, see the IdP hosted reference .

Using the `uri` NameFormat on attributes

The interoperable SAML 2 profile specifies that attributes should be delivered using the `urn:oasis:names:tc:SAML:2.0:attrname-format:uri` NameFormat. We therefore recommended enabling this in new installations. This can be done by adding the following to the saml20-idp-hosted configuration:

```
    'attributes.NameFormat' => 'urn:oasis:names:tc:SAML:2.0:attrname-format:uri',
    'authproc' => [
        // Convert LDAP names to oids.
        100 => ['class' => 'core:AttributeMap', 'name2oid'],
    ],
```

Adding SPs to the IdP

The identity provider you are configuring needs to know about the service providers you are going to connect to it. This is configured by metadata stored in `metadata/saml20-sp-remote.php` . This is a minimal example of a `metadata/saml20-sp-remote.php` metadata file for a SimpleSAMLphp SP:

```
<?php
$metadata['https://sp.example.org/simplesaml/module.php/saml/sp/metadata.php/default-sp'] = [
    'AssertionConsumerService' => 'https://sp.example.org/simplesaml/module.php/saml/sp/saml2-acs.php/default-sp',
    'SingleLogoutService'      => 'https://sp.example.org/simplesaml/module.php/saml/sp/saml2-logout.php/default-sp',
];
```

Note that the URI in the entityID and the URLs to the AssertionConsumerService and SingleLogoutService endpoints change between different service providers. If you have the metadata of the remote SP as an XML file, you can use the built-in XML to SimpleSAMLphp metadata converter, which by default is available as `/admin/metadata-converter.php` in your SimpleSAMLphp installation.

For more information about available options in the sp-remote metadata files, see the SP remote reference .

Adding this IdP to other SPs

The method for adding this IdP to a SP varies between different types of SPs. In general, most SPs need some metadata from the IdP. This should be available from `/saml2/idp/metadata.php` .

Testing the IdP

The simplest way to test the IdP is to configure a SimpleSAMLphp SP on the same machine. See the instructions for configuring SimpleSAMLphp as an SP .

Note

When running a SimpleSAMLphp IdP and a SimpleSAMLphp SP on the same computer, the SP and IdP **MUST** be configured with different hostnames. This prevents cookies from the SP to interfere with cookies from the IdP.

Support

If you need help to make this work, or want to discuss SimpleSAMLphp with other users of the software, you are fortunate: Around SimpleSAMLphp there is a great Open source community, and you are welcome to join! The forums are open for you to ask questions, contribute answers other further questions, request improvements or contribute with code or plugins of your own.

- SimpleSAMLphp homepage
- List of all available SimpleSAMLphp documentation
- Join the SimpleSAMLphp user's mailing list

A. IdP-first setup

If you do not want to start the SSO flow at the SP, you may use the IdP-first setup. To do this, redirect the user to the SSOService endpoint on the IdP with one parameter `spentityid` that match the SP EntityId that the user should be logged into.

Here is an example of such a URL:

```
https://idp.example.org/simplesaml/saml2/idp/SSOService.php?spentityid=sp.example.org
```

If the SP is a SimpleSAMLphp SP, you must also specify a `RelayState` parameter for the SP. This must be set to a URL the user should be redirected to after authentication. The `RelayState` parameter can be specified in the SP configuration , or it can be sent from the IdP. To send the RelayState parameter from a SimpleSAMLphp IdP, specify it in the query string to SSOService.php:

```
https://idp.example.org/simplesaml/saml2/idp/SSOService.php?spentityid=sp.example.org&RelayState=https://sp.example.org/welcome.php
```

To set it in the SP configuration, add it to `authsources.php` :

```
'default-sp' => [
    'saml:SP',
    'RelayState' => 'https://sp.example.org/welcome.php',
],
```