**COMPETITIVENESS AND INNOVATION FRAMEWORK PROGRAMME**

ICT Policy Support Programme (ICT PSP)

Towards pan-European recognition of electronic IDs (eIDs)

Project acronym: STORK

Project full title: Secure Identity Across Borders Linked

STORK-PHP Service Providers

Start Guide

|  |  |
| --- | --- |
| **Deliverable Id :** |  |
| **Deliverable Name :** |  |
| **Status :** |  |
| **Dissemination Level :** |  |
| **Due date of deliverable :** |  |
| **Actual submission date :** |  |
| **Work Package :** | **WP5.8** |
| **Organisation name of lead contractor for this deliverable :** |  |
| **Author(s):** | **Hugo Magalhães** |
| **Partner(s) contributing :** | **PT,ES** |

**Abstract:** Quick Guide to configure and deploy STORK-PHP Service Providers.

**Project co-funded by the European Community under the ICT Policy Support Programme**

© Copyright by the STORK-eID Consortium

History

|  |  |  |  |
| --- | --- | --- | --- |
| *Version* | *Date* | *Modification reason* | *Modified by* |
| 1.0 | 15/07/2011 | Initial Guide | Hugo Magalhães |
| 1.1 | 04/11/2011 | Added Version Control interface | Ricardo Ferreira |

Table of contents

[History 2](#_Toc308196302)

[Table of contents 3](#_Toc308196303)

[List of figures 4](#_Toc308196304)

[List of abbreviations 5](#_Toc308196305)

[Executive summary 6](#_Toc308196306)

[1 Introduction 7](#_Toc308196307)

[2 Before you start 8](#_Toc308196308)

[2.1 Keystore 8](#_Toc308196309)

[2.1.1 Installing Certificates 8](#_Toc308196310)

[2.2 Apache Configuration 8](#_Toc308196311)

[3 Quick start 9](#_Toc308196312)

[4 Configuring DemoSP-PHP 10](#_Toc308196313)

[4.1 SP 10](#_Toc308196314)

[4.1.1 config/authsources.php 10](#_Toc308196315)

[4.1.2 lib/SAML2/Constants.php 10](#_Toc308196316)

[4.2 PEPS (IdP) 10](#_Toc308196317)

[4.2.1 metadata/saml20-idp-remote.php 10](#_Toc308196318)

[4.3 SAML 11](#_Toc308196319)

[4.3.1 Attributes 11](#_Toc308196320)

[4.3.2 SP Country 11](#_Toc308196321)

[4.3.3 Citizen Countries 11](#_Toc308196322)

[4.3.4 Namespaces 11](#_Toc308196323)

[5 Running DemoSP-PHP 12](#_Toc308196324)

[6 Installing DemoSP-PHP from scratch 14](#_Toc308196325)

[7 SAML Engine API 15](#_Toc308196326)

[7.1 Generating a STORK Authentication Requests 15](#_Toc308196327)

[7.2 Validating and Reading STORK Authentication Responses 15](#_Toc308196328)

List of figures

[Figure 1 – DemoSP-PHP start page 12](#_Toc299012193)

[Figure 2 – DemoSP-PHP return page 12](#_Toc299012194)

List of abbreviations

<Abbreviation> <Explanation>

STORK Secure idenTity acrOss boRders linKed

PEPS Pan European Proxy Server

SP Service Provider

IDP Identity Provider

Executive summary

This document provides detailed information on how to configure, build and deploy a DemoSP PHP application for use in the STORK network.

Since a working Apache server is necessary to deploy the SP application, the document starts by providing some basic configurations of the server.

Afterwards, it describes what the user needs to know about all the potential configurations for the projects, in order to be able to successfully build the application.

After reading this document, the user will be able to configure, build and deploy the application by himself.

# Introduction

This document is divided in several chapters in order to allow the reader to easily access the most relevant section for the scenario on which he is working.

The next chapter shows how to set up your Apache server for using simplesamlphp framework and where to deploy the distributed package.

In the 3th chapter we provide a guide to quick start the DemoSP-PHP.

The 4th chapter describes every configuration needed by the DemoSP-PHP.

On the next chapter we show a typical running session.

On chapter 6 we demonstrate how to install the DemoSP-PHP from scratch.

The final chapter demonstrates how to use the SAML PHP API for generating and validating SAML messages.

# Before you start

You must make sure you have a fully-working apache server.

## Keystore

The SP Application uses two PEM files (private key and public key) to configure the certificate to sign the SAML Request and the certificate to include in the SAML Request.

## Installing Certificates

The DemoSP-PHP’s certificate public key and private key should be placed in the cert/ dir (see section 4.1.1 to configure the path).

## Apache Configuration

1. Edit the apache file that contains the information regarding virtual hosts
2. Add the following to the desired virtual host configuration:

Alias /SP /var/simplesamlphp/www/SP/

1. Save and quit.
2. Restart your apache server.

You may change the alias from /SP to whatever you want. That will become part of the URL when accessing the PHP SP. Throughout this guide we will assume that the alias is /SP.

# Quick start

The following procedure should help you get started in a few seconds using a pre-arranged distribution of simplesamlphp. For a more thorough install please see chapters 4 and 6.

1. Copy the distributed DemoSP-PHP content to /var/simplesamlphp/
2. Open the file DemoSP-PHP/lib/SAML2/Constants.php
   1. Edit the property ASSERTION\_URL to https://**insert.your.ip.here** /SP/return.php
3. Open the file DemoSP-PHP/metadata/saml20-idp-remote.php
   1. Edit the property SingleSignOnService to https://**insert.your.country.access url.to.STORK**

You can now go to your browser and start the application (see chapter 5).

# Configuring DemoSP-PHP

## SP

DemoSP-PHP Project provides configuration files that can be tweaked. In this section we will explain each property.

## config/authsources.php

The config/authsources.php provides the main Service Provider configurations.

By creating multiple ‘identifier’ entries it is possible to create more than one Service Provider per machine.

|  |  |
| --- | --- |
| **Key** | **Description** |
| ‘identifier’ | Identifier used to identify this SP. E.g.: DEMO-SP |
| Name | SP name |
| Certificate | Filename of the SP Certificate |
| Privatekey | Filename of the SP Private Key (PEM Format) |
| privatekey\_pass | SP Private Key pass |
| attributes.NameFormat | Name format used in the attributes |
| sign.authnrequest | Sign AuthnRequest |

The certificates and keys should be placed in the cert/ dir.

For more information please consult: http://simplesamlphp.org/docs/1.8/saml:sp

## lib/SAML2/Constants.php

There are two properties, located in this file, regarding SP.

|  |  |
| --- | --- |
| **Key** | **Description** |
| ASSERTION\_URL | The URL of this SP that will handle the PEPS response. |
| SPID | The ID of the SP in use. |
| SP\_VC\_FILE | The path to the version control file generated by the version control tool. |

## PEPS (IdP)

Since in STORK exists a Pan-European Proxy Service (PEPS) between the SP and IdP, from this point on, each reference to IdP should be understood as SPEPS.

## metadata/saml20-idp-remote.php

Provides information regarding the destination IdP.

By providing multiple ‘metadata’ entries it is possible to specify more than one IdP.

|  |  |
| --- | --- |
| **Key** | **Description** |
| $metadata[‘identifier’] | Identifier used to identify this IdP. E.g.: LOCAL |
| Name | IdP name. |
| SingleSignOnService | URL of the IdP that will handle the request |
| certFingerprint | SHA-1 fingerprint of the IdP certificate |
| sign.authnrequest | Sign AuthnRequest |
| Redirect.validate | Validate signature in redirects |

For more information please consult: http://simplesamlphp.org/docs/1.8/simplesamlphp-reference-idp-remote

## SAML

The file lib/SAML2/Constants.php has various configurations regarding the SAML Message.

## Attributes

The arrays $ids and $attrs hold the various SAML supported attributes. If you want to add a new attribute, just add a new entry in $ids and add the following entries in $attrs:

|  |  |
| --- | --- |
| **Key** | **Description** |
| ‘attr’.name | Name of the new attribute |
| ‘attr’.uri | URI of the new attribute |
| ‘attr’.nameFormat | Name format of the new attribute |
| ‘attr’.value | Default value of the new attribute |

## SP Country

If you want to add new countries for authentication or modify the current one, there’s a property, $spcountries, for that effect. Simply add a new entry to that property. Do not forget, however, to add a metadata entry in metadata/saml20-idp-remote.php (see section 4.2.1). The identifier of the metadata must be the exact same name as the identifier added in the $spcountries property.

## Citizen Countries

If you want to add new citizen countries there’s a property, $countries, for that effect. Simply add a new entry to that property.

## Namespaces

To adjust the namespaces to suite your specific requirements modify the following properties

|  |  |
| --- | --- |
| **Key** | **Description** |
| STORKP\_NS | STORK protocol namespace |
| SAMLP\_NS | SAML protocol namespace |
| STORK\_NS | STORK assertion namespace |

# Running DemoSP-PHP

Start your browser and navigate to the following page: “http(s)://**insert.your.ip.here** / SP/”. You should now be navigating in the DemoSP- PHP. To modify for your own SP refer to the next section.

You should now be looking at page similar to Figure 1.

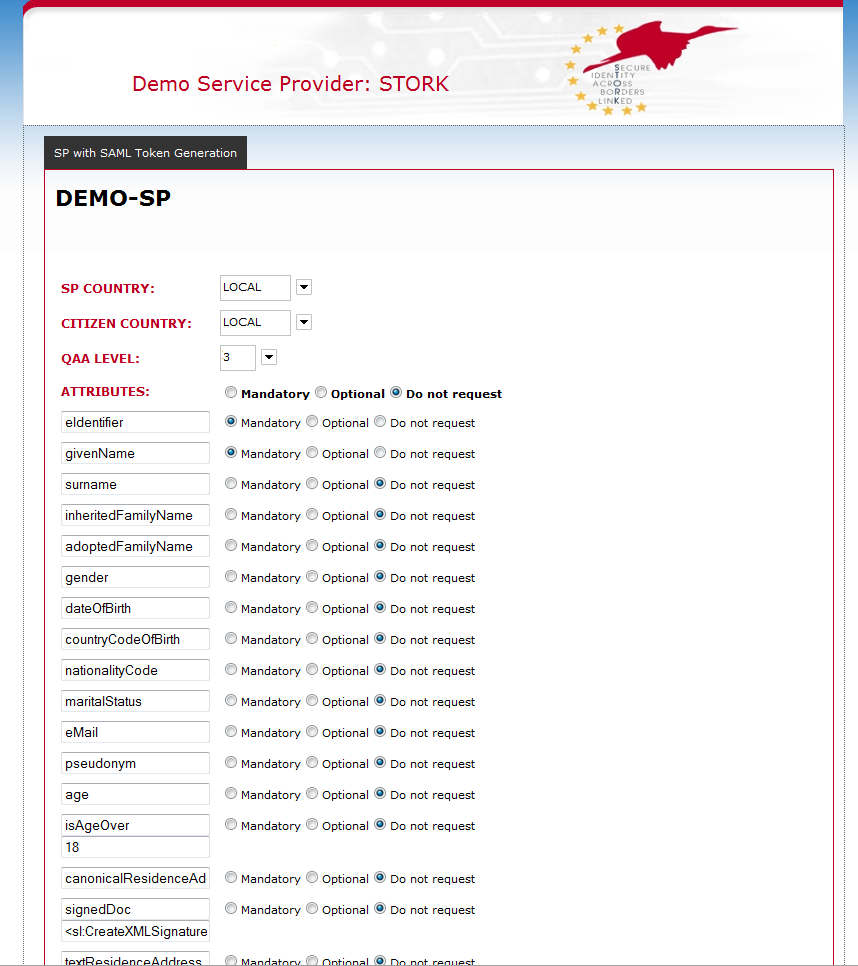


Figure 1 – DemoSP-PHP start page

Upon returning from the S-PEPS you should see a screen similar to Figure. 2.

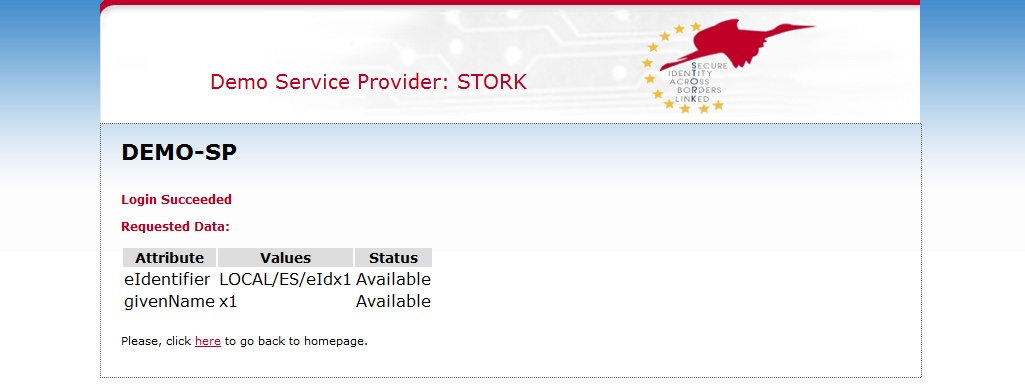


Figure 2 – DemoSP-PHP return page

The SP’s version control file is available here: “http(s)://**insert.your.ip.here** /SP/spInfo.php”.



# Installing DemoSP-PHP from scratch

In this chapter we will demonstrate how to install the DemoSP-PHP from scratch, using the simplesamlphp framework.

1. Download version 1.8.0 from <http://code.google.com/p/simplesamlphp/downloads/list>
2. Extract the downloaded package to /var/simplesamlphp/
3. Open /var/simplesamlphp/lib/SAML2/Assertion.php
   1. In method parseAuthnStatement comment the following lines:

$accr = SAML2\_Utils::xpQuery($ac, './saml\_assertion:AuthnContextClassRef');

if (empty($accr)) {

…

$this->authnContext = trim($accr[0]->textContent);

}

* 1. In method parseAttributes add the following lines:

if($attribute->hasAttribute('stork:AttributeStatus')) {

$this->attributes[$name]['AttributeStatus'] = $attribute->getAttribute('stork:AttributeStatus');

}

after:

if (!array\_key\_exists($name, $this->attributes)) {

$this->attributes[$name] = array();

}

and modify:

$this->attributes[$name][] = trim($value->textContent);

to:

$this->attributes[$name]['AttributeValues'] = trim($value-> textContent);

1. Open /var/simplesamlphp/lib/SAML2/AuthnRequest.php
   1. In method setExtensions the final line should be:

return $root

1. Copy the provided sources DemoSP-PHP-SRC to /var/simplesamlphp

# SAML Engine API

## Generating a STORK Authentication Requests

There are three fundamental properties that must be created before the SAML request, extensions, SP metadata and IdP metadata. Only then can we build the SAML.

//the metadata name of this SP

$authSource = ‘PT-SP’;

//load the SP metadata

$as = SimpleSAML\_Auth\_Source::getById($authSource);

$metadata = $as->getMetadata();

//load the IdP metadata. The ‘country’ parameter is expected to be set by the //HTML form

$idp = $\_POST['country'];

$idpMetadata = $as->getIdPMetadata($idp);

//load extensions

$extensions = StorkConstants::genAttrs($\_POST);

//build the Authentication Request

$ar = stork\_saml\_Message::buildAuthnRequest($extensions, $metadata, $idpMetadata);

After the SAML Request is built we must do a POST to the appropriate IdP.

$b = new SAML2\_StorkHTTPPost($idp);

$b->send($ar);

## Validating and Reading STORK Authentication Responses

Upon receiving a POST request we must validate the SAML signature.

//get response

$b = new SAML2\_HTTPPost();

$response = $b->receive();

//get metadata

$authSource = 'PT-SP';

$as = SimpleSAML\_Auth\_Source::getById($authSource);

$metadata = $as->getMetadata();

//validate signature

$retVal = stork\_saml\_Message::checkSign($metadata, $response);

if($retVal) {

//get assertions

$assertions = $response->getAssertions();

//get attributes

$attributes = $assertions[0]->getAttributes();

//get the saml response status

$status = $response->getStatus();

if('urn:oasis:names:tc:SAML:2.0:status:Success' !== $status['Code']) {

// Authentication succeeded!

} else {

// Authentication Failed!

}

} else {

// Signature validation Failed

echo '<h2>An error occurred';

echo '<p>The signature validation failed.</p>';

}