Connecting a service to the SURFconext Strong Authentication gateway

SURFconext Strong Authentication was released in production on 31July 2015. All documentation on SURFconext Strong Authentication has been moved to https://support.surfconext.nl/strong-authentication. Please visit this wiki for the latest info.

This specific page on how to connect a service to the SURFconext Authentication Gateway has been moved here.

Introduction

To use the SURFconext Strong Authentication for authentication with your service you must use the SAML 2.0 protocol. Your service is a SAML service provider (SP) and the SURFconext Strong Authentication gateway will be your SAML identity provider (IdP). Connecting your SP to the SURFconext Strong Authentication gateway is technically very similar to connecting to SURFconext. In a nutshell:

- Use the Interoperable SAML 2.0 Web Browser SSO Deployment Profile;
- which is a specialization of the SAML V2.0 Web Browser SSO Profile (pdf);

The core standard on which the profiles build is described in Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0 (pdf)

We do not recommend you write your own implementation of the SAML protocol. Instead use one of the many available libraries and products. Feel free to contact us to ask for suggestions.

If you are new to SAML you may want to read the SAML V2.0 Executive Overview (pdf). Because of the strong similarities between SURFconext and the Strong Authentciation gateway most of the technical documentation for SURFconext SPs applies. What is different between the SURFcon ext Strong Authentication gateway and SURFconext are the:

- metadata for the SURFconext Strong Authentication gateway;
- and some **optional** features supported by SURFconext Strong Authentication that are reflected in the SAML protocol messages

When a service provider (SP) is connected to SURFconext, connecting it to use the Strong Authentication gateway should require minimal changes on the part of the SP. The SURFconext Strong Authentication gateway replaces the regular SURFconext gateway.

From the perspective of a SP, authentication is still one SAML authentication request to a SAML IdP (the SURFconext Strong Authentication gateway) which results in one SAML response.

When the SP needs different authentications strengths e.g. low strength for a student accessing their own grades, high strength for staff entering student grades, more work may be involved. In that case the SP needs to communicate the required authentication strength the to SURFconext Strong Authentication gateway, and must verify the actual strength at which a user was authenticated. Both are expressed in the SAML request and response messages that are already being exchanged between the SP and the SURFconext Strong Authentication IdP, so no additional SAML exchanges or exchanges in other protocols are required.

Differences between 'SURFconext' en 'SURFconext Strong Authentication' for SPs

There are a few differences between the SAML implementation on the SURFconext Strong Authentication gateway and the regular SURFconext gateway.

Signing of the SAML Authentication Request

The SAML authentication request for the SP the SURFconext Strong Authentication gateway must be signed. The signature algorithm to be used is `http://www.w3.org/2001/04/xmldsig-more#rsa-sha256`. The key used for signing should be published in the metadata of the SP as X509Certificate. The RSA key used for signing should have a modulus of 2048 or more bits.

No SessionIndex in the AuthnStatement

In SURFconext the `AuthnStatement` in the SAML `Response` may have a `SessionIndex` attribute. The SURFconext Strong Authentication gateway does not provide a `SessionIndex` in the SAML Response that is returned to the SP.

More information on the `SessionIndex` attribute can be found at line 1107 in https://www.oasis-open.org/committees/download.php/35711/sstc-saml-core-errata-2.0-wd-06-diff.pdf

Strong Authentication

For expressing the "strength" of the authentication and the identity of the user a assurance framework as described in NIST Special Publication 800-63-1 and ISO/IEC 29115 is used. The SURFconext Strong Authentication gateway will Support 3 levels of assurance (LoA)

- LoA 1 : Password authentication through SURFconext at the user's home IdP
- LoA 2 : LoA 1 + SMS authentication or Tiqr authentication
- LoA 3 : LoA 1 + Yubikey (hardware token) authentication

Each LoA is assigned a unique identifier. For the pilot the following identifiers are used:

- LoA 1: `http://suaas.example.com/assurance/loa1`
- LoA 2: `http://suaas.example.com/assurance/loa2`
- LoA 3: `http://suaas.example.com/assurance/loa3`

The production gateway will use different identifiers.

These identifiers are used in SAML protocol messages to communicate the LoA between the SURFconext Strong Authentication gateway and a SP.

- The SURFconext Strong Authentication gateway will report the actual LoA at which authentication was performed in a `AuthnContextClassRef` element in a `AuthenticationContext` in the SAML `Assertion` that the SP receives from the SURFconext Strong Authentication gateway after successful authentication.
- A SP may request authentication at a specific LoA by specifying one of the defined LoA identifiers in a `AuthnContextClassRef` element in a `RequestedAuthnContext` in a SAML `AuthnRequest`.

How to determine at which LoA a user must be authenticated?

During discussions with IdPs and SPs several scenarios were identified:

1. An IdP wants to enforce that its users are authenticated at a certain minimum LoA for a specific SP.
2. A SP always wants to have a certain minimum LoA
3. A SP wants to be able to request a higher LoA for a user under control of the SP. E.g. only when the user accesses the "admin" part of the site.

The first two scenarios can be implemented by adding a static policy on the SURFconext Strong Authentication p gateway. Because at the time of the authentication the gateway knows both the IdP and the SP involved it can determine the minimum LoA required for the authentication. The third scenario requires a way for the SP to specify a minimum LoA during authentication.

Requesting authentication at a specific LoA

A SP can request authentication at a specific LoA by specifying the LoA in the `AuthnRequest`. Note that an SP can send an AuthnRequest to the gateway at any time, also when a user is already logged in. This allows an SP to raise the LoA for a user that is using the service depending on the context, for instance the operation performed by the user at the SP.

The requested LoA is interpreted as a minimum LoA. The SURFconext Strong Authentication gateway:

- Will not perform authentication below the requested low
- May perform authentication at a higher LoA level, in wich case the higher level LoA will be expressed in the returned SAML `Assertion`.

The LoA required by the SP is passed to the SURFconext Strong Authentication gateway in an `AuthnContextClassRef` element in a `RequestedAuthnContext` element in the SAML `AuthnRequest`:

RequestedAuthnConext

```
<samlp:RequestedAuthnContext>
    <saml:AuthnContextClassRef>http://suaas.example.com/assurance/loa2</saml:AuthnContextClassRef>
</samlp:RequestedAuthnContext>
```

Example AuthnRequest with a request for authentication at LoA 2

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                    ID="_ace040cdf97c2efba5aa4d973a32318217b9aaae09"
                    Version="2.0"
                    IssueInstant="2014-05-26T06:47:27Z"
                    Destination="https://suaas-gw.aai.surfnet.nl//sso.php"
                    AssertionConsumerServiceURL="http://localhost/simplesaml/module.php/saml/sp/saml2-acs.php
/pieter-local-test-sp"
                    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                    >
    <saml:Issuer>http://localhost/simplesaml/module.php/saml/sp/metadata.php/pieter-local-test-sp</saml:
Issuer>
    <samlp:NameIDPolicy Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
                        AllowCreate="true"
                        />
    <samlp:RequestedAuthnContext>
        <saml:AuthnContextClassRef>http://suaas.example.com/assurance/loa2</saml:AuthnContextClassRef>
    </samlp:RequestedAuthnContext>
</samlp:AuthnRequest>
```

Authentication failure

When a user cancels the authentication at the SURFconext Strong Authentication gateway, the SURFconext Strong Authentication gateway sends a SAML Response back to the SP indicating failure. The reason for the failure is given in the `StatusCode` in the `Response`. When the requested LoA cannot be fulfilled the second level `StatusCode` will be "`urn:oasis:names:tc:SAML:2.0:status:AuthnFailed`".

Example Response when users cancels authentication

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                ID="_Yasz/Kubip05bTwe7hIWOc5As+NxwmEliPJ88nUQ"
                Version="2.0"
                IssueInstant="2015-05-12T12:17:38Z"
                Destination="https://pieter.aai.surfnet.nl/simplesamlphp/module.php/saml/sp/saml2-acs.php
/default-sp"
                InResponseTo="_6d93f735ccfb8d98454999b4016d515834211b0dde"
                >
    <saml:Issuer>https://sa-gw.test.surfconext.nl/authentication/metadata</saml:Issuer>
    <samlp:Status>
        <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Responder">
                    <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:AuthnFailed" />
                </samlp:StatusCode>
    </samlp:Status>
</samlp:Response>
```

When the requested LoA cannot be provided by the SURFconext Strong Authentication gateway, for example because the user is not known at the SURFconext Strong Authentication gateway or the requested LoA exceeds the LoA at which the user can be authenticated, the gateway sends a SAML Response back to the SP indicating failure. The reason for the failure is given in the `StatusCode` in the `Response`. When the requested LoA cannot be fulfilled the second level `StatusCode` will be "`urn:oasis:names:tc:SAML:2.0:status:NoAuthnContext`".

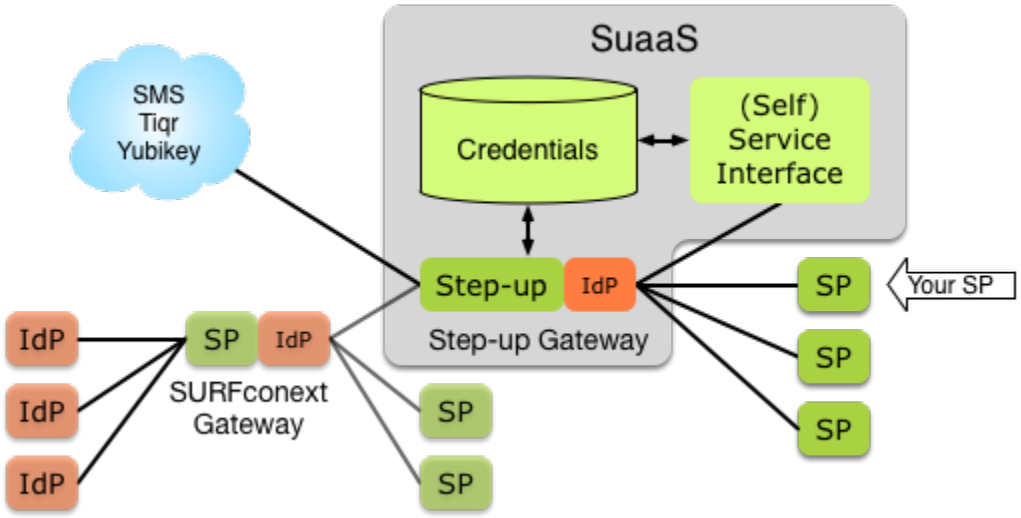
Example Response in case of authentication failure caused requesting unavailable LoA

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                ID="_Yasz/Kubip05bTwe7hIWOc5As+NxwmEliPJ88nUQ"
                Version="2.0"
                IssueInstant="2015-05-12T12:17:38Z"
                Destination="https://pieter.aai.surfnet.nl/simplesamlphp/module.php/saml/sp/saml2-acs.php
/default-sp"
                InResponseTo="_6d93f735ccfb8d98454999b4016d515834211b0dde"
                >
    <saml:Issuer>https://sa-gw.test.surfconext.nl/authentication/metadata</saml:Issuer>
    <samlp:Status>
        <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Requester">
                    <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:NoAuthnContext" />
                </samlp:StatusCode>
    </samlp:Status>
</samlp:Response>
```

SURFconext Strong Authentication gateway architecture

The picture below shows how the SURFconext Strong Authentication gateway, SURFconext, your SP and 2nd factors used for strong authentication (SMS, Tiqr and YubiKey) are related. Notice that:

- For IdPs there are no technical changes required to their IdPs. They still connect to SURFconext
- SPs connect the the SURFconext Strong Authentication gateway. No connection with SURFconext or integration with 2nd factor authentication devices is required.



SURFconext Strong Authentication authentication flow

The picture below shows the authentication flow of a SP using the SURFconext Strong Authentication gateway.

1. The SP sends a SAML 2.0 `AuthnRequest` to the SURFconext Strong Authentication gateway. The SP may use a `RequestedAuthnCo next` to specify the minimal LoA at which a user must be authenticated.
2. The SURFconext Strong Authentication gateway sends a Authn request to SURFconext. SURFconext takes care of the authentication of the user at their home IdP and applies policies: attribute release, user consent and institutional consent.
3. The SURFconext Strong Authenticationgateway receives a response from SURFconext with the identity and attributes of the user.
4. The SURFconext Strong Authenticationgateway determines whether strong authentication is required and, when required, sends the user to the authentication provider for their 2nd factor
5. The response from the 2nf factor authentication provider is returned to the SURFconext Strong Authentication gateway

6. The SURFconext Strong Authentication gateway sends a SAML Response with Assertion and the attributes and the identity of the user to the SP.