

DARIAH AAI Next Generation for Service Provider

FIM4D Workshop
Warsaw



Speakers: Peter Gietz, Martin Haase
DAASI International

Date: 14 May, 2019



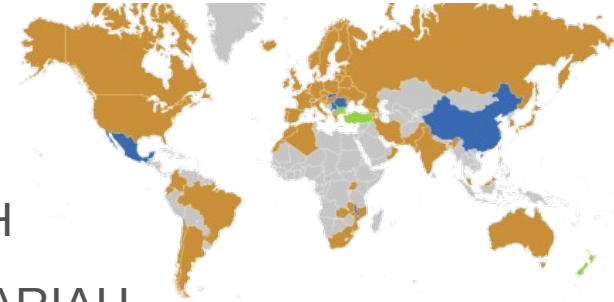
Agenda – Introduction

1. Welcome
2. Federated Identity Management in Research Infrastructures
3. Single Sign-On (SSO) concepts for Web applications - IdPs and SPs
4. Federations and eduGAIN
5. Introduction to SSO using SAML
6. Introduction to Shibboleth as a language-independent SP solution
7. On top of SAML and eduGAIN: benefits of the DARIAH AAI
8. Architecture of the DARIAH AAI IdP Proxy Solution
9. Other SP implementations

Agenda – Hands-On

1. Install the Shibboleth SP on Linux
2. Configuration against the DARIAH AAI Proxy IdP
3. Initiating a session (active and passive protection)
4. Authorization options: application-based, htaccess, XML-based
5. Interaction with applications in various programming languages
6. Using DARIAH AAI attributes for authorization
7. Management of authorization groups using the DARIAH SelfService

FIM4D



- FIM4D - Federated Identity Management for DARIAH
- Authentication and Authorization Infrastructure for DARIAH
- Authenticate to DARIAH services with the password of a researcher's home institution
- Leveraging eduGAIN
 - Almost all European national higher education federations,
 - Plus a lot more national federations (Algeria, Australia, Brazil, India, U.S., ...)
- FIM4D working group aims at
 - disseminating information about this infrastructure,
 - setting up a number of pilots to integrate services from different DARIAH countries into this infrastructure,
 - tackling challenges like IdP integration, Attribute handling, Level-of-trust handling, and interoperability with other AAI infrastructures.

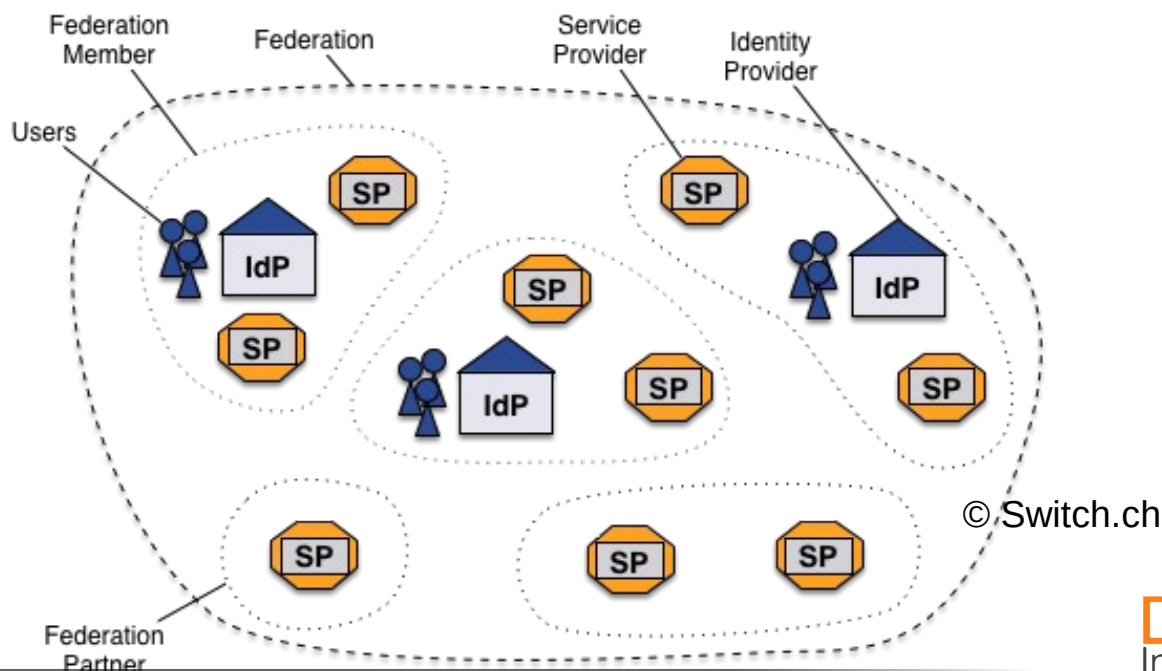
Single Sign-On Concepts

Components

- Identity Provider (IdP)
 - Central component for authentication (Log-In)
 - Provides user attributes (Attribute Authority, AA)
- Service Provider (SP)
 - Protects Web resources
 - Provides applications with user attributes
- Discovery Service (DS)
 - Lets users choose their IdP

Federations

- Federations as collection of organizations
- Federation operator as trusted third party
- Scalable way to allow SSO across organizational boundaries
- SAML Metadata to connect entities

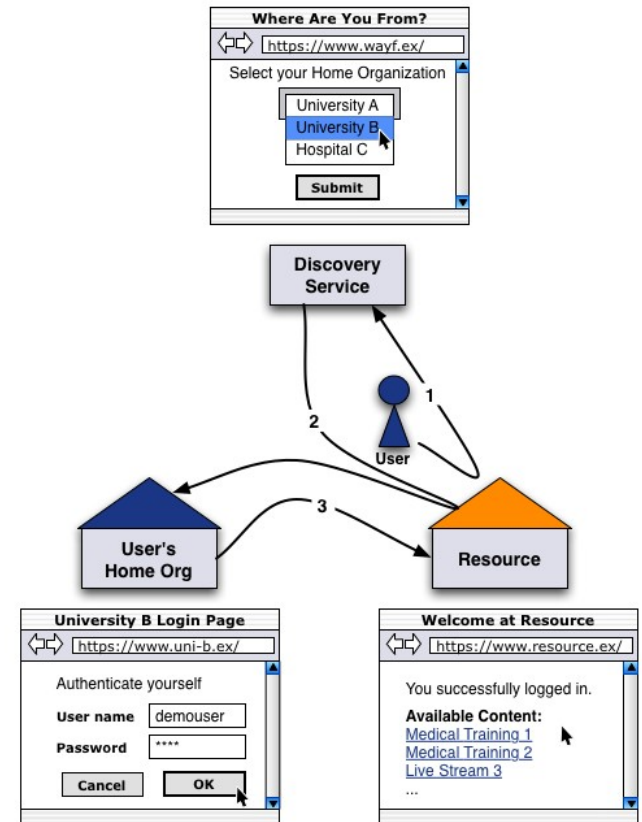


Terms

- Federation
 - Crosses organizational boundaries
 - Union of many SPs and IdPs
 - Central provider of mutual trust SPs ↔ IdPs
- Single Log-On
 - Use the same credentials for different services
- Single Sign-On (SSO)
 - Only one log-on event per session, or day

Short introduction to Web SSO

- Most important use case: Web Browser SSO
- Identity Provider (IdP) authenticates users and issues assertions
- Service Provider (SP) relies on assertions from IdP and uses this information to control access to protected services
- Discovery service (DS) lets users choose their home organization (and therefore IdP)

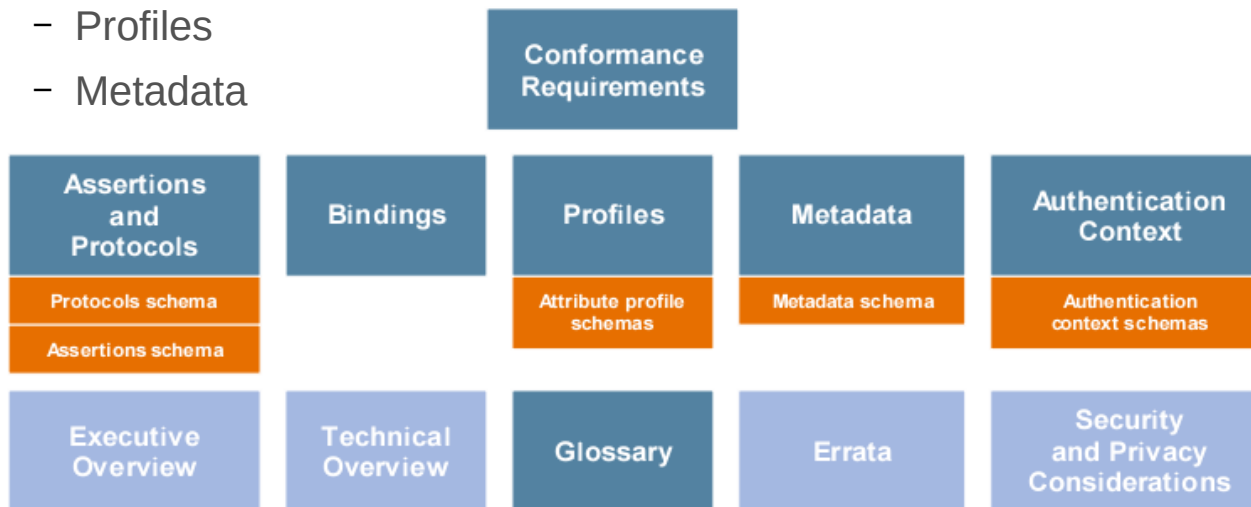


© Switch.ch

SSO using SAML

SAML Basics

- Security Assertion Markup Language (SAML)
- Exchange of information between SAML Entities
 - Asserting Party (IdP)
 - Relying Party (SP)
- OASIS standard, current version 2.0, March 2005
- Main components:
 - SAML Core: Assertions und Protocols
 - Bindings
 - Profiles
 - Metadata



SAML-docset

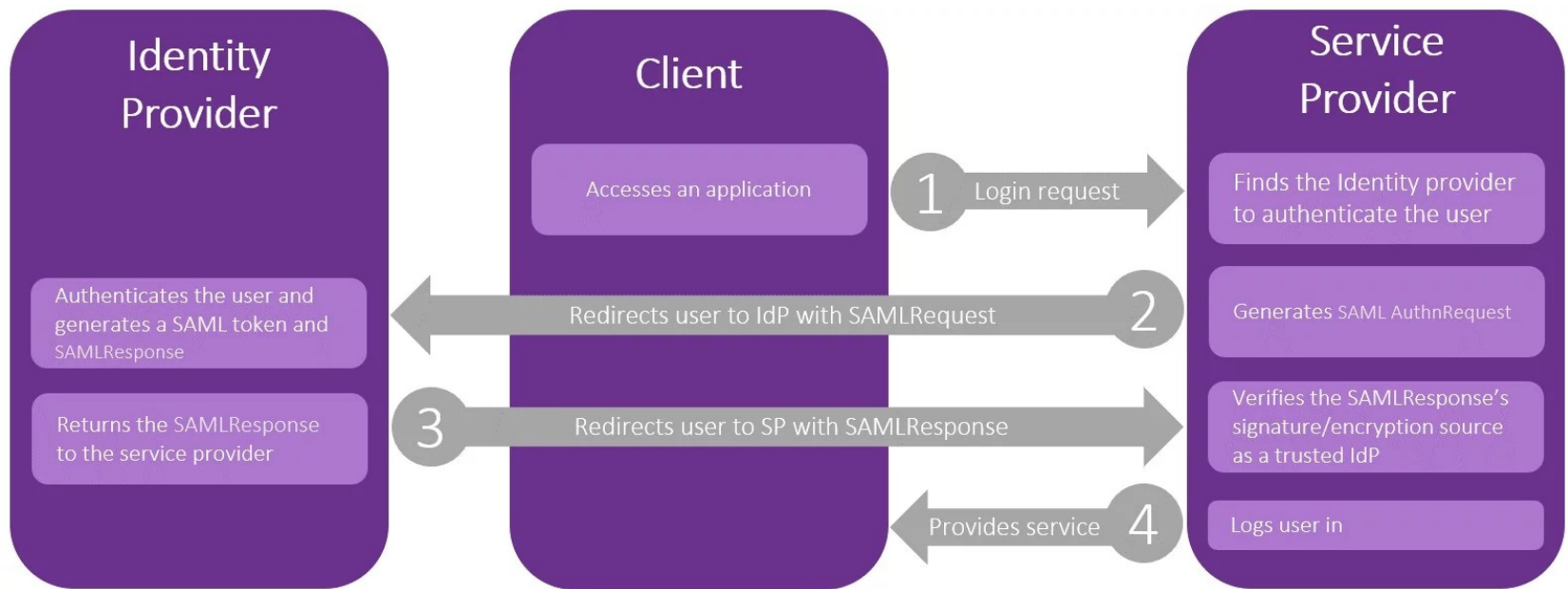
SAML Glossary

- Components: IdP, SP, Discovery Service
- Terms: Federation, SSO
- SAML Entity ID: ID of an IdP or SP (looks like a URL often)
- SAML Assertion: information about a subject, with details about the authentication event, user attributes, ...
- Attribute Authority: an IdP endpoint that releases attributes
- Session: „Security context“, both the IdP and the SP keep a session for the user

„Top down“: SAML Profiles

- Description of use cases
 - Definition of SAML messages (→ Assertions und Protocols)
 - Definition of the transport mechanism (→ Bindings)
- SSO Profiles
 - Web Browser SSO Profile
 - Enhanced Client or Proxy (ECP) Profile
- Further Profiles
 - IdP Discovery
 - Single Logout
 - Artifact Resolution
 - Attribute Query
 - Etc.

Web Browser SSO Profile

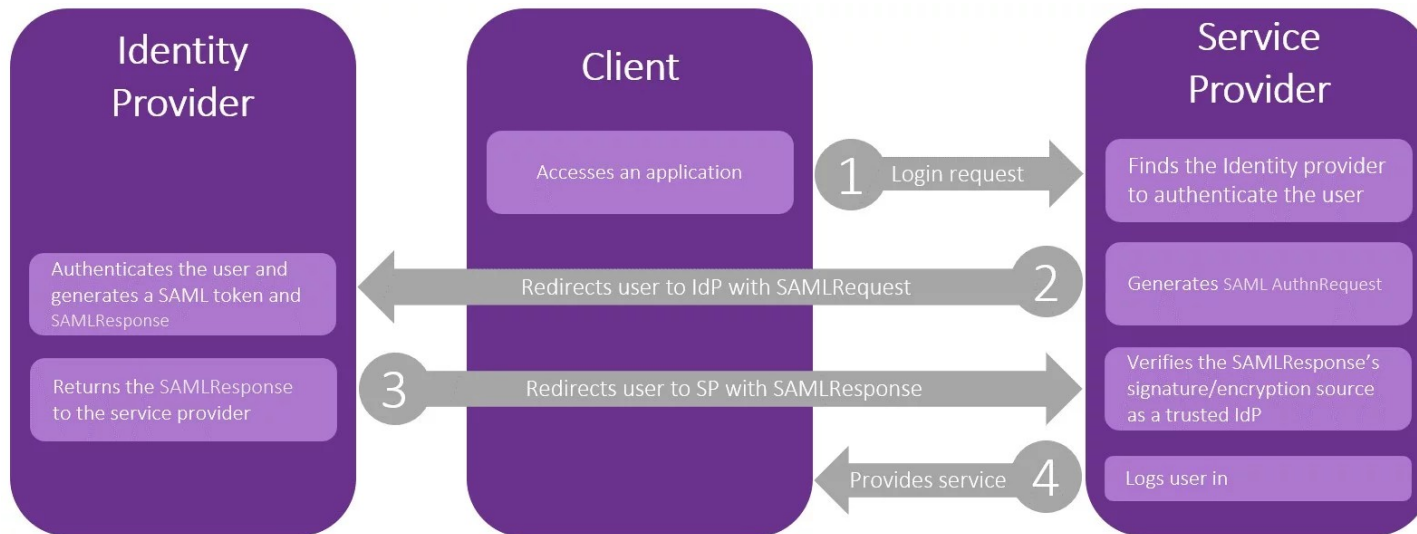


© cyberark.com

Protocols

- Specify XML Schema for SAML protocol messages
- Abstract form
 - RequestAbstractType
 - ResponseAbstractType
- Concrete Examples
 - <AuthnRequest> (SP → IdP)
 - <Response> (IdP → SP)
 - <LogoutRequest> (SP → IdP, IdP → SP)
- Specify validation of SAML messages

<AuthnRequest>



```
<samlp:AuthnRequest
AssertionConsumerServiceURL="https://sp.org/Shibboleth.sso/SAML2/POST"
Destination="https://idp.org/idp/profile/SAML2/Redirect/SSO"
ID="_926bb3c39ce15feb5f07595badd2e17a"
IssueInstant="2017-11-27T14:39:48Z"
ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Version="2.0" xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">
  <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
    https://sp.org/shibboleth
  </saml:Issuer>
  <samlp:NameIDPolicy AllowCreate="1"/>
</samlp:AuthnRequest>
```

<Response>

```
<saml2p:Response Destination="https://sp.org/Shibboleth.sso/SAML2/POST"
ID="_e799fb247f68025b1971a1ab6fa7a5c7"
InResponseTo="_926bb3c39ce15feb5f07595badd2e17a"
IssueInstant="2017-11-27T14:48:52.792Z" Version="2.0"
xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol">
  <saml2:Issuer xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
    https://idp.com/idp/shibboleth
  </saml2:Issuer>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    ...
  </ds:Signature>
  <saml2p:Status>...</saml2p:Status>
  <saml2:Assertion xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
    ...
  </saml2:Assertion>
</saml2p:Response>
```

<Assertion>

```
<Assertion>
  <Issuer>...</Issuer>
  <Subject>
    <NameID>UserIdentifier</NameID>
    ...
  </Subject>
  <Conditions>...</Conditions>
  <AuthnStatement>
    <AuthnContext>
      <AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
      </AuthnContextClassRef>
    </AuthnContext>
  </AuthnStatement>
  <AttributeStatement>
    <Attribute Name="uid">
      <AttributeValue>test</AttributeValue>
    </Attribute>
  </AttributeStatement>
</Assertion>
```

- Usually an <EncryptedAssertion> → XML Encryption

Bindings

- Specify how SAML protocol messages are bound to common transport protocols
- HTTP Redirect Binding
- HTTP POST Binding (→ „Auto Submit Form“)
- HTTP Artifact Binding
- SAML SOAP Binding
- Reverse SOAP Binding
- SAML URI Binding

E.g. HTTP Redirect Binding

HTTP/1.1 302 Object Moved

Date: 26 Nov 2017 16:00:49 GMT

Location:

[https://idp.org/SAML2/SSO/Redirect?](https://idp.org/SAML2/SSO/Redirect?SAMLRequest=fZFfa8IwFMXfBb9DyXvaJtZ1BqsURRC2Mabbw95i)

[SAMLRequest=fZFfa8IwFMXfBb9DyXvaJtZ1BqsURRC2Mabbw95i](https://idp.org/SAML2/SSO/Redirect?SAMLRequest=fZFfa8IwFMXfBb9DyXvaJtZ1BqsURRC2Mabbw95i)
[vc5Am3TJrXPffmmLY3%2FA15Pzuyf33On8XJXBCaxTRmeEhTEJQB](https://idp.org/SAML2/SSO/Redirect?SAMLRequest=fZFfa8IwFMXfBb9DyXvaJtZ1BqsURRC2Mabbw95i)
[dmr\[...\]](https://idp.org/SAML2/SSO/Redirect?SAMLRequest=fZFfa8IwFMXfBb9DyXvaJtZ1BqsURRC2Mabbw95i)

- SAMLRequest=URLEncode(base64(DEFLATE(x)))
- x: SAML protocol message
 - E.g. an <AuthnRequest>

Security Mechanisms in SAML

- SAML Metadata as a base for SP ↔ IdP trust, defined by the SAML-Standard
- SPs and IdPs need information about each other
 - Description of the communication partner's capabilities
 - Endpoint locations (URLs etc.)
 - Inline PKI certificate for XML signature and encryption
- Use OIDs (from LDAP) for unambiguous attribute name encoding, e.g. givenName *urn:oid:2.5.4.42*
- Server clocks must be synchronized, use of TLS, etc.

Introduction to Shibboleth

Shibboleth Project

*„Shibboleth is a **standards based, open source** software package for web single sign-on across or within organizational boundaries.“*

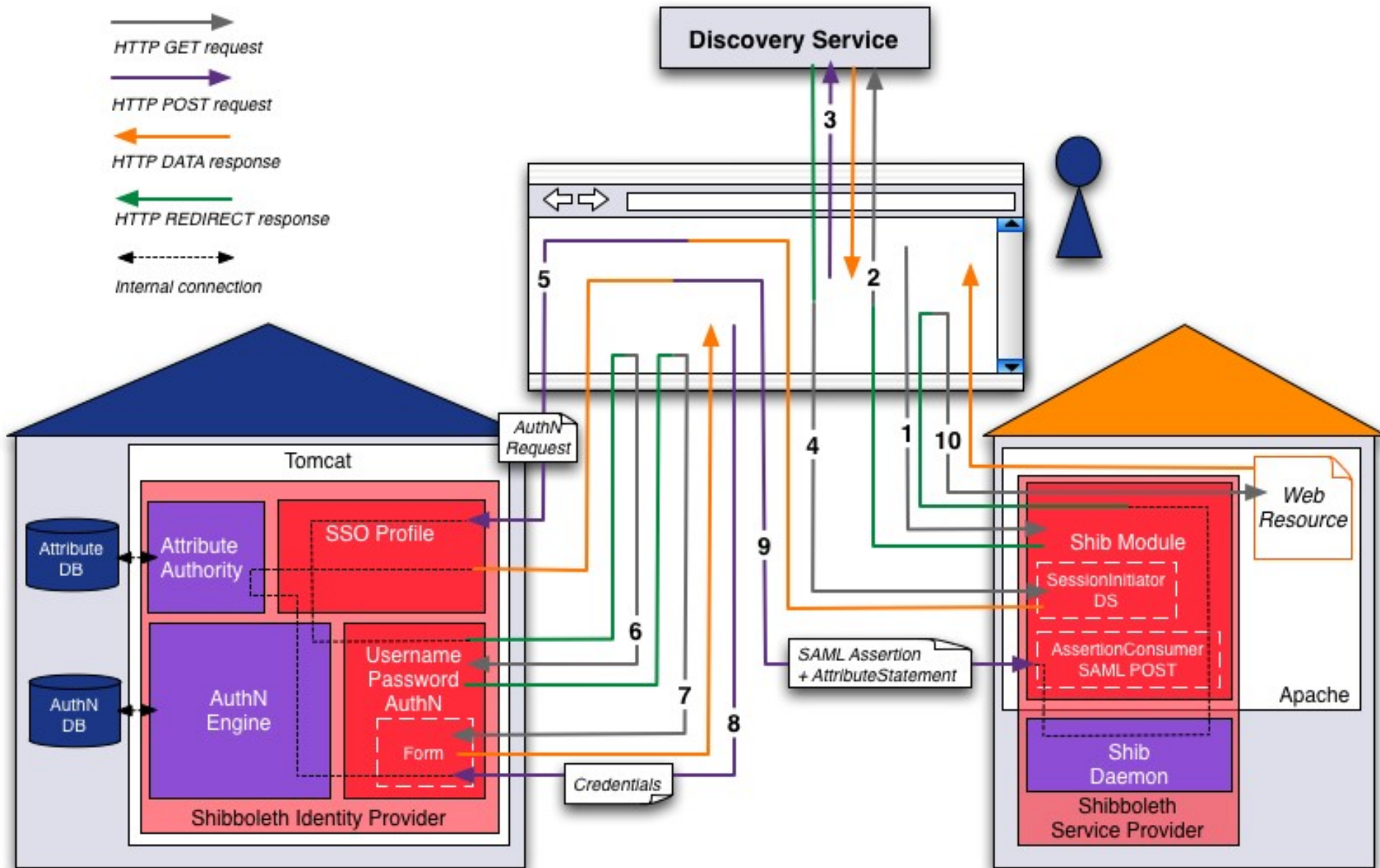


- Began as an Internet2 activity in 2000
- SAML (2.0)
- Maintained by the Shibboleth Consortium
- Widespread useage in research & education
- Origin of the word: see the Bible, Judges 12,5.6

Shibboleth Project

- Identity Provider
- Service Provider
- OpenSAML Libraries (C++ & Java)
- Metadata Aggregator
- Centralized Discovery Service (discontinued, use SWITCH WAYF as an alternative)
- Embedded Discovery Service

WebSSO in Detail



Shibboleth IdP

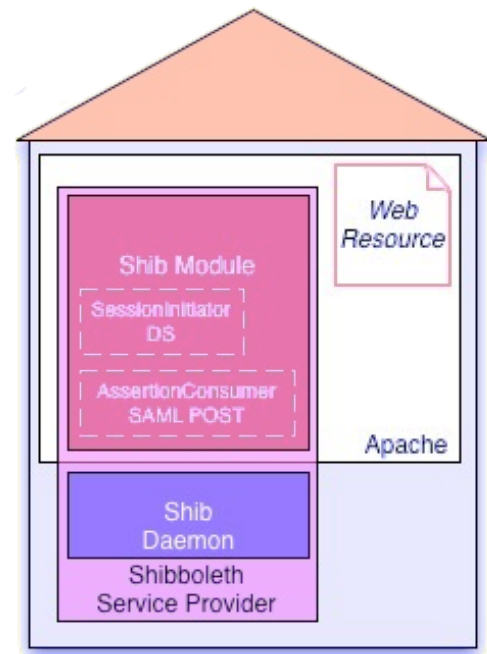
- Implements the Identity Provider (IdP) Component
- AuthN of users from an organisation happens centrally at its IdP
- Assertions (SAML) to convey identity informations (AuthN result + attributes) to Relying Parties (RP)
- Shibboleth IdP is a Java application
- Protocol Support includes SAML 1.1, most SAML 2.0 profiles, CAS
- Based on Spring for configuration & wiring of components
- Current release: V3.4.x

Shibboleth IdP

- Highly configurable
 - Authentication sources
 - Attribute Resolution
 - Attribute Filtering
 - Metadata Management
 - Per - Relying Party (i.e. per-SP) Configuration
 - Policies
 - Backend / Storage

Shibboleth SP

- Implements the SP component
- There are two parts
 - C++ daemon (shibd)
 - Keeps state, evaluates protocol messages
 - Webserver module (mod_shib)
 - Protects Locations/Directories, defines Access Rules
- Current release: V3.0.x
- Binaries for Windows, OSX, RPM-based Linux
- SWITCH Repository for Ubuntu / Debian (current v3 and v2 backports)



(c) Switch

Shibboleth SP Overview

- General configuration in /etc/shibboleth/shibboleth2.xml

```
<ApplicationDefaults entityID="https://sp.org/shibboleth"
    REMOTE_USER="eppn persistent-id targeted-id">
```

- Applications as „containers“ to group common configuration

```
<Sessions lifetime="28800" timeout="3600"
    relayState="ss:mem"
    checkAddress="false" handlerSSL="true"
    cookieProps="https">
```

- Configuration of <Session> properties (e.g. duration)

```
<SSO discoveryProtocol="SAMLDS"
    discoveryURL="https://wayf.org/ds">
    SAML2 SAML1
</SSO>
```

- <SessionInitiator> define how the SP should start a session (shorthands for e.g. <SSO> and <Logout> exist)

```
<Handler type="MetadataGenerator"
    Location="/Metadata" signing="false"/>
```

- <Handler> provide various endpoints

```
<Handler type="Status" Location="/Status"
    acl="127.0.0.1 ::1"/>
```

```
<Handler type="Session" Location="/Session"
    showAttributeValues="false"/>
```

Shibboleth SP Overview

- General configuration in `/etc/shibboleth/shibboleth2.xml`

- Load **metadata** (static or dynamic)

```
<MetadataProvider type="XML" validate="true"
  uri="http://example.org/fed-metadata.xml"
  backingFilePath="fed-metadata.xml"
  reloadInterval="7200">
```

- Configure how the SP should provide **attributes** to applications

```
  <MetadataFilter type="Signature"
    certificate="md_federation.pem"/>
</MetadataProvider>
```

- **Keys and Certificates**

```
<AttributeExtractor type="XML" validate="true"
  reloadChanges="false"
  path="attribute-map.xml"/>
```

- Possible **ApplicationOverrides** to run multiple logical SPs with one installation

```
<AttributeFilter type="XML" validate="true"
  path="attribute-policy.xml"/>
```

```
<CredentialResolver type="File"
  key="sp-key.pem"
  certificate="sp-cert.pem"/>
```

```
<!--<ApplicationOverride /> -->
</ApplicationDefaults>
```

Processing Attributes

- /etc/shibboleth/attribute-map.xml to extract attributes and provide them internally

- Simple mapping from incoming name to internal id

```
<Attribute name="urn:oid:0.9.2342.19200300.100.1.3" id="mail"/>
```

- You can then use them as apache environment variables in protected applications, e.g. `$_SERVER['mail']`
- /etc/shibboleth/attribute-policy.xml to define processing policies
- Default configuration includes e.g. scope checking of eduPersonPrincipalName, eduPersonScopedAffiliation, etc.

Filter Attributes

- /etc/shibboleth/attribute-policy.xml
- <https://wiki.shibboleth.net/confluence/display/SP3/AttributeFilter>

```
<afp:AttributeRule attributeID="sn">
```

```
  <afp:PermitValueRule  
    xsi:type="AttributeIssuerString"  
    value="https://testidp.example.org/idp/shibboleth"/>
```

```
</afp:AttributeRule>
```

```
<afp:AttributeRule attributeID="entitlement">
```

```
  <afp:PermitValueRule  
    xsi:type="AttributeValueString"  
    value="urn:mace:dir:entitlement:common-lib-terms" />
```

```
</afp:AttributeRule>
```

Process Attributes

- Attributes can be changed various times on their way to the application
 - IdP resolves attributes
 - IdP filters attributes, usually on a per-SP basis
 - SP accepts or rejects attributes...
 - ...filters them according to policies...
 - ...and maps them to internal variables
 - Webserver provides variables as environment variables
- Be aware that names might change and attributes might get lost

Protecting Applications

- Three common options
 - Directly in the web server via Apache Access Rules
 - In shibboleth2.xml configuration via SPXMLAccessControl
 - Use application („lazy session“ or „passive protection“)

```
<Location /application>  
  AuthType shibboleth  
  ShibRequestSetting requireSession 1  
  require shib-session  
</Location>
```

```
<Location /lazy>  
  AuthType shibboleth  
  ShibRequestSetting requireSession 0  
  require shibboleth  
</Location>
```

Protecting Applications

- Usually Apache Access Rules is the easiest approach
- Require shib-attr allows to decide based on attributes

```
<Location /application>  
  AuthType shibboleth  
  ShibRequestSetting requireSession 1  
  require shib-session  
  require shib-attr affiliation student  
</Location>
```

Troubleshooting – SP Handlers

- `https://sp.example.org/Shibboleth.sso/Session`
switch on showing attribute values:

```
<Handler type="Session" Location="/Session"  
        showAttributeValues="true"/>
```

- `.../Metadata`
- `.../Status` – permit own IP address

```
<Handler type="Status" Location="/Status"  
acl="127.0.0.1 ::1 11.22.33.44"/>
```

Further SP Handlers

- `.../DiscoFeed` (JSON feed that contains all IdPs known to the SP, to be used in IdP discovery)
- `.../Login?param1=val1¶m2=val2&...`
 - z.b. `target`, `entityID`, `forcedAuthn`
 - <https://wiki.shibboleth.net/confluence/display/SP3/SessionInitiator>
- `.../Logout?return=http://google.de`

Troubleshooting

- Firefox Plugin SAML Tracer
- Log Files
 - /var/log/shibboleth/shibd.log
 - /var/log/shibboleth/transaction.log (shows attributes)
 - /var/log/shibboleth/httpd/native.log (Apache's mod_shib Log)
 - /var/log/httpd/ssl_access_log und ssl_error_log
- Place a phpinfo() under /secure/index.php to see all variables, or /Shibboleth.sso/Session

Shibboleth SP Single Logout

- Complete implementation of the SAML standard
- Default: [https://sp.example.org/Shibboleth.sso/Logout?
return=https://sp.example.org/myApp/
unauthenticated.php](https://sp.example.org/Shibboleth.sso/Logout?return=https://sp.example.org/myApp/unauthenticated.php)
- Konfiguration Options:
 - Trigger no Logout or only Local Logout
 - Trigger Asynchronous Logout (default): Do local logout and pass control to the IdP
 - Synchronous Logout: after IdP Logout, return control to SP
 - Back Channel Logout: use SOAP SLO endpoint of IdP (also Synchronous)
- Notify application using front channel or back channel

Shibboleth DS

- Two pieces still missing
- SP uses Discovery Service (DS) to allow users to choose their IdP

```
<SSO discoveryProtocol="SAMLDS"  
    discoveryURL=https://example.org/ds">  
    SAML2 SAML1  
</SSO>
```

- Shibboleth *did* provide Centralized Discovery Service (CDS)
 - Central as in „not on every SP“
 - The federation operator might run this
 - Shibboleth CDS is no longer maintained
- Shibboleth does provide Embedded Discovery Service (EDS)
 - Embedded as in „runs on every SP host“
 - Can be combined with Metadata management of Shibboleth SP

Shibbolizing Applications

Apache Access Control

- Apache Access control in httpd.conf:

```
<Location /secure>  
    AuthType shibboleth  
    ShibRequestSetting requireSession 1  
    Require shib-attr entitlement common-lib-terms  
</Location>
```
- Supports Regexp
- .htaccess files: only works for "real" directories
- Apache >=v2.4: Combine using boolean rules
- Apache < 2.4: use ShibRequireAll if needed
- <https://wiki.shibboleth.net/confluence/display/SP3/htaccess>

XML Access Control

- /etc/shibboleth/shibboleth2.xml (the only option with IIS):

```
<Path name="secure" authType="shibboleth"
  requireSession="true">
  <AccessControl>
    <Rule require="affiliation">
      member@example.org
    </Rule>
  </AccessControl></Path>
```

- Requires shibd restart (watch XML errors!)
- Only way to evaluate query parameters
- <https://wiki.shibboleth.net/confluence/display/SP3/XMLAccessControl>

Application Access Control

- Need at least lazy Session on the URL
- Can make use of arbitrary attributes
- E.g. PHP:

```
if ($_SERVER['affiliation'] == 'staff')  
    { grantAccess(); }
```
- E.g. in a Perl CGI Script:

```
if ($ENV{'affiliation'} eq 'staff')  
    { grant_access() }
```
- E.g. Java Servlet in Tomcat with Apache Proxy:

```
affiliation = (String)  
    request.getAttribute("affiliation");  
if (affiliation.equals("staff"))  
    grantAccess();
```

Lazy Session

- Application must implement access control
- Need passive protection:

```
<Location /lazy>  
    AuthType shibboleth  
    Require shibboleth  
</Location>
```

- Application must trigger login:

```
https://sp.example.org/Shibboleth.sso/Login?  
target=https://sp.example.org/cgi-bin/  
application.php&  
entityID=https://idp.example.org/
```

Lazy Session: Option 2

- No need to call /Shibboleth.sso/Login directly
- Put the login URL under active protection and redirect there, when user wants to log in

```
<Location /login>
```

```
AuthType shibboleth
```

```
ShibRequestSetting requireSession 1
```

```
require valid-user
```

```
</Location>
```

- Use this part of the application to evaluate attributes and store them to the session.

isPassive at the SP

- Do I have a session at the IdP? The SP cannot know without trying!
- Pass the Discovery Service (user must have her Cookie for the chosen IdP there)
- SAML request passes the IdP and returns to the SP silently:
 - Existing IdP session: with a SAML Assertion
 - No IdP session: with a SAML "error" response *noPassive*. **User does not see a login form!**
- SP knows how to handle *noPassive* and would show a default page
- isPassive at Shibboleth IdP depends on Authentication Mechanism, e.g. RemoteUser does not support it

forceAuthn

- Force Re-Authentication despite of SSO
- Can be used for critical (parts of) applications
- Either set in the <SSO> handler for forcing generally,
- Or force case-wise: evaluate Authn Timestamp and call...

/Shibboleth.sso/Login?forceAuthn=true

DARIAH AAI

DAASI
International



DARIAH AAI - challenges

- Goal 1: users of DARIAH services (SPs) should authenticate via their home organization (campus IdP).
- Goal 2: certain DARIAH services only allow particular user groups. This should be configurable centrally by the respective admins, for all DARIAH services.
- Goal 3: DARIAH needs some user information
 - 3.a) she agrees to DARIAH Terms
 - 3.b) she is a researcher (e.g. her organization or e-mail)
- Goal 4: cope with a situation where users either
 - 4.a) have no campus IdP
 - 4.b) their campus IdP would not release Personally Identifiable Information (PII) to hitherto unknown SPs

DARIAH AAI - Self Service

DARIAH Self Service

https://auth.dariah.eu/cgi-bin/selfservice/ldportal.pl?mode=selfreg&isreset=1&newtoken=70235548: Suchen

DARIAH Self Service



DARIAH Self Service / Account Request / Help texts

Start

Forgot Password

Account Registration

Request a DARIAH Account

Please provide details about yourself. Mandatory fields are marked with an asterisk.

Given Name *

Surname *

E-Mail *

Please enter your institutional e-mail address and *no private or commercial account* if possible.

Organisation *

Initial group

Terms of Use * ☐ I accept the Terms of Use for DARIAH-DE.

Captcha *

DARIAH AAI - Register Federation Users

DARIAH Self Service

https://auth.dariah.eu/cgi-bin/selfservice/ldapportal.pl?mode=authenticate;shibboleth=1;ne: Suchen

DARIAH Self Service

Logged in as MartinHaase

DARIAH Self Service / Registration / Help texts

Start

My User Data

Request New Service

Log out

Registration for DARIAH Services

Your Home Organization has sent DARIAH the following Data about you. Please complete at least the mandatory fields marked with an asterisk.

Home Organisation ID *

External Login *

Given Name *

Surname *

E-Mail *

Organisation *

Department

DARIAH AAI - Self Service

DARIAH Self Service

https://auth.dariah.eu/cgi-bin/selfservice/ldapportal.pl?mode=modifyentry&isreset=1&newtoken=867€

Suchen

DARIAH Self Service

Logged in as MartinHaase

DARIAH Self Service / User Data /

Help texts

Start

My User Data

Request New Service

Log out

Your Data in DARIAH

The following data is stored in the DARIAH directory.

Login * MartinHaase

Given Name * Martin

Surname * Haase

Full Name Martin Haase

E-Mail * martin.haase@daasi.de

Please write to register@dariah.eu in order to change your email address.

Organisation * DAASI International GmbH

Please write to register@dariah.eu in order to change your organisation.

DARIAH AAI - Administration: AuthZ Groups

DARIAH-DE Administration

https://auth.dariah.eu/cgi-bin/adminnew/ldapportal.pl?mode=grouplist&isreset=1&newtoken

Suchen

DARIAH-DE Administration

Logged in as MartinHaase, National Representative - DE

Organisations / Group Management /

Help texts

Home Page

Search

Complex Search

New Account

Groups

New Organisation

New Country

LDAP Statistics

Organisations

Account Requests

Group Management

You can edit the following groups:

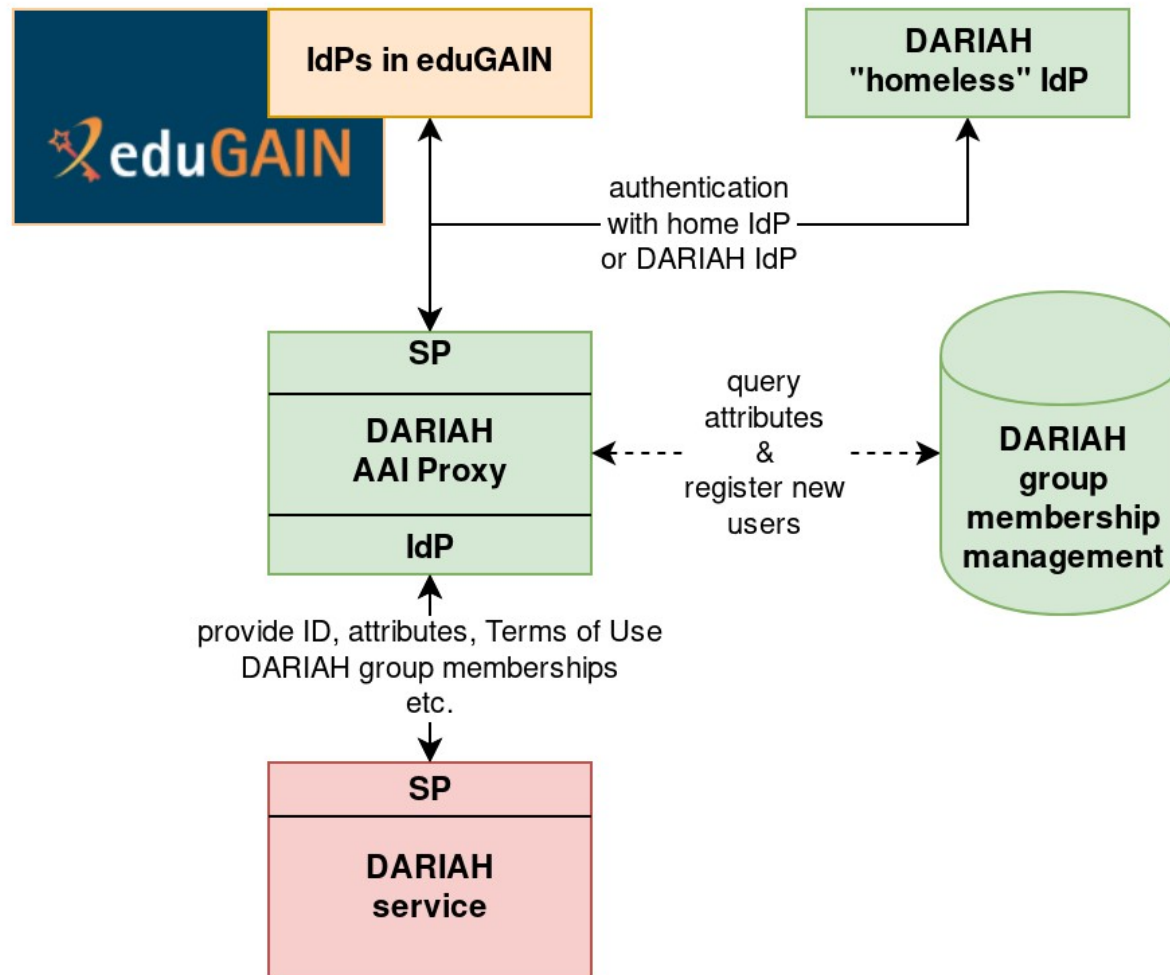
textgrid-contrib

Group name	Owner		
textgrid-contributors	groupowner	Modify	Delete
tf-dariah-textgrid-contributors	groupowner	Modify	Delete

Showing 1 to 2 of 2 rows

+ new group

DARIAH AAI NG - Overview



DARIAH AAI NG (1)

- In the past, it was difficult to connect services to the DARIAH AAI.
- In mid 2018, DARIAH introduced a central AAI proxy that brokers between DARIAH services and eduGAIN.
- The proxy took over many tasks from the services, which makes it much easier to connect new services.

DARIAH AAI NG (2)

- Almost any SAML SP library can be used in an application
- No registration of the SP in a federation needed anymore - just exchange SAML metadata with the proxy
- The proxy ensures Identity Provider (IdP) Discovery and the connection to eduGAIN
- It provides a service with all IdP attributes, plus information from the central DARIAH directory

DARIAH AAI NG (3)

- The DARIAH AAI Proxy is based on the open source Shibboleth IdP and SP software. It implements the Blueprint Architecture of the Horizon 2020 AARC project and consists of
 - an IdP component that is connected to DARIAH Services, and of
 - an SP component that is connected to all IdPs in eduGAIN.
- These two components are strung together, such that the AAI proxy forwards all data from eduGAIN IdPs on to DARIAH SPs.

DARIAH AAI NG (4)

- Besides just forwarding IdP attributes, the DARIAH AAI proxy does a bit more to ensure DARIAH needs:
 - Check if an eduGAIN user is registered in the DARIAH directory, and send her to DARIAH SelfService
 - Check the DARIAH and possible Service-specific Terms of Use have been accepted
 - Enrich IdP attributes with central authorization group information to be used for access control in services

DARIAH AAI - SelfService

- Register users that authenticate via the eduGAIN meta-federation
- Let users agree to the general DARIAH and specific service Terms of Use
- Manage user's data in DARIAH
- Manage central authorization groups which services can use for access control
- For users that do not have an eduGAINIdP: apply for DARIAH accounts and manage passwords

Other SAML SP Implementations

Shibboleth SP vs. Libraries

- Shibboleth SP is (albeit written in C++) language-independent; Communication with application via
 - Handler URLs
 - Apache Environment Variables (optionally HTTP Headers)
- Can be used with Apache and IIS
- There exist many language-specific SAML libraries for common programming languages

Recommended Open Source Libraries

- simpleSAMLphp for PHP (can be used also as an IdP, or as an IdP-SP-Proxy)
- pySAML2 for Python (the "2" is important)
- Spring Security SAML for Java+Spring (based on OpenSAML-Java, like the Shibboleth IdP)
- *Actually, since the AAI proxy is there, any well-maintained SAML implementation that can deal with encrypted SAML assertions will do.*

Hands-On

SP Installation

- Installation instructions for all operating systems:
- <https://www.switch.ch/aai/guides/sp/installation/>
- After installation, cd to /etc/shibboleth/
- edit shibboleth2.xml: Set <ApplicationDefaults entityID="https://**your.hostname.eu**/shibboleth" ...
- Make sure sp-{*signing|encryption*}-{*cert|key*}.pem exist
 - create these two self-signed certificate pairs using the *shib-keygen* command
 - it is possible to use just one key pair for both purposes
- restart shibd: `systemctl restart shibd`

Connect an SP to the DARIAH AAI

Set up Trust with the AAI Proxy

- Send your own SP's metadata to register@dariah.eu (from <https://your.hostname.eu/Shibboleth.sso/Metadata>)
- Download the DARIAH AAI Proxy's Metadata file like this (mind the capital "-O", and all on one line)

```
wget https://aaiproxy.de.dariah.eu/idp -O  
/etc/shibboleth/dariah-proxy-idp.xml
```

- Locate the `<MetadataProvider>` section in `/etc/shibboleth/shibboleth2.xml` and add a row:

```
<MetadataProvider type="XML" path="dariah-  
proxy-idp.xml"/>
```

Further SP Configuration

- In `/etc/shibboleth/shibboleth2.xml`, set the SP to direct login using
`<SSO entityID="https://aaiproxy.de.dariah.eu/idp">`

- Use the following ID preference order in `<ApplicationDefaults ...>`
`REMOTE_USER="eppn unique-id"`

- Enable `eduPersonUniqueID` in `/etc/shibboleth/attribute-map.xml`:
`<Attribute name="urn:oid:1.3.6.1.4.1.5923.1.1.1.13"`
`id="unique-id">`
`<AttributeDecoder`
`xsi:type="ScopedAttributeDecoder"/>`
`</Attribute>`

Attributes in the DARIAH AAI

- Released by default: eppn, affiliation, unscoped-affiliation, entitlement
- Add and use unique-id for personalization (see previous slide)
- Configure in /etc/shibboleth/attribute-map.xml, by uncommenting: cn (CommonName), givenName, sn (surname), displayName, preferredLanguage, o (Organization), mail
- Configure DARIAH-specific Attributes, by adding:

```
<Attribute name="urn:oid:1.3.6.1.4.1.5923.1.5.1.1"  
id="isMemberOf"/>
```

```
<Attribute name="urn:oid:1.3.6.1.4.1.10126.1.52.5.2"  
id="dariahRole"/>
```

```
<Attribute name="urn:oid:1.3.6.1.4.1.10126.1.52.4.15"  
id="dariahTermsOfUse"/>
```

Attributes from Campus IdPs

- Adapt attribute-policy.xml to accept any scope
- Remove the Scope Checking stanza for affiliation and eppn

```
<afp:AttributeRule attributeID="affiliation">
    <afp:PermitValueRule xsi:type="AND">
        <RuleReference
            ref="eduPersonAffiliationValues"/>
<!-- accept any scope <RuleReference ref="ScopingRules"/> -->
    </afp:PermitValueRule>
</afp:AttributeRule> [...]
```

```
<!-- accept any scope for legacy users
    <afp:AttributeRule attributeID="eppn">
        <afp:PermitValueRuleReference ref="ScopingRules"/>
    </afp:AttributeRule> -->
```

Terms of Use for Service

- Generally DARIAH AAI ToU must be accepted by anyone using the DARIAH AAI
- The AAI proxy checks the ToU acceptance
- A DARIAH Service can have *additional* ToU
- Upload your ToU document to the DARIAH Repository, cf. DARIAH ToU:

<https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CB44-4>

Terms of Use for Service (2)

- Then request an *authorization group* attached with your service's ToU:
- Log in to the SelfService, and choose
 - Manage Groups
 - "+ new group" (at the bottom of the page)
 - Choose a group name, e.g. ***myservice-users***
 - Put the link to your ToU in the "Remarks" box
- DARIAH staff will create the group for you

Thanks!

<https://www.daasi.de>

<https://wiki.de.dariah.eu/display/publicde/DARIAH+AAI+Documentation>

