deutsches forschungsnetz

Upgrade auf Shibboleth IdP 4.x

DFN-AAI

Silke Meyer (smeyer@dfn.de)

Intro

- Support für Shibboleth IdP 3.4.6 / 3.4.7 läuft Ende 2020 aus.
  - End of Life bei Dependency: Spring Framework 4.x
- IdPs mit dem Entity Attribut Sirtfi *müssen* bis dahin auf 4.0.1 laufen.
- IdPs ohne Sirtfi: Bitte bis Ende April 2021 upgraden!
  - DFN-AAI Hotline kann nicht langfristig mehrere Versionen unterstützen.
  - Sicherheitslücken lassen sich durch Einspielen eines Minor Update auf einem aktuellen IdP schnell beheben.
- Vortrag richtet sich an IdP-Betreiber*innen

Überblick

- allgemeine Neuerungen im IdP 4.x
- Neuerungen bei Neuinstallation
- SAML Subject ID / SAML Pairwise ID
- Vorgehensweise beim Upgrade
- Ausblick auf 4.1

High Score 14.11.2020

IdP-Versionen	Anzahl der IdPs
3.1.2 – 3.3.3	44
3.4.1 – 3.4.5	53
3.4.6 – 3.4.7	124 (42% der Shibboleth IdPs in DFN-AAI)
4.0.0 – 4.0.1	75 (25% der Shibboleth IdPs in DFN-AAI)

Allgemeine Neuerungen im IdP 4.x

Systemanforderungen

- Java 11 (Amazon Coretto, OpenJDK, ähnliche Distributionen)
- Servlet Container mit Servlet API 3.1 (Jetty 9.4+, Tomcat 9+)
- bei Installation aus Paketquellen debianbasierter Linuxsysteme:
  - Debian 10
  - Ubuntu 18.04, 20.04
- ggf. ist der erste Schritt ein Betriebssystem-Upgrade

Geänderte Defaults für alle 4er IdPs

- vollständige Release Notes 4.0.0 , 4.0.1, 4.0.1.1 (nur Windows)
- idp.cookie.secure = true
- Output in Logs: Zeitformat geändert → evtl. für Logauswertung relevant

LDAP-Anbindung

- LDAP provider: UnboundID statt JNDI
- Änderung von Schreibweisen bei Zeitangaben für Timeouts und Pool-Einstellungen, z.B.:
  - vorher: `idp.authn.LDAP.connectTimeout = 10000`
  - nachher: `idp.authn.LDAP.connectTimeout = PT10S`
- Empfehlung aus dem Shibboleth-Wiki: 2 Dateien mit der mitgelieferten Version aus dem dist-Ordner zurücksetzen
  - `/opt/shibboleth-idp/conf/authn/ldap-authn-config.xml`
  - `/opt/shibboleth-idp/conf/authn/password-authn-config.xml`

Zur Wiederholung: Der dist-Ordner

- `/opt/shibboleth-idp/dist`
- enthält für die installierte IdP-Version die Originaldateien für
  - conf
  - flows
  - messages
  - views
  - webapp

Neuerungen bei Neuinstallation

Geänderte Defaults für Neuinstallationen

- HTML Local Storage, `idp.session.trackSPSessions` und `idp.session.secondaryServiceIndex` sind aktiviert (in `idp.properties`)
  - Single Logout-Unterstützung als Standard
  - HTML Local Storage nur relevant bei Speicherung der Sessions im Browser
  - DFN-Empfehlung: Serverseitige Speicherung der Sessions
- IdP-interne Secrets sind in neue Datei ausgelagert, für Tomcat-User nur lesbar (Doku)
- Cross-Site Request Forgery Protection ist aktiviert

Bei Neuinstallationen: AES-GCM

- Standard-Verschlüsselungsalgorithmus: AES-GCM statt AES-CBC
  - Ausnahmen nötig für SPs, die AES-CGM nicht entschlüsseln können
  - unsere Online-Dokumentation dazu
- Von Version 3.x aktualisierte IdPs nutzen weiterhin AES-CBC
  - Umstellung in idp.properties:

```
idp.encryption.config=shibboleth.EncryptionConfiguration.GCM
```

Ausnahmen für SPs mit AES-CBC

- conf/relying-party.xml

```
<util:list id="shibboleth.RelyingPartyOverrides">
 <bean parent="RelyingPartyByName" c:relyingPartyIds="#{{'ENTITYID1', 'ENTITYID2'}}">
  <property name="profileConfigurations">
   <list>
    <bean parent="SAML2.SSO"
     p:securityConfiguration-ref="shibboleth.SecurityConfiguration.CBC"
     p:postAuthenticationFlows="#{{'terms-of-use', 'attribute-release'}}"
     p:nameIDFormatPrecedence="#{{'urn:oasis:names:tc:SAML:2.0:nameid-format:persistent',
        'urn:oasis:names:tc:SAML:2.0:nameid-format:transient'}}" />
    <bean parent="SAML2.Logout" p:securityConfiguration-ref="shibboleth.SecurityConfiguration.CBC"/>
    <bean parent="SAML2.ArtifactResolution"
     p:securityConfiguration-ref="shibboleth.SecurityConfiguration.CBC"/>
   </list>
  </property>
 </bean>
</util:list>
```

Bei Neuinstallationen: relying-party.xml

- Standardmäßig sind die Profile für SAML 1.1 und SAML 2.0 Attribute Query nicht mehr aktiv:

```
<bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">
    <property name="profileConfigurations">
        <list>
            <bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" />
            <ref bean="SAML2.ECP" />
            <ref bean="SAML2.Logout" />
            <ref bean="SAML2.ArtifactResolution" />
            <ref bean="Liberty.SSOS" />
        </list>
    </property>
</bean>
```

Anleitung User Deprovisionierung via Attribute Query

Bei Neuinstallationen: idp-audit.log

▶127.0.0.2|2020-10-26T13:52:00.687823Z|2020-10-26T13:52:15.600145Z|user123|https://sp1.local/shibboleth|_9f786899400f14c054cfe3cea785c3a6|password|2020-10-26T13:52:06.204758Z|uid,mail,eduPersonScopedAffiliation,surname|NMTR6SVZOCUWF5MNGDDIYQQBY4QCB76N|persistent|false|false|AES128-GCM|Redirect|POST||Success||600961a7bf0de3d1c8995ab258dc3e3c60bcffc3e79a0d6370869d5ab2fa2cfc|Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefox/81.0

▶Source IP|Timestamp|Timestamp|Principal|SP EntityID|ID der SAML Assertion|Anmeldung mit Password|Timestamp|Übertragene Attribute|persistentId|Info, dass es die persistentId war|||Algorithmus|||StatusCode Value|||Session ID|User Agent

Attribute Registry Service

- Aufsplittung der Konfiguration:
  - vereinfachter Attribute Resolver: Attributquellen und -definitionen
  - Attribute Registry: Wie werden die Attribute in UI beschrieben, für SPs kodiert und welche Protokolle zur Übermittlung werden unterstützt?
- Regeln für Attribute in Schemata oder für einzelne Attribute hinterlegen
- Reload:

```
curl https://idp.beispiel-uni.de/idp/profile/admin/reload-service?id=shibboleth.AttributeRegistryService
```

Attribute Registry Service

Datei	Zweck
`conf/services.xml`	Aktivierung der Attribute Registry
`conf/attributes/`	• **.xml-Dateien** mit Transcoding-Regeln und UI-Infos • eine Datei pro Schema • Referenz in default-rules.xml
`conf/attributes/custom`	Vorgesehen für eigene Transcoding Rules, **.properties-Dateien**

conf/attributes/default-rules.xml

```
<!-- Default Attribute transcoding rules. -->
<import resource="inetOrgPerson.xml" />
<import resource="eduPerson.xml" />
<import resource="eduCourse.xml" />
<import resource="samlSubject.xml" />
```

conf/attributes/eduPerson.xml

```
<bean parent="shibboleth.TranscodingProperties">
  <property name="properties">
    <props merge="true">
      <prop key="id">eduPersonScopedAffiliation</prop>
      <prop key="transcoder">SAML2ScopedStringTranscoder</prop>
      <prop key="saml2.name">urn:oid:1.3.6.1.4.1.5923.1.1.1.9</prop>
      <prop key="displayName.de">Zugehörigkeit</prop>
      <prop key="description.de">Art der Zugehörigkeit zur Heimatorganisation</prop>
    </props>
  </property>
</bean>
```

Einbindung von dfnEduPerson und dfnMisc

- Transcoding Regeln für DFN-spezifische Attribute werden nicht mitgeliefert.
- Transcoding Regeln für Attribute im dfnEduPerson-Schema herunterladen
- Zusammenstellung von uns „dfnMisc“ (eduPersonTargetedId, bwidmOrgId, einige schac-Attribute) herunterladen
- beide einbinden in `conf/attributes/default-rules.xml`

Beispiel für einzelnes Attribut

- Achtung: nur Beispiel! Das Attribut ist bereits Teil von dfnMisc.xml.
- conf/attributes/custom/eduPersonTargetedId**.properties**

```
id=eduPersonTargetedID
transcoder=SAML2XMLObjectTranscoder
saml2.name=urn:oid:1.3.6.1.4.1.5923.1.1.1.10
displayName.de=Targeted ID (pseudonyme Kennung)
displayName.en=Targeted ID (pseudonymous ID)
description.de=Targeted ID: Eindeutige, pseudonyme Nutzerkennung, unterschiedlich pro SP
description.en=Targeted ID: A unique, pseudonymous identifier for a person, different for each SP
```

Bei Neuinstallationen: Aktivierte Attribute Registry

- Anschalten bei upgegradeten IdPs mit legacy `attribute-resolver.xml` führt zu doppelten Attributen!
- `siehe ./conf/services.xml`

```
<!--
This is suitable for new installs but will usually produce duplicate
Attribute output if a legacy resolver file is used that contains
AttributeEncoders.
-->
<util:list id ="shibboleth.AttributeRegistryResources">
    <value>%{idp.home}/conf/attribute-registry.xml</value>
    <value>%{idp.home}/system/conf/attribute-registry-system.xml</value>
    <value>%{idp.home}/conf/attributes/default-rules.xml</value>
    <value>%{idp.home}/conf/attribute-resolver.xml</value>
</util:list>
```

Feature: Attributexport aus DataConnector

- Export von kodierbaren Attributen aus DataConnector ohne Attributdefinition
- Vereinfachung von `attribute-resolver.xml` möglich
- Referenzierung über InputDataConnector (nicht: InputAttributDefinition)

```
<DataConnector id="myLDAP" xsi:type="LDAPDirectory"
    ldapURL="%{idp.attribute.resolver.LDAP.ldapURL}"
    baseDN="%{idp.attribute.resolver.LDAP.baseDN}"
    <!-- usw. -->
    exportAttributes="givenName sn displayName mail eduPersonAssurance">
    <FilterTemplate>
        <![CDATA[
            %{idp.attribute.resolver.LDAP.searchFilter}
        ]]>
    </FilterTemplate>
</DataConnector>
```

SAML Subject ID / SAML Pairwise ID

Von Attribut zu NameID und zurück...

IdP 2.x	eduPersonTargetedId (ePTId)	• Pseudonyme ID, pro SP unterschiedlich („targeted“) • beständig → für Wiedererkennung am SP nutzbar • Übertragung als Attribut
IdP 3.x	PersistentId (pid)	• Nachfolge von eduPersonTargetedId (selber Wert) • Übertragung als SAML2 NameID
	transientId	• unbeständig → pro SAML-Assertion generiert
IdP 4.x	SAML Pairwise-Id	• Spezifikation • Ablösung von ePTId und pid, aber scoped • Übertragung wieder als Attribut
	SAML Subject-Id	• Spezifikation • Unique ID + Scope („non-targeted“) • Ablösung von eduPersonPrincipalName und eduPersonUniqueId

zu NameIdentifiers im Shibboleth-Wiki

SAML Pairwise ID

- Ablösung von eduPersonTargetedId und persistentId
- `urn:oasis:names:tc:SAML:attribute:pairwise-id`
- langlebiger, nicht wiedervergebbarer, unidirektionaler, eindeutiger Identifier
- abhängig vom Service Provider („targeted“)
- Unique ID + Scope, Beispielwert:

  `Q3K2U5GAJ3BZHAASVMOWZTLJTOHMZPC2@beispiel-uni.de`
- Empfehlung: die beiden neuen Identifier zusätzlich zu den altbekannten am IdP konfigurieren

Konfigurationsbeispiel SAML Pairwise ID

```
<AttributeDefinition xsi:type="Scoped" id="samlPairwiseID" scope="%{idp.scope}">
    <InputDataConnector ref="myStoredId" attributeNames="persistentId"/>
</AttributeDefinition>
```

```
<DataConnector id="myStoredId" xsi:type="StoredId"
    generatedAttributeID="persistentId"
    salt="%{idp.persistentId.salt}">
    <InputAttributeDefinition ref="%{idp.persistentId.sourceAttribute}" />
    <BeanManagedConnection>shibboleth.MySQLDataSource</BeanManagedConnection>
</DataConnector>
```

SAML Subject ID

- Ablösung von eduPersonPrincipalName und eduPersonUniqueId
- `urn:oasis:names:tc:SAML:attribute:subject-id`
- langlebiger, nicht wiedervergebbarer, global eindeutiger Identifier für den Account
- unabhängig vom Service Provider
- case-insensitive! (Kleinschreibung wird empfohlen)
- Unique ID + Scope, Beispielwert:

`7179708adc9d5b03aa46544eafd1e4bb2ef3719ab27b8ee5e02e0c38499cf9@beispiel-uni.de`

Konfigurationsbeispiel SAML Subject ID

```
<AttributeDefinition id="subjectHash" xsi:type="ScriptedAttribute" dependencyOnly="true">
  <InputDataConnector ref="myLDAP" attributeNames="%{idp.persistentId.sourceAttribute}" />
  <Script><![CDATA[
    var digestUtils = Java.type("org.apache.commons.codec.digest.DigestUtils");
    var saltedHash = digestUtils.sha256Hex(%{idp.persistentId.sourceAttribute}.getValues().get(0) +
        "%{idp.persistentId.salt}");
    subjectHash.addValue(saltedHash);
  ]]></Script>
</AttributeDefinition>


<AttributeDefinition xsi:type="Scoped" id="samlSubjectID" scope="%{idp.scope}">
    <InputAttributeDefinition ref="subjectHash" />
</AttributeDefinition>

<!-- eduPersonUniqueId: gleicher Wert wie für Subject Id -->
<AttributeDefinition xsi:type="Scoped" id="eduPersonUniqueId" scope="%{idp.scope}">
    <InputAttributeDefinition ref="subjectHash" />
</AttributeDefinition>
```

Vorgehen beim Upgrade

Vorgehen beim Upgrade I

- Empfehlung der IdP-Entwickler für das Upgrade auf Version 4: Räumen Sie nur auf, ...
- … was in den Logs als „Deprecated“ markiert ist,
- … was in den Release Notes erwähnt ist.
- „We urge people in the strongest possible terms NOT to worry about "cleaning up" anything but the things the log warns are deprecated or that are highlighted in the Release Notes. The V4 horizon is not expected to be short and there's plenty of time later to worry about cleaning up or updating settings.“
- Quelle: Shibboleth Development Center

Vorgehen beim Upgrade I

- empfohlen für IdPs ab Version 3.4.1
- ggf. Upgrade des Betriebssystem, Java, Tomcat
- Datensicherung von idp.home und Datenbank
- nicht überschrieben werden ./conf, ./views, ./messages und ./edit-webapp
- Aktualisierung auf 3.4.7
  - Abarbeiten der Deprecation Warnings im idp-process.log bzw. idp-warn.log
  - Hilfestellung: https://doku.tid.dfn.de/de:shibidp3upgrade
- Wenn IdP 3.4.6 bzw. 3.4.7 ohne Warnungen / Fehler läuft, geht es weiter.

Aufwändigste Anpassung

- ab 3.4.2 Ersetzen der `Dependency`-Direktive in `attribute-resolver.xml`:

```
<!-- alt -->
<AttributeDefinition id="mail" xsi:type="Simple" sourceAttributeID="mail">
    <Dependency ref="myLDAP" />



<!-- neu -->
<AttributeDefinition id="mail" xsi:type="Simple">
    <InputDataConnector ref="myLDAP" attributeNames="mail"/>



<!-- neu -->
<AttributeDefinition xsi:type="Scoped" id="eduPersonScopedAffiliation" scope="%{idp.scope}">
    <InputAttributeDefinition ref="eduPersonAffiliation"/>
```

Vorgehen beim Upgrade II

- IdP 4.x Release Notes
- Gewohnte Upgrade-Routine:
  - Download und Entpacken des IdP 4.x
  - Aufruf des Installers (Linux: `./bin/install.sh -Didp.conf.filemode=644`), Angabe des Zielverzeichnisses der alten Installation
  - ./war/idp.war neu bauen mit `./bin/build.sh`
- IdP-Logs beobachten und testen
- neue Deprecation Warnings abarbeiten

Neuinstallation

Vorgehen bei Neuinstallation

- empfohlen für IdP <= 3.3.x
- Systemanforderungen prüfen
- Tutorial für Linux (Debian 10) mit OpenJDK 11, Apache Tomcat 9 und Apache Webserver 2.4
- Übertragen der eigenen Konfiguration in die neue Syntax
- ideal: erst Test-IdP, dann separates Produktivsystem aufsetzen
- Neues Produktivsystem bekommt **dieselbe EntityID** wie das alte!
- Doku: Migration im laufenden Betrieb

Ausblick auf Shibboleth IdP 4.1

Grobe Roadmap

- Quelle: Shibboleth Development Center
- Die Version 5.0 soll erst kommen, wenn es technisch nötig ist.
  - Bsp: abgekündigte Abhängigkeiten
- angepeilter Zeitraum für 4.1-Release: Q1 2021
- „the next ‚big' release of the IdP"
- Vereinfachungstrend setzt sich fort
- Es profitieren v.a. Neuinstallationen davon.

Laufender Umbau der IdP-Architektur

- Auslagern von Funktionalitäten aus Core in IdP-Module mit eigenem CLI
  - Beispiel: IdP Login Flows werden Module
  - Konfigurationen von Modulen bis zum Anschalten versteckt → Vereinfachung
  - Konfiguration in `.properties`- statt `.xml`-Dateien
- neue Plugin-Architektur (Bsp.: Alternative Scripting Enging zu Nashorn)
  - Plan: Bestandteile agiler entwickeln können
- Einführung von Tabs im Shib Wiki zur Trennung von 4.0 und 4.1-Dokumentation

Vielen Dank! Gibt‘s Fragen?

- Kontakt
  - DFN-AAI Team

    E-Mail: aai@dfn.de
    Tel.: +49-30-884299-9124
    Fax: +49-30-884299-370

    Anschrift:
    DFN-Verein, Geschäftsstelle
    Alexanderplatz 1
    10178 Berlin