IDP 3 USI Wien eduPersonEntitlement

Funktionales *Beispiel* einer Konfiguration für den Shibboleth 3.x IDP und Java 8 (Scripting Engine "Nashorn"), um das gewünschte USI Wien-eduPersonEntitlement zu berechnen, mittels dessen Studierende unter 25 Jahre ohne zusätzliche Umstände vergünstigte Tarife bei Kursen des USI Wien i n Anspruch nehmen können. Dieses etwas komplexe Vorgehen verhindert, das Alter oder Geburtsdatum von Personen an den Service Provider schicken zu müssen. Gleichzeitig dient es als Beispiel für die Erzeugung bzw. komplexere Manipulation von Attributen im Shibboleth IDP.

LDAP oder RDBMS

Zur Berechnung des aktuellen Alters einer Person wird zuerst einmal das Geburtsdatum einer Person benötigt. Hat man dafür noch kein eigenes Attribut im LDAP-Verzeichnisdienst vorgesehen, bietet sich `schacDateOfBirth` aus dem SCHAC-Schema an. Dazu muß das Schema dem LDAP DSA bekannt gemacht, sowie allen (Benutzer-)Objekten, die einen `schacDateOfBirth`-Wert beinhalten können, die "auxiliary objectClass" `schacPersonalCharacteristics` zugewiesen werden. Wie im SCHAC-Schema angegeben, wird das Datum im Format `JJJJMMTT` gespeichert. Der 31. Dezember 1970 etwa würde so zu `19701231`.

IDP-Konfiguration

attribute-resolver.xml

Um aus dem Geburtsdatum im IDP das Alter zu errechnen, braucht es etwas eigenen Code, über eine `ScriptedAttribute`-Definition des Shiboleth IDP. Damit wird die Integration von interpretierten Programmiersprachen wie ECMAScript/JavaScript (oder auch Jython) mit den Java-Objekten des IDP möglich, ohne komplexe Erweitungen für den IDP in Java programmieren zu müssen.

Im folgenden Beispiel erzeugen wir `eduPersonEntitlement`-Attributwerte für die Zwecke des USI Wien-Services durch Berechnung aus `schacDateOfBirth`-Attributwerten. Bereits im LDAP-Verzeichnisdienst existierende (andere) Werte von `eduPersonEntitlement` werden hier automatisch inkludiert. Soll der IDP noch weitere Entitlements erzeugen können, so kann man die Definitionen, wo diese anderen Werte herkommen, einfach als zusätzliche Dependency-Elemente hinzufügen – untenstehend ein Beispiel (`MappedEntitlements`) unserer Anleitung zur Erzeugung von Entitlements folgend:

eduPersonEntitlements per Script erzeugen (Java8/Nashorn)

```
<AttributeDefinition id="eduPersonEntitlement" xsi:type="ScriptedAttribute" >
    <Dependency ref="myLDAP" />
    <Dependency ref="MappedEntitlements" />
    <AttributeEncoder xsi:type="SAML1String" name="urn:mace:dir:attribute-def:eduPersonEntitlement" encodeType="
false" />
    <AttributeEncoder xsi:type="SAML2String" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.7" friendlyName="
eduPersonEntitlement" encodeType="false" />
    <Script><![CDATA[
var logger = Java.type("org.slf4j.LoggerFactory").getLogger("net.shibboleth.idp.attribute.resolver.script.
usidiscount");
if (typeof schacDateOfBirth == "undefined" || schacDateOfBirth.getValues().size() < 1) {
  logger.debug("no schacDateOfBirth, ignoring");
} else {
  var dob = schacDateOfBirth.getValues().get(0);
  logger.trace("schacDateOfBirth is " + dob);
  var LocalDate  = Java.type("java.time.LocalDate");
  dob   = LocalDate.of(dob.slice(0,4), dob.slice(4,6), dob.slice(6,8));
  logger.trace("parsed dob is " + dob);
  var age = Java.type("java.time.Period").between(dob, LocalDate.now()).getYears();
  logger.trace("age is " + age);
  if (age < 25) {
    logger.debug("age is within USI limits, adding USI-entitlement");
    logger.trace("eduPersonEntitlement had values " + eduPersonEntitlement.getValues());
    eduPersonEntitlement.addValue("http://usi.at/student-discount");
    logger.trace("eduPersonEntitlement has values " + eduPersonEntitlement.getValues());
  } else {
    logger.debug("age is not within USI limits, doing nothing");
  }
}
]]></Script>
</AttributeDefinition>

<!-- Beispiel fuer "MappedEntitlements" -->
<AttributeDefinition id="MappedEntitlements" xsi:type="Mapped" sourceAttributeID="eduPersonAffiliation">
    <Dependency ref="eduPersonAffiliation" />
    <ValueMap>
        <ReturnValue>urn:mace:dir:entitlement:common-lib-terms</ReturnValue>
        <SourceValue>member</SourceValue>
    </ValueMap>
    <ValueMap>
        <ReturnValue>urn:mace:terena.org:tcs:personal-user</ReturnValue>
        <SourceValue>member</SourceValue>
    </ValueMap>
</AttributeDefinition>
```

Hier wird mit JavaScript ein Attribut namens `eduPersonEntitlement` definiert. Ausgangsbasis für dessen Werte ist hier das Attribut `schacDateOfBirth`, das aus einem LDAP-Verzeichnisdienst gelesen wird (Dependency "`myLDAP`"). Ist `schacDateOfBirth` definiert (weil die LDAP-Abfrage einen Wert zurückgeliefert), wird der Wert zerlegt und in ein Datumsobjekt verwandelt. Die Differenz in Jahren zwischen diesem Objekt und dem heutigen Tag ergibt das aktuelle Alter der zugreifenden Person.

Ist nun das Alter kleiner 25, wird ein Entitlement-Wert spezifisch für das USI Wien erzeugt. Ist das Alter größer oder gleich 25 oder gab es kein `schacDateOfBirth` in der LDAP-Abfrage, wird kein zusätzlicher Entitlement-Wert erzeugt.

logback.xml

Um bei Bedarf im laufenden IDP zu sehen, was im Detail bei der obigen Altersberechnung passiert, kann dies gezielt mit einem Eintrag in der Datei logback.xml aktiviert (und wieder deaktiviert) werden:

Debug-Logging für Script-AttributeDefinition

```
<logger name="net.shibboleth.idp.attribute.resolver.script.usidiscount" level="DEBUG" />
```

Stellt man das Loglevel für dieses spezifische Modul auf `DEBUG` (und wartet bis zu 5 Minuten, bis die logback.xml Konfiguration neu eingeladen wurde – oder man lädt diese manuell neu), schreibt der IDP in die Datei `idp-prozess.log`, ob die Werte vorhanden sind oder nicht, sowie ob aufgrund des Alters nun ein Entitlement erzeugt wurde oder nicht. Nur wenn man das Loglevel hier auf `TRACE` stellt, sind auch das zur Berechnung verwendete Geburtsdatum und die eduPersonEntitlement-Werte (vorher/nachher) selbst zu sehen. Ist man mit der IDP-Konfiguration zufrieden, kann das Loglevel für diesen logger zurück auf `INFO` gestellt werden, womit nach wenigen Minuten (oder nach Neuladen der `logback`-Konfiguration, jedenfalls ohne Neustart des IDP) keine weiteren Logzeilen für diesen logger geschrieben werden.

attribute-filter.xml

> Eine eigene Freigaberegel für das USI Wien ist **nicht nötig, wenn** man bereits die (empfohlene) "ACOnet-registered services" Regel für Attributweitergabe verwendet!

Für die gezielte Weitergabe der/des erzeugten eduPersonEntitlements kann nun eine neue `AttributeFilterPolicy` eingefügt werden:

Freigabe in der Attribute Filter-Konfiguration

```
<!-- USI-Wien Student Discount -->
<AttributeFilterPolicy id="USIWien">
    <PolicyRequirementRule xsi:type="AND">
        <Rule xsi:type="Requester" value="https://www.usi-wien.at/shibboleth" />
        <Rule xsi:type="Value" attributeID="eduPersonAffiliation" value="student" />
    </PolicyRequirementRule>
    <AttributeRule attributeID="eduPersonEntitlement">
        <PermitValueRule xsi:type="Value" value="http://usi.at/student-discount" />
    </AttributeRule>
</AttributeFilterPolicy>
```

Damit wird für das Attribut mit `id="eduPersonEntitlement"` nur der Wert `http://usi.at/student-discount` weitergegeben, und zwar *nur*, wenn das Attribut nach der Berechnung oben einen Wert hat, wenn der Service Provider jener des USI Wien ist, *und* wenn die zugreifende Person eine eduPersonAffiliation mit dem Wert `student` hat. Andere Service Provider bekommen das Attribut nicht, das USI Wien bekommt andere Entitlements nicht, und für Personen, die nicht auch Studierende sind, wird ebenfalls nichts weitergegeben.