Steffen Hofmann
Freie Universität Berlin
ZEDAT
Identity Management & Media (IM)

Freie Universität Berlin

IdP3-Authentifizierung

Freie Universität Berlin

Übersicht

- Verfügbare Authentifizierungsverfahren
- Aktivierung Authentifizierungsverfahren
- Authentifizierungsverfahren Password
- Authentifizierungsverfahren External inkl. Übung
- PostLoginSubjectCanonicalization
- Principals und AuthenticationContextClasses
- Diskussion 2-Faktor-Authentifizierung

Freie Universität Berlin

Verfügbare Authentifizierungsverfahren

- in **`${idp.home}/conf/authn/general-authn.xml`**
  - authn/IPAddress
  - authn/RemoteUser
  - authn/RemoteUserInternal
  - authn/X509
  - authn/X509Internal
  - authn/Password
  - authn/External

Freie Universität Berlin

Aktivierung Authentifizierungsverfahren

- in **`${idp.home}/conf/idp.properties`**
  für alle Profile mit Property
  `idp.authn.flows=Password`
  oder auch
  `idp.authn.flows=IPAddress|External|Password`

- Für einzelne Profile in
  `${idp.home}/conf/relying-party.xml`
  mit z.B.

```
<bean parent="SAML2.SSO"
p:authenticationFlows="#{{'RemoteUser'}}" />
```

Freie Universität Berlin

Authentifizierungsverfahren Password

- in **`${idp.home}/conf/authn/password-authn-config.xml`** kann/darf genau ein Verfahren ausgewählt werden

```
<!-- Choose an import based on the back-end you want to use. -->
<import resource="jaas-authn-config.xml" />
<!-- <import resource="krb5-authn-config.xml" /> -->
<!-- <import resource="ldap-authn-config.xml" /> -->
```

- weitere Details über https://wiki.shibboleth.net/confluence/display/IDP30/PasswordAuthnConfiguration

Authentifizierungsverfahren External

- in **${idp.home}/conf/authn/external-authn-config.xml**

```
<bean id="shibboleth.authn.External.externalAuthnPath“
   class="java.lang.String"
   c:_0="contextRelative:myownauth.jsp" />
```

Freie Universität Berlin

Authentifizierungsverfahren External, Teil 2

- aus https://wiki.shibboleth.net/confluence/display/IDP30/ExternalAuthnConfiguration

1. *Call ExternalAuthentication.startExternalAuthentication(HttpServletRequest), saving off the result as a key.*
2. *Do work as necessary (reading request details from the attributes below). Any redirects must preserve the key value returned in step 1 because it must be used to complete the login later.*
3. *Set request attributes to communicate the result of the login back.*
4. *Call ExternalAuthentication.finishExternalAuthentication(String, HttpServletRequest, HttpServletResponse). The first parameter is the key returned in step 1.*

Authentifizierungsverfahren External, Teil 3

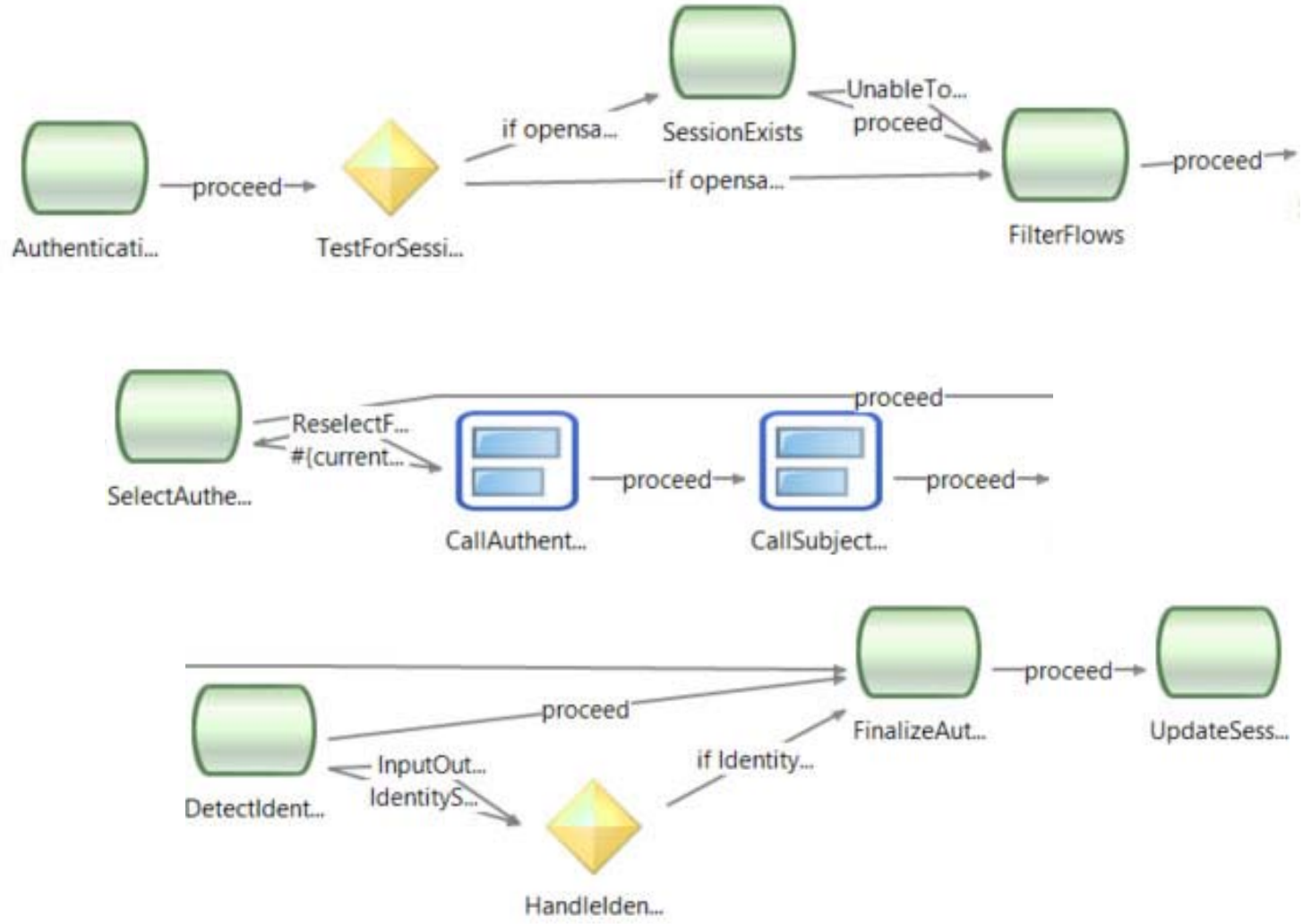
```
ExternalAuthentication.startExternalAuthentication(request);

String conversationKey = (String)
   request.getParameter(ExternalAuthentication.CONVERSATION_KEY);

# eigenes Authentifizierungsverfahren mit z.B. uid als Rückgabewert

request.setAttribute(ExternalAuthentication.PRINCIPAL_NAME_KEY,
   uid);
request.setAttribute(ExternalAuthentication.DONOTCACHE_KEY, false);

ExternalAuthentication.finishExternalAuthentication(conversationKey
   , request, response);
```

Freie Universität Berlin

Authentifizierungsverfahren External, Übung

→ Eigenes externes Authentifizierungsverfahren mit einer JSP, die statisch eine uid setzt

Authentication Flow vom IdP



Freie Universität Berlin

PostLoginSubjectCanonicalization

- in **${idp.home}/conf/conf/c14n/subject-c14n.xml**
  wird definiert, wie der Principal(Name) für die Nachfolgeprozesse wie den AttributeResolver ausgewählt wird

```
<util:list id="shibboleth.PostLoginSubjectCanonicalizationFlows">
   <!--<ref bean="c14n/x500" /> -->
   <!--<ref bean="c14n/attribute" /> -->
   <ref bean="c14n/simple" />
</util:list>
```

- Auschnitt aus
  ${idp.home}/conf/conf/c14n/simple-subject-c14n-config.xml

```
<!-- Simple transforms to apply to username after authentication. -->
<util: [..] simple.Lowercase" static-field="java.lang.Boolean.FALSE"/>
<util: [..] simple.Uppercase" static-field="java.lang.Boolean.FALSE"/>
<util: [..] simple.Trim" static-field="java.lang.Boolean.TRUE"/>
```

Freie Universität Berlin

Principals und AuthenticationContextClasses

- AuthenticationContextClasses werden als Principals zu den Authentifizierungsverfahren hinzugefügt
- Wird eine konkrete AuthenticationContextClass vom Service Provider angefordert, so wird geguckt, ob dieser Principal Bestandteil aller unterstützen Principals des Authentifizierungsverfahrens ist
- Für die Entscheidung der Zuständigkeit können auch andere Matching-Rules definiert werden
- Eine Map mit Weights entscheidet darüber, welche AuthenticationContextClass zurückgegeben wird, wenn mehrere mögliche verbleiben
- Default: urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport

Freie Universität Berlin

Diskussion

→ 2-Faktor-Authentifizierung

Freie Universität Berlin

Kontakt

Steffen Hofmann
steffen.hofmann@fu-berlin.de
+49 30 838 56031