Bundesministerium
für Bildung
und Forschung

Anwendungen schützen mit Shibboleth

4. Shibboleth-Workshop

Berlin, 28. Februar 2007

Bernd Oberknapp
Universitätsbibliothek Freiburg
E-Mail: bo@ub.uni-freiburg.de

Übersicht

- Unterstützte Plattformen
- Nutzungsvarianten
- SessionInitiator (Teil1)
- Beispiele:
  - Autorisierung über Attribute
  - Bereitstellung des REMOTE_USER
  - Kombination mit IP-Kontrolle
  - optionale Attribute
- SessionInitiator (Teil 2)
- LazySession:
  - Beispiele
  - Umstellung von Anwendungen
- Empfehlungen

Unterstützte Plattformen

- Für den Schutz von Anwendungen wird der Shibboleth Service Provider (SP) benötigt.
  - Der C++ SP (aktuell Version 1.3f) steht zur Verfügung für
    - Apache (Linux/Unix, Windows)
    - IIS (Windows)
    - weitere Webserver eventuell über FastCGI (z.B. lighttpd)
  - Bei Shibboleth 2.0 wird es auch einen Java SP geben.
- Die Beispiele im Folgenden verwenden den C++ SP unter Apache, die Konfiguration erfolgt über die HtAccessControl, d.h. die Apache-Konfiguration.
- Beim Apache ist die XMLAccessControl, d.h. die Konfiguration über die shibboleth.xml des SP, eine Alternative, beim IIS gibt es nur diese Variante.

Nutzungsvarianten

- Der Shibboleth SP bietet zwei Möglichkeiten, die auch kombiniert werden können:
  - Apache Authentication und Access Control, d.h. Shibboleth löst die Authentifizierung beim ersten Zugriffsversuch aus und kontrolliert den Zugriff auf die Ressourcen.
  - LazySession, d.h. die Anwendung muss bei Bedarf selbst die Authentifizierung auslösen und selbst den Zugriff auf die Ressourcen kontrollieren.
- Bei beiden Varianten können die Informationen über den Nutzer (Attribute), die der Shibboleth Identity Provider (IdP) übermittelt hat, der Anwendung über HTTP-Header zur Verfügung gestellt werden (über die Attribute Acceptance Policy in AAP.xml konfiguriert).

SessionInitiator

- Was „löst die Authentifizierung aus“?
  Antwort: Aufruf eines SessionInitiator!
- Ein SessionInitiator
  - definiert eine lokale URL, die aufgerufen werden kann, wenn eine Authentifizierung erfolgen soll
  - verweist auf einen WAYF oder direkt auf einen IdP
  - kann über seine ID angesprochen werden.
- Soll die Authentifizierung ausgelöst werden, wird der per ID angegebene oder der Default-SessionInitiator aufgerufen, der den Nutzer zu einem WAYF oder direkt zu einem IdP leitet.

SessionInitiator

Beispiel SessionInitiator-Konfiguration des SP auf dem AAR-Webserver (in shibboleth.xml):

```
<SessionInitiator id="AAR" isDefault="true"
  Location="/WAYF/DEMOaar"
  Binding="urn:mace:shibboleth:sp:1.3:SessionInit"
  wayfURL="https://aar.vascoda.de/DEMOaar/WAYF"
  wayfBinding="urn:mace:shibboleth:1.0:profiles:AuthnRequest"/>

<SessionInitiator id="unifr"
  Location="/WAYF/unifr"
  Binding="urn:mace:shibboleth:sp:1.3:SessionInit"
  wayfURL="https://www-fr.redi-bw.de/idp/unifr/SSO"
  wayfBinding="urn:mace:shibboleth:1.0:profiles:AuthnRequest"/>
```

Beispiel 1

- Im einfachsten Fall erfolgt nur eine Authentifizierung für die Anwendung (HTML-Dateien, Skripte, ...) ohne Zugriffskontrolle über Attribute.
- Beispiel AAR-Demoseite:

```
<Location /demo>
  AuthType Shibboleth
  ShibRequireSession On
  require valid-user
</Location>
```

löst beim Zugriff die Authentifizierung über den Default-SessionInitiator aus

beliebige authentifizierte Sitzung

- **Achtung**: Ohne weitere Absicherung des SP ist eine solche Konfiguration gefährlich, da sich alle Nutzer von allen IdPs einloggen können, von denen der SP SAML Authentication Assertions akzeptiert!

Beispiel 2

- Der Zugriff kann durch eine „require user“-Regel auf bestimmte Nutzer eingeschränkt werden.
- Beispiel interne AAR-Webseite mit Zugriff nur für das Freiburger AAR-Team:

```
<Location /interna>
  AuthType Shibboleth
  ShibRequireSessionWith unifr
  ShibRedirectToSSL 443
  require user fb91@uni-freiburg.de \
               hu25@uni-freiburg.de \
               lienhard@uni-freiburg.de \
               oberknap@uni-freiburg.de \
               ruppert@uni-freiburg.de
</Location>
```

SessionInitiator mit der ID unifr verwenden, Redirect direkt zum Freiburger IdP

bei Zugriff über Port 80/http Redirect auf Port 443/https

eduPerson-PrincipalNames des Freiburger AAR-Teams

Beispiel 3

- Statt die Nutzer explizit einzutragen, kann auch ein Attribut verwendet werden, das genau den zugriffs-berechtigten Nutzern über das Identity Management zugewiesen wird – das ist in vielen Fällen die bessere Alternative.
- Beispiel interne AAR-Webseiten (vgl. Beispiel 2):

```
<Location /interna>
  AuthType Shibboleth
  ShibRequireSessionWith unifr
  ShibRedirectToSSL 443
  require entitlement \
     urn:mace:ub.uni-freiburg.de:entitlement:aar:team
</Location>
```

Zugriff wird nur gestattet, wenn der IdP dieses eduPersonEntitlement für den Nutzer übermittelt

Achtung: Jeder IdP kann dieses Entitlement liefern, wenn dies nicht über die AAP eingeschränkt wird!

REMOTE_USER

- Viele Anwendungen benötigen eine Benutzerkennung, da die Rechte in der Anwendung darüber gesteuert werden.
- Shibboleth kann eine Benutzerkennung genau wie die Apache (Basic) Authentication im REMOTE_USER zur Verfügung stellen.

→ Damit kann jede Anwendung, die über Apache (Basic) Authentication und Access Control geschützt werden kann, auch über Shibboleth geschützt werden – ohne Änderungen an der Anwendung vorzunehmen!

- Beispiele: Nagios, BackupPC, WebSVN, ...

REMOTE_USER

- Per Default wird der Attributwert von eduPerson-PrincipalName (ePPN) als REMOTE_USER verwendet (siehe Default AAP.xml).
- Falls eine Anwendung Probleme mit dem Scope beim ePPN hat, kann alternativ auch die uid (Principal ohne Scope: user statt user@*domain*) verwendet werden – aber nur, wenn die Anwendung nur innerhalb einer Einrichtung genutzt wird, sonst ist die Eindeutigkeit der uid nicht gewährleistet!
- Wenn der Nutzer wiedererkannt werden muss, die Identität aber nicht bekannt sein muss, sollte aus Datenschutzgründen die eduPersonTargetedID (Pseudonym für den Nutzer) verwendet werden.

Beispiel 4

- Wenn der IdP das entsprechende Attribut nicht liefert, ist der REMOTE_USER zwar definiert, aber leer! Mit einer entsprechenden „require user“-Regel kann dies abgefangen werden:

```
<Location /backuppc>
  AuthType Shibboleth
  ShibRequireSession On
  ShibRequireAll On
  Require user ~ ^.+$
  Require entitlement \
     urn:mace:ub.uni-freiburg.de:entitlement:unifr:backup:admin
</Location>
```

und-Verknüpfung, beide Require-Regeln müssen erfüllt sein

regulärer Ausdruck stellt sicher, dass REMOTE_USER mindestens aus einem Zeichen besteht

URNs dürfen nur verwendet werden, wenn der entsprechende Namensraum registriert ist!

Beispiel 5

- Shibboleth kann mit IP-Kontrolle kombiniert werden.
- Beispiel Zugriff auf CILP (PDF- und Text-Dateien):

```
<Location /data/base/cilp>
  AuthType Shibboleth
  ShibRequireSessionWith unifr
  ShibRequireAll On
  Require affiliation ~ ^.+@uni-freiburg\.de$
  Require entitlement \
     urn:mace:dir:entitlement:common-lib-terms
  Order allow,deny
  Allow from 132.230.21.21 132.230.21.22 132.230.21.23 \
            132.230.21.24 132.230.21.25 132.230.21.27 \
            132.230.21.29 132.230.21.30
  Satisfy any
</Location>
```

Zugriff nur für berechtigte Nutzer der Uni Freiburg

oder

von den öffentlichen Rechercheplätzen in der UB Freiburg

Empfehlungen

- Es sollten nur die Attribute über eine require-Regel verlangt werden, die für den Schutz der Anwendung unbedingt notwendig sind.
- Weitere, optionale Attribute können aus den HTTP-Headern ausgelesen und genutzt werden, wenn sie vom IdP geliefert werden.
- Falls nur ein Teil der Anwendung (bestimmte URLs) geschützt werden muss, so sollten nur für diesen entsprechende require-Regeln definiert werden.
- Dies gilt insbesondere für Anwendungen mit eigenem Session-Management – hier genügt es häufig, nur den Teil der Anwendung zu schützen, der für den Aufbau der Anwendungssession zuständig ist.

Beispiel 6

- Beispiel Schnellsuchportal der UB Freiburg (IPS):
  - Geschützt wird nur die URL /shiblogin, unter der das Login-Skript liegt, das die IPS eigene Session aufbaut.
  - urn:mace:dir:entitlement:common-lib-terms wird über eine Regel in der AAP nur vom IdP der Uni Freiburg akzeptiert, so dass nur berechtigte Nutzer der Uni Freiburg Zugriff auf die lizenzpflichtigen Datenbanken haben.
  - Wird vom IdP auch die Benutzerkennung (momentan noch uid, zukünftig eduPersonTargetedID) übermittelt, kann der Nutzer Suchstrategien und Zitate persistent speichern.

```
<Location /shiblogin>
  AuthType Shibboleth
  ShibRequireSession On
  Require entitlement urn:mace:dir:entitlement:common-lib-terms
</Location>
```

SessionInitiator

- Ein SessionInitiator kann auch direkt mit folgenden optionalen Parametern aufgerufen werden:
  - **providerId**: schickt den Nutzer direkt zum angegebenen IdP, egal welche wayfURL im SessionInitiator angegeben ist
  - **target**: URL, zu der der Nutzer nach der Authentifizierung (Default: Application/@homeURL) weitergeleitet wird
  - **acsIndex**: Endpunkt für die SAML-Antwort

→ Durch Aufruf eines SessionInitiators mit providerId- und target-Parametern kann eine Anwendung selbst steuern, welcher IdP angesprochen wird und welche Seite nach Rückkehr vom IdP aufgerufen wird!

- Beispiel Testseite auf dem ReDI-Webserver

LazySession

- Bei LazySession muss die Anwendung selbst die Kontrolle darüber übernehmen,
  - wann eine Authentifizierung ausgelöst wird,
  - welcher WAYF oder IdP dabei angesprochen wird,
  - welche URL nach der Authentifizierung aufgerufen wird und
  - wie die von Shibboleth über HTTP-Header zur Verfügung gestellten Attribute ausgewertet und für die Zugriffskontrolle verwendet werden.
- Konfiguration für LazySession:

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

Achtung! Beim Zugriff auf /lazy wird **keine Authentifizierung** ausgelöst (kein ShibRequireSession!) und es werden **keine Attribute geprüft** (keine „require <Attribut>”-Regel!) – das muss die Anwendung selbst übernehmen!

Beispiele mit LazySession

- Standortkatalog der Universität Freiburg: LazySession für gesamte Anwendung unter /stk, externer WAYF
- ReDI: LazySession nur für Login-Skript zum Aufbau der ReDI eigenen Session unter /shib, integrierter eigener WAYF
- WISO (GENIOS-GBI):
  - Homepage
  - direkter Aufruf einer Zeitung (wie z.B. aus der EZB)
- ScienceDirect (Elsevier)
- JSTOR
- Shibboleth-Wiki

Umstellung von Anwendungen auf Shibboleth mit LazySession

- Für die Umstellung von Anwendungen auf Shibboleth mit LazySession gibt es kein „Kochrezept“!
- Folgende Punkte sind u.a. für die Umstellung relevant:
  - Wie werden die Ressourcen bisher geschützt (Apache, Tomcat, eigenes Verfahren, ...)?
  - Existiert ein eigenes Session-Management?
  - Kann dieses weiter verwendet werden, z.B. indem eine Sitzung über Shibboleth aufgebaut wird?
  - Existiert eine eigene Rechteverwaltung?
  - Können die dafür notwendigen Informationen per Shibboleth über Attribute bereitgestellt werden?
  - Können die Identity-Provider die Attribute liefern?

Der schnellste Weg zu einem funktionierenden Test-SP

- Setzen Sie keinen eigenen IdP auf (der IdP ist deutlich komplexer als der SP!), sondern verwenden Sie einen vorhandenen IdP, z.B. den AAR oder DFN Test-IdP oder TestShib.org.
- Verwenden Sie eine gut unterstützte Plattform, am besten Apache unter einer gängigen Linux-Distribution.
- Verwenden Sie die angebotenen Binärpakete (Fedora, Red Hat, Ubuntu) oder die Source RPMs (SuSE) für die Installation!
- Testen Sie den SP zunächst mit einem Testskript (z.B. von der Uni Leuven) um sicherzustellen, dass der SP funktioniert und die gewünschten Attribute ankommen!
- Testen Sie dann die Anwendung.

Aufbau einer kompletten Shibboleth-Umgebung

- Für eine komplette Shibboleth-Umgebung sind mindestens ein IdP und ein SP notwendig – zwei SPs, wenn das Single Sign-On mit Shibboleth genutzt oder demonstriert werden soll.
- Falls es in Ihrer Einrichtung bereits einen IdP gibt, verwenden Sie diesen oder sprechen Sie sich mit dem Betreiber ab – es sollte in jeder Einrichtung möglichst nur einen IdP geben!
- Empfehlung: Fangen Sie mit der Umstellung auf Shibboleth bei internen Anwendungen an, die bisher per Apache Basic Authentication und Access Control geschützt werden – das erfordert keine Änderungen an den Anwendungen und minimiert die Außenwirkung, wenn am Anfang mal etwas nicht funktioniert.

Und zum Schluss...

…noch eine Bitte: Warten Sie mit dem Aufbau eigener Shibboleth-Systeme noch bis nach dem Shibboleth 2.0-Vortrag am Ende des Workshops 🙂, aber:

Warten Sie nicht auf Shibboleth 2.0!

Vielen Dank für Ihre Aufmerksamkeit!