2008年度実証実験成果報告 大阪大学サイバーメディアセンター1

大阪大学サイバーメディアセンターに戻る
Shibboleth 2.0 の属性マッピング機能の検証（2008年度実証実験成果報告より）

システム構成

HP BladeSystem c3000

ブレード: BL460c x 2
CPU:QuadCoreXeon X5355 2.66GHz*1
MEM:1GB FB-DIMM *2 + 2GB FB-DIMM x 2 (10GB)
ストレージ: SB40c
146GB SAS HDD 146GB 10krpm 2.5” SAS

ホストOS: VMWare Infrastructure Standard Edition

ゲストOS-01: CentOS 5
IdPとして利用
サーバソフトウェア構成
Apache 2.2.3 (httpd-2.2.3-11.el5_1.centos.3)
Sun Java SE 1.6.0_07-b06
Tomcat 6.0.18
Shibboleth IdP 2.0.0
SQLite 3.3.6 (sqlite-3.3.6-2)

ゲストOS-02: CentOS 5
SPとして利用
Apache 2.2.3 (httpd-2.2.3-11.el5_1.centos.3)
Shibboleth C++ SP 2.0.0

Sun Fire T2000

CPU: 1.2GHz UltraSPARC T1 (8Core)
MEM: 16GB
ストレージ: 73GB 10000回転 SAS x 4

ホストOS: Solaris 10 8/07
ldap01p-test, ldap02p-test はマルチマスタ Directory Server

ゲストOS-03 (zone): Solaris 10 8/07
LDAP master サーバとして利用
Sun Java SE 1.5.0_12
Sun Java(TM) System Directory Server/5.2_Patch_4 B2005.230.0041

ゲストOS-04 (zone):
LDAP slave サーバとして利用
Sun Java SE 1.5.0_12
Sun Java(TM) System Directory Server/5.2_Patch_4 B2005.230.0041

検証内容

Shibboleth IdP の属性解決機能を調査することで、既存システムへの変更点を最小限にしたまま eduPerson 形式での属性を受け渡しを実現し、既存の認証基盤を持つ大学の Federation への参加を容易にすることを目指す。

Mapped AttributeDefinition を用いた静的な属性マッピング

Shibboleth IdP の Mapped AttributeDefinition の機能を用いて、全学IT認証基盤の身分情報を eduPersonAffiliation に変換して提供する。

本学の LDAP における categoryCode は 1-7 が学生，8-9 が教員、10-11 が職員となっている。それぞれ，eduPersonAffiliation の student, faculty, staff にマッピングする。以下 IdP の設定変更内容を示す。

```
% emacs /opt/shibboleth/shibboleth-idp-2.0.0/conf/attribute-resolver.xml
   <!-- eduPersonAffiliation (converted by using static definition) -->
   <resolver:AttributeDefinition id="eduPersonAffiliation" xsi:type="Simple"
    xmlns="urn:mace:shibboleth:2.0:resolver:ad">
     <resolver:Dependency ref="mappedAffiliation" />

     <resolver:AttributeEncoder xsi:type="SAML2String"
      xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
      name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
      friendlyName="eduPersonAffiliation" />
   </resolver:AttributeDefinition>

   <!-- mapping definition from categoryCode to eduPersonAffiliation -->
   <resolver:AttributeDefinition id="mappedAffiliation" xsi:type="Mapped"
    xmlns ="urn:mace:shibboleth:2.0:resolver:ad"
    sourceAttributeID="categoryCode">
     <resolver:Dependency ref="myLDAP" />

     <!-- if the name is not in the expected format, just return it as-is -->
     <DefaultValue>affiliate</DefaultValue>

     <ValueMap>
       <ReturnValue>faculty</ReturnValue>
       <SourceValue>8</SourceValue>
       <SourceValue>9</SourceValue>
     </ValueMap>

     <ValueMap>
       <ReturnValue>student</ReturnValue>
       <SourceValue>1</SourceValue>
       <SourceValue>2</SourceValue>
       <SourceValue>3</SourceValue>
       <SourceValue>4</SourceValue>
       <SourceValue>5</SourceValue>
       <SourceValue>6</SourceValue>
       <SourceValue>7</SourceValue>
     </ValueMap>

     <ValueMap>
       <ReturnValue>staff</ReturnValue>
       <SourceValue>10</SourceValue>
       <SourceValue>11</SourceValue>
     </ValueMap>
   </resolver:AttributeDefinition>
% emacs /opt/shibboleth/shibboleth-idp-2.0.0/conf/attribute-filter.xml
   <AttributeFilterPolicy>
       <PolicyRequirementRule xsi:type="basic:AttributeRequesterString"
        value="https://sp01.auth.cmc.osaka-u.ac.jp/shibboleth"/>
      ...
       <AttributeRule attributeID="eduPersonAffiliation">
           <PermitValueRule xsi:type="basic:ANY" />
       </AttributeRule>
     ....
   </AttributeFilterPolicy>
```

myLDAP DataConnector は Sun Java System Directory Server を指しており、categoryCode に身分コードが格納されている。SP の設定は UPKI から提示されたサンプルと同様で良い。

以上により、eduPerson スキーマを LDAP に導入しなくても、eduPerson 属性を提供できることを確認した。

RelationalDatabase DataConnector を用いた動的な属性マッピング

Shibboleth IdP の RDB DataConnector の機能と、軽量 RDB SQLite を用いて、頻繁に更新される可能性がある属性情報のマッピングを実施する。

現状本学では所属の英語名称を人事データベース上で管理できておらず、正式な英語名称がない部署も存在する。今後これらの情報を順次整備することを 想定し、所属情報は人事データベースとは別の RDB 上に格納して動的にアップデートすることを目指す（最終的には、人事データベース上に反映されるのが望ましい）。以下軽量な RDB SQLite を用いて属性マッピングを実現する手順を示す。なお、SQLite の JDBC ドライバとして、SQLiteJDBC を利用した。

```
### JDBC ドライバのインストール
### Windows, Linux, MacOS X であれば native code を含む
### sqlitejdbc-v053.jar を用いてもよい。
% cp ~/Downloads/sqlitejdbc-v053-pure.jar /opt/apache-tomcat-6.0.18/webapps/shibboleth/WEB-INF/lib/
% mkdir /opt/shibboleth/shibboleth-idp-2.0.0/data/
% cd /opt/shibboleth/shibboleth-idp-2.0.0/data/
### データベースの作成
% sqlite3 departments.db
> create table departments (id integer primary key, dept_id varchar(48), name varchar(255),
name_ja varchar(255));
> insert into departments values (1, '999001', 'Test Department', 'テスト部局');
> .quit
### IdP の設定変更
% emacs /opt/shibboleth-idp-2.0.0/conf/attribute-resolver.xml
   <!-- organizationalUnitCode (the same as uid but only for internal use) -->
   <resolver:AttributeDefinition id="organizationalUnitCode" xsi:type="Simple"
    xmlns="urn:mace:shibboleth:2.0:resolver:ad"
    sourceAttributeID="organizationalUnitCode">
     <resolver:Dependency ref="myLDAP" />
     <resolver:AttributeEncoder xsi:type="SAML2String"
      xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
      name="urn:test:organizationalUnitCode"
      friendlyName="organizationalUnitCode" />
   </resolver:AttributeDefinition>

   <!-- ou (converted from number to name by using SQLite3 database) -->
   <resolver:AttributeDefinition id="organizationalUnit" xsi:type="Simple"
    xmlns="urn:mace:shibboleth:2.0:resolver:ad"
    sourceAttributeID="ou">
     <resolver:Dependency ref="mySQLite" />

     <resolver:AttributeEncoder xsi:type="SAML1String"
      xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
      name="urn:mace:dir:attribute-def:ou" />

     <resolver:AttributeEncoder xsi:type="SAML2String"
      xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
      name="urn:oid:2.5.4.11" friendlyName="ou" />
   </resolver:AttributeDefinition>
...
   <resolver:DataConnector id="mySQLite" xsi:type="RelationalDatabase"
    xmlns="urn:mace:shibboleth:2.0:resolver:dc">
       <resolver:Dependency ref="organizationalUnitCode" />

       <ApplicationManagedConnection jdbcDriver="org.sqlite.JDBC"
           jdbcURL="jdbc:sqlite:/opt/shibboleth/shibboleth-idp-2.0.0/data/departments.db" />

       <QueryTemplate>
           <![CDATA[
               SELECT name, name_ja FROM departments WHERE dept_id = '${organizationalUnitCode.get(0)}'
           ]]>
       </QueryTemplate>

       <Column columnName="name" attributeID="ou" />
       <Column columnName="name_ja" attributeID="upkiOrganizationalUnit" />
   </resolver:DataConnector>
% emacs /opt/shibboleth/shibboleth-idp-2.0.0/conf/attribute-filter.xml
   <AttributeFilterPolicy>
       <PolicyRequirementRule xsi:type="basic:AttributeRequesterString"
        value="https://sp01.auth.cmc.osaka-u.ac.jp/shibboleth"/>

   ...
       <AttributeRule attributeID="organizationalUnit">
           <PermitValueRule xsi:type="basic:ANY" />
       </AttributeRule>
   ...
   </AttributeFilterPolicy>
```

以上で 999001 という所属コードを持った人物の所属名称を受け渡しできる。必要に応じて upkiOrganizationalUnit として日本語名も受け渡しできる。英語名称、日本語名称の取得は適宜データベースを更新すればよい。なお文字コードは UTF-8 に統一している。

Shibboleth のソースコードを見ると、attribute-resolver.xml の QueryTemplate の処理には Velocity Template Language (VTL) が利用されている。また、DataConnector からの戻り値は複数になる可能性があるため（上記の場合 organizationalUnitCode Attribute における myLDAP からの戻り値）、id に指定した文字列（上記の場合 organizationalUnitCode）を変数名とした ArrayList に格納されている。よって，

```
'${organizationalUnitCode.get(0)}'
```

のように ArrayList の 1 番目の要素を Query に指定している。

以上のような設定により、既存のシステムの変更を最小限にしたまま、Shibboleth IdP による連携が実現できると考えられる。

ID のマッピング

SAML2.0 では，属性のマッピング以外に連携サイトに対してどのようにユーザ ID を公開するかを決める ID マッピングの課題も残されている．
Shibboleth 2.0 における ID マッピングについては，京都産業大学の成果「軽量アプリケーションでの Shibboleth 対応手順の調査 > プライバシを考慮したID受け渡し」 で紹介されているのでそちらも参照されたい．

今後の課題

SQLite による属性マッピング機能を導入することで、学内ユーザの属性情報が整備されるまでの間、テンポラリに作成したデータベースを用いて柔軟な運用を行うことができる。ただし、性能面に関しては RDB を用いることによる影響を調査する必要がある。

一方で、将来的には属性情報の整備が重要となってくるため、別途検討を進めている属性連携基盤等により電子職員録システムなどから最新の属性情報を 反映する機能を実装する必要がある。また、最終的には大学に適した属性管理の形態に合わせて、属性の管理場所を変更するなどシステムを順次更新していく必 要がある。そのため、属性連携の仕組みはアジャイルな開発基盤を用いるのが望ましい。

大阪大学サイバーメディアセンターに戻る