２００９年9月２日

# IdP 構築・運用手順書
# (Ver2.0)

(軽井沢セミナー用)

# 1. 概要

本書は IdP の構築手順、および運用方法を説明したものです。

## 1-1. IdP の機能

まず、IdP の動作について簡単に説明します。

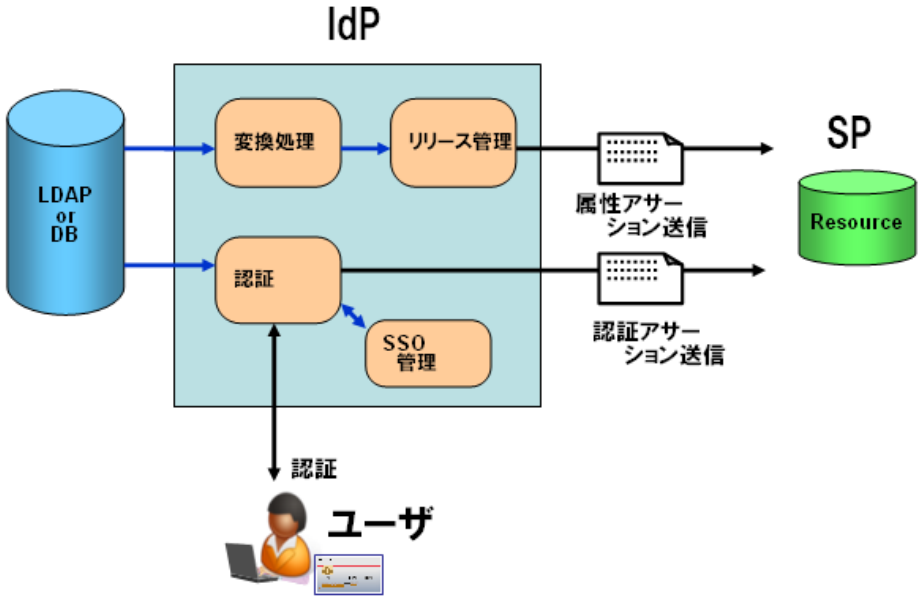



図１　IdP の機能ブロック

図１ IdP の機能ブロックは、IdP の機能を単純化したブロックで示しています。IdP は SP からの要求を受けて、以下の２つの動作を行います。

- ユーザを認証する
- ユーザの属性を安全に SP に送信する

■ 認証

ユーザが SP にアクセスすると、SP は IdP にリダイレクトを行います。IdP はこれを受けてユーザの認証を行います。認証方式としては、ID／パスワード認証や、クライアント証明書による認証等の認証方式が設定可能です。

また、IdP はユーザの Cookie を確認して、既に認証済みである場合は、２回目以降のユーザに対する認証は行わず、シングルサインオン機能を実現します。

■ 属性の安全な送信

SP が要求する属性を、LDAP もしくは DB から取得して、SP に送信します。この際、下記を行います。

- LDAP に格納されている情報を元にして、SP が要求する属性とするために、名称の変更や、ドメインの追加といった変換を行います（図１の変換機能）
- SP に送信して良いかどうか、ポリシーを確認します。各属性が送信可能である場合、SAML2.0 に準拠して安全に属性を送信します（図１のリリース機能）

以下の章では、IdP の構築手順を示すとともに、上記機能の設定方法、および、これらの機能を用いて IdP を運用するための方法について説明します。

## 1-2. 構築方式について

IdP の構築にあたっては、以下の２つの方式のいずれかを採択することができます。

■ VMWare イメージを利用した構築

NII が提供する“OS から shibboleth(IdP)までインストールされた”システムを利用する方式です。貴学では、貴学のサーバに VMWareServer をインストールし、その上にこのシステムイメージを稼動することで利用できます。

■ 自分で IdP をインストールする場合の構築手順

貴学にて、貴学のサーバに OS から shibboleth(IdP)までインストール・設定を行い、構築する方式です。

本書では、上記２方式の手順を明記するとともに、構築後の基本的な設定方法や運用方法も記載しています。

# 2. インストール

## 2-1.　VMWare イメージを利用した構築手順

本セミナーでは VMWare イメージは利用しません。

## 2-2.　貴学にて IdP をインストールする場合の構築手順

### 2-2-1. shibboleth (IdP version2.0)の動作要件

- Apache HTTP Server 2.2 以上
- Apache Tomcat 5.5.25 以上
- Java 5 以上

（ただし、CentOS に付属する Gnu Java は利用できません。 Sun の Java を利用してください。）

### 2-2-2. OS をインストールする

① OS での設定

- OS：CentOS 5.3

  インストーラでインストールするもの。

  Web サーバー (HTTP のみ)

  Open Ldap

  その他のパッケージは必要に応じてインストールしてください。

  ただし、Java 開発と Tomcat は後の手順で別にインストールします。

- ネットワーク設定

  環境に合わせ、ホスト名・ネットワーク・セキュリティを設定して下さい。

  SP では shibd サービスが通信を行います。

② DNS へ登録する

新しいホスト名と IP アドレスを DNS に登録してください。

③ 時刻同期を設定する

ntp サービスを用い、貴学環境の ntp サーバと時刻同期をしてください。

※Shibboleth では、通信するサーバ間の時刻のずれが約５分を越えるとエラーになります。

## 2-2-3. jdk6、tomcat6 をインストールする

① tomcat5-5.5.25 以前のバージョンの削除

tomcat5-5.5.25 以前のバージョンが入っている場合は、削除してください。

＜セミナー注： tomcat はインストールされていません。＞

② jdk 6.0 のインストール

http://java.sun.com/javase/downloads/index.jsp よりダウンロードした最新版の jdk（2009/8/28 時点での最新版は jdk-6u16-linux-i586-rpm.bin、OS が 64bit 版の場合は jdk-6u16-linux-x64-rpm.bin）を適当なフォルダに置いて、以下のコマンドを実行してください。

```
chmod a+x jdk-6u16-linux-i586-rpm.bin
./jdk-6u16-linux-i586-rpm.bin
```

③ tomcat6.0 のインストール

http://tomcat.apache.org/download-60.cgi よりダウンロードした tomcat6 の最新版（2009/8/28 時点で apache-tomcat-6.0.20.tar.gz）を /usr/java にを置いて、以下のコマンドを実行してください。

```
cd /usr/java
  tar zxvf apache-tomcat-6.0.20.tar.gz
  ln -s apache-tomcat-6.0.20 /usr/java/tomcat
```

jsvc などを用いて、自動起動させると便利です。

ソースファイルを展開し、make で jsvc を作成した後、 $CATALINA_HOME/bin にコピーします。その後、tomcat 起動用スクリプトを/etc/rc.d/init.d にコピーします。

```
  cd /usr/java/tomcat/bin
  tar xzvf jsvc.tar.gz
  cd jsvc-src
  chmod a+x configure
  ./configure –with-java=/usr/java/jdk1.6.0_16
  make
  cp jsvc ..
  cp native/Tomcat5.sh /etc/rc.d/init.d/tomcat
```

/etc/rc.d/init.d/tomcat の先頭にコメントを追加することにより chkconfig コマンドが利用できます。

```
# chkconfig: - 86 15
# description: Tomcat
# processname: tomcat
```

また、/etc/rc.d/init.d/tomcat ファイル中の以下の環境変数も変更が必要です。

JAVA_HOME
CATALINA_HOME
DAEMON_HOME
CATALINA_BASE

設定例

```
JAVA_HOME=/usr/java/default
CATALINA_HOME=/usr/local/tomcat
DAEMON_HOME=$CATALINA_HOME
CATALINA_BASE=$CATALINA_HOME
```

＜セミナー注：CATALINA_HOME は/usr/java/tomcat として下さい。
for multi instances セクションの$CATALINA_BASE も設定します。
さらに、$DAEMON_HOME/src/native/unix/jsvc を、
$DAEMON_HOME/bin/jsvc に変更して下さい。（２箇所）＞

tomcat を実行するユーザ“tomcat”を作成した場合には以下も設定します。

```
TOMCAT_USER=tomcat
```

＜セミナー注：tomcat ユーザを作成します。
useradd –d /var/empty/tomcat –s /sbin/nologin tomcat
chown –R tomcat:tomcat /usr/java/apache-tomcat-6.0.20＞

他に、”tomcat”ユーザがログファイルを出力できるよう、ディレクトリの所有者を変更します。(シンボリックリンク先を変更するため最後の“/”が必要です)

```
chwon –R tomcat:  /usr/java/tomcat/
```

自動起動の設定コマンドを実行します。(オプションは マイナス‘ー‘ が２つ必要です)

```
chkconfig --add tomcat
chkconfig --level 345 tomcat on
```

④　/etc/profile に下記を追加してください。

```
JAVA_HOME=/usr/java/default
MANPATH=$MANPATH:$JAVA_HOME/man
CATALINA_HOME=/usr/java/tomcat
TOMCAT_HOME=$CATALINA_HOME
PATH=$JAVA_HOME/bin:$CATALINA_HOME/bin:$PATH
export PATH JAVA_HOME CATALINA_HOME
```

⑤　httpd の設定

/etc/httpd/conf/httpd.conf の修正

```
（省略）
ServerName upki-test-idpvm.nii.ac.jp:80  ←ホスト名
（省略）
```

/etc/httpd/conf.d/ssl.conf の修正

```
（省略）
ServerName upki-test-idpvm.nii.ac.jp:443  ←ホスト名
（省略）
<VirtualHost _default_:443>
（省略）
ProxyPass /idp/ ajp://localhost:8009/idp/    ←追加
（省略）
</VirtualHost>
```

⑥　/usr/java/tomcat/conf/server.xml の修正

・　必要に応じて Connector port="8080"をコメントアウトしてください。

```
<!--
    <Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
-->
```

・　Connector port="8009"に以下のように追加してください。

```
<Connector port="8009"
    protocol="AJP/1.3"  redirectPort="8443" />
↓
<Connector port="8009"
    protocol="AJP/1.3"  redirectPort="8443"  enableLookups="false"
    request.tomcatAuthentication="false" address="127.0.0.1" />
```

### 2-2-4. shibboleth のインストール

① shibboleth-idp-bin.zip のダウンロード

http://shibboleth.internet2.edu/downloads/shibboleth/idp/ から最新版ディレクトリ(2009/8/28 時点で 2.1.3)に移動し、shibboleth-identityprovider-2.1.2-bin.zip をダウンロードします。

② インストール

shibboleth-identityprovider-2.1.3-bin.zip を展開し、インストールします。

```
# unzip shibboleth-identityprovider-2.1.3-bin.zip
# cd shibboleth-identityprovider-2.1.3
# bash install.sh
```

いくつかの質問に答えながら、インストールを行います。

途中で入力するパスワードはデフォルトで作成されるキーストアファイルのパスワードとなります。

③ Java の設定

shibboleth-identityprovider-2.1.3-bin.zip を展開した shibboleth-identityprovider-2.1.3/lib/ にある shibboleth-jce-1.1.0.jar を $JAVA_HOME/jre/lib/ext にコピーします。

さらに、$JAVA_HOME/jre/lib/security/java.security ファイルに以下を追加します。

```
security.provider.9=edu.internet2.middleware.shibboleth.DelegateToApplicationProvider
```

番号:9 は既に記載されている番号に合わせて順番にして下さい。

④ Tomcat の設定

shibboleth-identityprovider-2.1.3-bin.zip を展開した
shibboleth-identityprovider-2.1.3/endorsed にある 5 つの jar ファイルを、$CATALINA_HOME/endorsed ディレクトリを作成してそこへコピーします。
(Tomcat5.5 の場合は $CATALINA_HOME/common/endorsed ディレクトリ)

resolver-2.9.1.jar
serializer-2.9.1.jar
xalan-2.7.1.jar
xml-apis-2.9.1.jar
xercesImpl-2.9.1.jar

これらの jar ファイルが有効となるよう、Tomcat 起動スクリプトを変更します。
/etc/rc.d/init.d/tomcat を作成していた場合は、以下の追加となります。

```
CATALINA_OPTS="-Djava.endorsed.dirs=${CATALINA_HOME}/endorsed"
```

上記例ではデフォルトで記述されている
“-Djava.library.path=/home/jfclere/jakarta-tomcat-connectors/jni/native/.libs”を使用していないため記述を削除しています。

tomcat を、”tomcat”ユーザで実行する場合は、ログファイルを出力できるようディレクトリの所有者を変更します。

```
chown –R tomcat: /opt/shibboleth-idp/logs
```

⑤ idp.war の配置

/opt/shibboleth-idp/war/idp.war ファイルを、${CATALINA_HOME}/webapps ディレクトリにコピーします。

## 2-2-5. サービスを起動・停止する方法

httpd の起動方法

service httpd start

tomcat の起動方法

service tomcat start　　(jsvc を利用した場合)

httpd の停止方法

service httpd stop

tomcat の停止方法

service tomcat stop　　(jsvc を利用した場合)

# 3. 運用・設定・カスタマイズ

## 3-1. 設定と接続テスト

### 3-1-1. openldap の設定（貴学で IdP をインストールする場合のみ）

① 追加のスキーマファイル

eduperson スキーマを以下からダウンロードしたものを/etc/openldap/schema/ に置いてください。

http://middleware.internet2.edu/eduperson/

「eduPerson (200806)」から「OpenLDAP eduPerson (updated to 200806)」を クリックしてください。

OpenLDAP 用の eduPerson(200806) を取得します。

/etc/openldap/schema/eduperson-200806.schema という名称にしてください。

② ldap のデフォルト設定

/etc/openldap/slapd.conf を変更します。

```
（省略）
include  /etc/openldap/schema/eduperson-200806.schema  ←追加

suffix   "o=Test Organization,dc=ac,c=JP"                ←suffix
rootdn   "cn=olmgr,o=Test Organization,dc=ac,c=JP "   ←rootdn
rootpw   {CRYPT}ijFYNcSNctBYg                ←暗号化※したものを記載
```

ここで設定したパスワードは IdP の設定ファイルにも記述します。「3-2-4. 属性のリリース方法」をご確認ください。

※暗号化の例 ： 「csildap」というパスワードを暗号化する

```
#slappasswd -h {crypt} -s csildap
{CRYPT}ijFYNcSNctBYg        ←これを slapd.conf の rootpw に記載
```

openldap を起動します。

```
service ldap start
```

### 3-1-2. ldap のテストデータ作成

以下のサンプルを基に、テスト用データを作成し、ldap へ登録します。

Shibboleth を利用した ID/パスワードでの認証に使用される ID は uid 、パスワードは userPassword になります。

・ test.ldif ファイル作成

```
dn: o=Test Organization,dc=ac,c=JP
objectClass: organization
o: Test Organization

dn: ou=Test Unit1,o=Test Organization,dc=ac,c=JP
objectClass: organizationalUnit
ou: Test Unit1

dn: ou=Test Unit2,o=Test Organization,dc=ac,c=JP
objectClass: organizationalUnit
ou: Test Unit2

dn: ou=Test Unit3,o=Test Organization,dc=ac,c=JP
objectClass: organizationalUnit
ou: Test Unit3

dn: uid=test001,ou=Test Unit1,o=Test Organization,dc=ac,c=JP
objectClass: eduPerson
objectClass: inetOrgPerson
uid: test001
o;lang-ja: テスト001_大学
ou: Test Unit1
ou;lang-ja: テスト001_学部1
sn: test001_sn
sn;lang-ja: テスト001_sn
cn: test001_cn
userPassword: test001
givenName: test001_givenname
givenName;lang-ja: テスト001_givenname
displayName: test001_displayname
displayName;lang-ja: テスト001_displayname
mail: test001_email@nii.ac.jp
eduPersonAffiliation: member

dn: uid=test002,ou=Test Unit2,o=Test Organization,dc=ac,c=JP
objectClass: eduPerson
objectClass: inetOrgPerson
uid: test002
o;lang-ja: テスト002_大学
ou: Test Unit2
ou;lang-ja: テスト002_学部2
sn: test002_sn
sn;lang-ja: テスト002_sn
cn: test002_cn
userPassword: test002
givenName: test002_givenname
givenName;lang-ja: テスト002_givenname
displayName: test002_displayname
displayName;lang-ja: テスト002_displayname
mail: test002_email@nii.ac.jp
eduPersonAffiliation: faculty
```

```
dn: uid=test003,ou=Test Unit3,o=Test Organization,dc=ac,c=JP
objectClass: eduPerson
objectClass: inetOrgPerson
uid: test003
o;lang-ja: テスト 003_大学
ou: Test Unit3
ou;lang-ja: テスト 003_学部 3
sn: test003_sn
sn;lang-ja: テスト 003_sn
cn: test003_cn
userPassword: test003
givenName: test003_givenname
givenName;lang-ja: テスト 003_givenname
displayName: test003_displayname
displayName;lang-ja: テスト 003_displayname
mail: test003_email@nii.ac.jp
eduPersonAffiliation: student
```

・ ldap への登録

```
#ldapadd -x -h localhost -D "cn=olmgr,o=Test Organization,dc=ac,c=JP" -w csildap  -f test.ldif
```

### 3-1-3. shibboleth の設定

デフォルトでは shibboleth は /opt/shibboleth-idp ディレクトリにインストールされます。変更する各設定ファイルは /opt/shibboleth-idp/conf にあります。

・ relying-party.xml ファイルの確認

ホスト名を修正します。(VM イメージ利用の場合)

ホスト名が正しいことを確認します。(貴学でインストールする場合)

```
    <!-- ======================================== -->
    <!--      Relying Party Configurations        -->
    <!-- ======================================== -->
    <AnonymousRelyingParty provider="https://upki-test-idpvm.nii.ac.jp/idp/shibboleth" />
                                                                        ←ホスト名
    <DefaultRelyingParty provider="https://upki-test-idpvm.nii.ac.jp/idp/shibboleth"
                                                                        ←ホスト名
                         defaultSigningCredentialRef="IdPCredential">
```

UPKI Federation のメタデータを自動的にダウンロードする設定をします。

- 運用フェデレーション用メタデータ

  http://upki-repo.nii.ac.jp/Metadata/upki-fed-metadata-signed.xml

- テストフェデレーション用メタデータ

  http://upki-repo.nii.ac.jp/Metadata/upki-test-fed-metadata-signed.xml

```
<!-- ========================================= -->
<!--      Metadata Configuration               -->
<!-- ========================================= -->
<!-- MetadataProvider the combining other MetadataProviders -->
<MetadataProvider id="ShibbolethMetadata" xsi:type="ChainingMetadataProvider"
                  xmlns="urn:mace:shibboleth:2.0:metadata">
<!-- Load the IdP's own metadata.  This is necessary for artifact support. -->
  <MetadataProvider id="IdPMD" xsi:type="ResourceBackedMetadataProvider"
                  xmlns="urn:mace:shibboleth:2.0:metadata" >
    <MetadataResource xsi:type="resource:FilesystemResource"
                  file="/opt/shibboleth-idp/metadata/idp-metadata.xml" />
  </MetadataProvider>

<!-- Example metadata provider. -->
  <!-- Reads metadata from a URL and store a backup copy on the file system. -->
  <!-- Validates the signature of the metadata and filters out all by SP entities in order to save
memory -->
  <!-- To use: fill in 'metadataURL' and 'backingFile' properties on MetadataResource element -->
  <!-- --> ←コメントアウトを閉じます
  <MetadataProvider id="URLMD" xsi:type="FileBackedHTTPMetadataProvider"
                  xmlns="urn:mace:shibboleth:2.0:metadata"
          metadataURL="https://upki-repo.nii.ac.jp/Metadata/upki-test-fed-metadata-signed.xml"
                  backingFile="/opt/shibboleth-idp/metadata/upki-metadata-backing.xml">
    <MetadataFilter xsi:type="ChainingFilter" xmlns="urn:mace:shibboleth:2.0:metadata">
    <!-- ←現段階でメタデータが対応していないのでコメントアウトします
    <MetadataFilter    xsi:type="RequiredValidUntil"    xmlns="urn:mace:shibboleth:2.0:metadata"
maxValidityInterval="604800" />
    -->
    <MetadataFilter xsi:type="SignatureValidation" xmlns="urn:mace:shibboleth:2.0:metadata"
                                trustEngineRef="shibboleth.MetadataTrustEngine"
                                requireSignedMetadata="true" />
      <MetadataFilter xsi:type="EntityRoleWhiteList" xmlns="urn:mace:shibboleth:2.0:metadata">
        <RetainedRole>samlmd:SPSSODescriptor</RetainedRole>
      </MetadataFilter>
    </MetadataFilter>
  </MetadataProvider>
  <!-- --> ←コメントアウト
```

メタデータを検証する設定をします。IdP 申請時に入手した証明書を任意のディレクトリに置き、そのパスを設定して下さい。

```
    <!-- ========================================= -->
    <!--     Security Configurations               -->
    <!-- ========================================= -->
    <security:Credential id="IdPCredential" xsi:type="security:X509Filesystem">
        <security:PrivateKey>/opt/shibboleth-idp/credentials/idp.key</security:PrivateKey>
        <security:Certificate>/opt/shibboleth-idp/credentials/idp.crt</security:Certificate>
    </security:Credential>

    <!-- --> ←コメントアウトを閉じます
    <!-- Trust engine used to evaluate the signature on loaded metadata. -->
    <security:TrustEngine                                    id="shibboleth.MetadataTrustEngine"
xsi:type="security:StaticExplicitKeySignature">
        <security:Credential id="MyFederation1Credentials" xsi:type="security:X509Filesystem">

<security:Certificate>/opt/shibboleth-idp/credentials/upki-fed-signer-ca-20090204.cer</security
:Certificate>
        </security:Credential>
    </security:TrustEngine>
    <!-- --> ←コメントアウト
```

・handler.xml ファイルの変更（貴学で IdP をインストールする場合のみ）

LDAP のデータを用いた ID/パスワード認証のために handler.xml ファイルを変更します。

```
    <!-- Login Handlers -->
    <!--               ←コメントアウトします
    <LoginHandler xsi:type="RemoteUser">
        <AuthenticationMethod>
              urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified</AuthenticationMethod>
    </LoginHandler>
    -->

    <!--  Username/password login handler -->    ←こちらを有効にします
    <LoginHandler xsi:type="UsernamePassword"
                  jaasConfigurationLocation="file:///opt/shibboleth-idp/conf/login.config">
        <AuthenticationMethod>
               urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
        </AuthenticationMethod>
    </LoginHandler>
```

・login.config ファイルの変更（貴学で IdP をインストールする場合のみ）

LDAP のデータを用いた ID/パスワード認証のために login.config ファイルを変更します。 各設定値は、ldap のデフォルト設定と同じ値とします。

```
    ShibUserPassAuth {


// Example LDAP authentication
// See: https://spaces.internet2.edu/display/SHIB2/IdPAuthUserPass
/*  */      ←コメントを閉じて有効にします
   edu.vt.middleware.ldap.jaas.LdapLoginModule required
      host="localhost"
      base="o=Test Organization,dc=ac,c=JP"
      ssl="false"
      userField="uid"
      subtreeSearch="true"
      ;       ← 最後の ; は必ず記述
/*  */     ←コメントを閉じます
```

・attribute-resolver.xml ファイルの変更

テンプレート（attribute-resolver-template2009xxxx.xml）を、学術認証フェデレーションのリポジトリ（http://upki-repo.nii.ac.jp/Template）からダウンロードします。

デフォルトの /opt/shibboleth-idp/conf/attribute-resolver.xml をテンプレートで差し替え、修正して下さい。コメントアウトされている定義を有効にしたり、不足している定義を追加したりする必要がありますので、次ページ以降を参照して設定してください。

表　設定が必要な項目リスト

| Id | 設定 |
|---|---|
| mail | コメントアウトを外して有効にする |
| sn | コメントアウトを外して有効にする |
| o | コメントアウトを外して有効にする |
| ou | コメントアウトを外して有効にする<br>要素の追加・変更が必要 |
| givenName | コメントアウトを外して有効にする |
| displayName | コメントアウトを外して有効にする |
| eduPersonAffiliation | コメントアウトを外して有効にする |
| eduPersonPrincipalName | コメントアウトを外して有効にする<br>要素の変更が必要 |
| eduPersonEntitlement | コメントアウトを外して有効にする |
| eduPersonScopedAffiliation | コメントアウトを外して有効にする<br>要素の追加・変更が必要 |
| eduPersonTargetID | コメントアウトを外して有効にする |
| jasn | コメントアウトを外して有効にする |
| jaGivenName | コメントアウトを外して有効にする |
| jaDisplayName | コメントアウトを外して有効にする |
| jao | コメントアウトを外して有効にする |
| jaou | コメントアウトを外して有効にする |

・ email

```
    ※「email」を検索し、場所を特定してください。


--->      ←コメント終了を追加して、以下の“email”の値を公開します
    <resolver:AttributeDefinition id="email" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="mail">
        <resolver:Dependency ref="myLDAP" />

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

        <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:0.9.2342.19200300.100.1.3" friendlyName="mail" />
    </resolver:AttributeDefinition>
<!--      ←コメント開始を追加します
```

・ surname

```
    ※「surname」を検索し、場所を特定してください。


--->      ←コメント終了を追加して、以下の“surname”の値を公開します
    <resolver:AttributeDefinition id="surname" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="sn">
        <resolver:Dependency ref="myLDAP" />

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

        <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:2.5.4.4" friendlyName="sn" />
    </resolver:AttributeDefinition>
<!--      ←コメント開始を追加します
```

・ organizationName

```
    ※「organizationName」を検索し、場所を特定してください。


 -->      ←コメント終了を追加して、以下の“organizationName”の値を公開します
  <resolver:AttributeDefinition id="organizationName" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="o">
        <resolver:Dependency ref="staticOrganization" />

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

        <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:2.5.4.10" friendlyName="o" />
    </resolver:AttributeDefinition>
<!--      ←コメント開始を追加して、以下をコメントアウトします
```

・ organizationUnit

**※「organizationUnit」を検索し、場所を特定してください。**

```
->      ←コメント終了を追加して、“organizationalUnit”の値を公開します
<resolver:AttributeDefinition id="organizationalUnit" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="ou">
        <resolver:Dependency ref="myLDAP" />

        <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>
<!--      ←コメント開始を追加して、以下をコメントアウトします
```

・ givenName

**※「givenName」を検索し、場所を特定してください。**

```
--->      ←コメント終了を追加して、以下の“givenName”の値を公開します
    <resolver:AttributeDefinition id="givenName" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="givenName">
        <resolver:Dependency ref="myLDAP" />

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

        <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:2.5.4.42" friendlyName="givenName" />
    </resolver:AttributeDefinition>
<!--      ←コメント開始を追加します
```

・ eduPersonAffiliation

**※「eduPersonAffiliation」を検索し、場所を特定してください。**

```
<!-- Schema: eduPerson attributes -->
--->      ←コメント終了を追加して、以下の“givenName”の値を公開します
<resolver:AttributeDefinition id="eduPersonAffiliation" xsi:type="Simple"
                                                        xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="eduPersonAffiliation">
        <resolver:Dependency ref="myLDAP" />

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

        <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>
<!--      ←コメント開始を追加して、以下をコメントアウトします
```

・ eduPersonPrincipalName

**※「eduPersonPrincipalName」を検索し、場所を特定してください。**

```
<!-- -->  ←← コメント終了を追加して、“eduPersonEntitlement”を有効とします
<resolver:AttributeDefinition id="eduPersonPrincipalName" xsi:type="Scoped"
                      xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        scope="***.ac.jp" sourceAttributeID="uid">      ←Scope を変更します
        <resolver:Dependency ref="myLDAP" />

        <resolver:AttributeEncoder xsi:type="SAML1ScopedString" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:mace:dir:attribute-def:eduPersonPrincipalName" />

        <resolver:AttributeEncoder xsi:type="SAML2ScopedString" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6" friendlyName="eduPersonPrincipalName" />
    </resolver:AttributeDefinition>
<!--      ←コメント開始を追加して、以下をコメントアウトします
```

・ eduPersonEntitlement

**※「eduPersonPrincipalName」を検索し、場所を特定してください。**

```
--->  ←コメント終了を追加して、以下を有効とします
<resolver:AttributeDefinition id="eduPersonEntitlement" xsi:type="Simple"
 xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="eduPersonEntitlement">
        <resolver:Dependency ref="staticEntitlement" />

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

        <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:1.3.6.1.4.1.5923.1.1.1.7" friendlyName="eduPersonEntitlement" />
    </resolver:AttributeDefinition>
<!--  ←コメント開始を追加して、以下をコメントアウトします
```

・ eduPersonScopedAffiliation

**※「eduPersonScopedAffiliation」を検索し、場所を特定してください。**

```
--->  ←コメント終了を追加して、以下を有効とします
<resolver:AttributeDefinition id="eduPersonScopedAffiliation" xsi:type="Scoped"
 xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        scope="***.ac.jp" sourceAttributeID="eduPersonAffiliation">      ←Scope を変更します
        <resolver:Dependency ref="myLDAP" />

        <resolver:AttributeEncoder xsi:type="SAML1ScopedString"
 xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:mace:dir:attribute-def:eduPersonScopedAffiliation" />

        <resolver:AttributeEncoder xsi:type="SAML2ScopedString"
 xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:1.3.6.1.4.1.5923.1.1.1.9" friendlyName="eduPersonScopedAffiliation" />
</resolver:AttributeDefinition>
<!--  ←コメント開始を追加して、以下をコメントアウトします
```

・ eduPersonTargetedID

**※「eduPersonTargetdID」を検索し、場所を特定してください。**

```
--->  ←コメント終了を追加して、以下を有効とします
<resolver:AttributeDefinition id="eduPersonTargetedID" xsi:type="SAML2NameID"
 xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        nameIdFormat="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent"
        sourceAttributeID="computedID">
        <resolver:Dependency ref="computedID" />

        <resolver:AttributeEncoder xsi:type="SAML1XMLObject"
xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
                name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10" />

        <resolver:AttributeEncoder xsi:type="SAML2XMLObject"
 xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
                name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10" friendlyName="eduPersonTargetedID" />
</resolver:AttributeDefinition>
<!--  ←コメント開始を追加して、以下をコメントアウトします
```

・ jasurname、jagivenName, jadisplayName, jaorgnizationName,

**※下の項目を検索し、コメントアウトを外してください。**

```
   <!-- ======================================== -->
   <!--      ja Definition                       -->
   <!-- ======================================== -->

   <!-- Definition of jaSurName -->
   <resolver:AttributeDefinition id="jasurname" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
       sourceAttributeID="sn;lang-ja">
       <resolver:Dependency ref="myLDAP" />

       <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
           name="urn:oid:1.3.6.1.4.1.32264.1.1.1" friendlyName="jasn" />
   </resolver:AttributeDefinition>
   <!-- Definition of jaGivenName -->
   <resolver:AttributeDefinition id="jagivenName" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
       sourceAttributeID="givenName;lang-ja">
       <resolver:Dependency ref="myLDAP" />

       <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
           name="urn:oid:1.3.6.1.4.1.32264.1.1.2" friendlyName="jaGivenName" />
   </resolver:AttributeDefinition>
   <!-- Definition of jaDisplayName -->
   <resolver:AttributeDefinition id="jaDisplayName" xsi:type="Simple"
xmlns="urn:mace:shibboleth:2.0:resolver:ad"
       sourceAttributeID="displayName;lang-ja">
       <resolver:Dependency ref="myLDAP" />

       <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
           name="urn:oid:1.3.6.1.4.1.32264.1.1.3" friendlyName="jaDisplayName" />
   </resolver:AttributeDefinition>
   <!-- Definition of jaOrganizationName -->
   <resolver:AttributeDefinition id="jaorganizationName" xsi:type="Simple"
xmlns="urn:mace:shibboleth:2.0:resolver:ad"
       sourceAttributeID="jao">
       <resolver:Dependency ref="staticjao" />

       <resolver:AttributeEncoder xsi:type="SAML2String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
           name="urn:oid:1.3.6.1.4.1.32264.1.1.4" friendlyName="jao" />
   </resolver:AttributeDefinition>
```

・attribute-filter.xml ファイルの変更

テンプレート（attribute-filter-template2009xxxx.xml）を、学術認証フェデレーションのリポジトリ（http://upki-repo.nii.ac.jp/Template）からダウンロードします。
テンプレートは、運用フェデレーション用とテストフェデレーション用の２種類がありますので、ご注意ください。
＜セミナー注： セミナーではテストフェデレーション用を利用下さい。
また、upki-test-sp00 を seminar-sp00 に、upki-test-sp02 を seminar-sp09 にそれぞれ変更します。＞

## 3-1-4. サーバ証明書を申請、登録する

① サーバ証明書申請

「UPKI オープンドメイン証明書自動発行検証プロジェクト」のサイトをご参照ください。
https://upki-portal.nii.ac.jp/docs/odcert
を参考に、サーバ証明書を申請します。
証明書の交付までには数日を要するので、お早めに申請してください。
＜セミナー注： セミナーでは上記の代わりにローカル認証局から発行した
サーバ証明書を利用します。サーバ証明書は実習時に配布します。＞

② 入手したサーバ証明書をもとに、以下のファイルに設定してください。

■ /etc/httpd/conf.d/ssl.conf

```
（省略）
SSLCertificateFile /etc/pki/tls/certs/server.crt    ←サーバ証明書の格納先
（省略）
SSLCertificateKeyFile /etc/pki/tls/private/server.key  ←秘密鍵の格納先
（省略）
#SSLCACertificateFile /etc/pki/tls/certs/ca-bundle.crt ←コメントアウト
SSLCACertificatePath /opt/shibboleth-idp/credentials/CA ←中間 CA 証明書の格納先
```

/opt/shibboleth-idp/credentials/CA ディレクトリが無い場合は作成してください。
このディレクトリには、ファイル名をハッシュ値とした中間 CA 証明書を配置します。
詳しくは「UPKI オープンドメイン証明書自動発行検証プロジェクト ― 利用の手引き（https://upki-portal.nii.ac.jp/docs/odcert/howto）を参照してください。

■ /opt/shibboleth-idp/conf/relying-party.xml

```
（中略）
<security:Credential id="IdPCredential" xsi:type="security:X509Filesystem">
  <security:PrivateKey>/etc/pki/tls/private/server.key ←ssl.confと同一のものを
  </security:PrivateKey>                                    左記のパスにも格納
  <security:Certificate>/etc/pki/tls/certs/server.crt ←ssl.confと同一のものを
  </security:Certificate>                                   左記のパスにも格納
</security:Credential>
（中略）
```

③ メタデータテンプレート取得

学術認証フェデレーションのリポジトリ（https://upki-repo.nii.ac.jp/Template）から、IdP 用メタデータテンプレートをダウンロードし、必要な項目を変更します。

ダウンロードしたメタデータテンプレートを下記のように変更してください。

```
（中略）
<!-- IdP: Your Organization Name, Your IdP Site URL, Date -->

    <EntityDescriptor entityID="https://upki-test-idpvm.nii.ac.jp/idp/shibboleth"> ←ホスト名

<IDPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:1.1:protocol
                                                                    urn:mace:shibboleth:1.0

urn:oasis:names:tc:SAML:2.0:protocol">
       <Extensions>
          <shibmd:Scope regexp="false">YourScope.ac.jp</shibmd:Scope>
       </Extensions>
          <KeyDescriptor>
             <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                <ds:X509Data>
                   <ds:X509Certificate>
MIIFIjCCBAqgAwIBAgIERcdlfzANBgkqhkiG9w0BAQUFADB3MQswCQYDVQQGEwJK
UDEQMA4GA1UEBxMHQWNhZGVtZTEqMCgGA1UEChMhTmF0aW9uYWwgSW5zdGl0dXRl
IG9mIEluZm9ybWF0aWNzMQ0wCwYDVQQLEwRVUEtJMRswGQYDVQQLExJOSUkgT3Bl
biBEb21haW4gQ0EwHhcNMDkwMTI4MDUwNzU0WhcNMTAwNjMwMTQ1OTU5WjCBizEL
MAkGA1UEBhMCSlAxEDAOBgNVBAcTB0FjYWRlbWUxKjAoBgNVBAoTIU5hdGlvbmFs
IEluc3RpdHV0ZSBvZiBJbmZvcm1hdGljczEmMCQGA1UECxMdSW5mb3JtYXRpb24g
VGVjaG5vbG9neSBDZW50ZXIxFjAUBgNVBAMTDWlkcC5uaWkuYWMuanAwgZ8wDQYJ
                                    ********************
                                    ********************
kXqlMB0GA1UdDgQWBBREGVw1FHm9O0DToStMhpa0DMKcITANBgkqhkiG9w0BAQUF
AAOCAQEAV1t3KEunnhNThspCiiJQyVAk7MHztneZZ6BCaSvWAhq0mF9hHhIpZFut
4VwalcATGmiPKoj4bDxOrUPKYmxVK9elvvWEaT4yPnluuUv5taviwufgsgnfDCIQ
di/0RpGBDzxbHz6DPnljhFykXxnIjg7gmGeKIPSlwCCZKZkazkVWq2Gj7MfsADow
ZAJrq12ERt/yqfvytNSI3iFSONxn3+GsQcjKDhUczi3mHGu21xKLAK2bWRdHRMYB
GjtOHyJRosJOSZY79puyOLEKKTs4lRfce9dWzTfMGPna53hFGXlavOWejlh5+v16
srflHJUIcX/aCPSTd+LJw1IEP6ivjw==
                   </ds:X509Certificate>
                </ds:X509Data>
             </ds:KeyInfo>
          </KeyDescriptor>

          <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
          <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
          <SingleSignOnService Binding="urn:mace:shibboleth:1.0:profiles:AuthnRequest"
                               Location="https://upki-test-idpvm.nii.ac.jp/idp/profile/Shibboleth/SSO" />
                                                                              ←ホスト名
          <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                               Location="https://upki-test-idpvm.nii.ac.jp/idp/profile/SAML2/POST/SSO" />
                                                                              ←ホスト名
          <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
                               Location="https://upki-test-idpvm.nii.ac.jp/idp/profile/SAML2/Redirect/SSO" />
                                                                              ←ホスト名
       </IDPSSODescriptor>
```

入手した証明書に変更

```
        <AttributeAuthorityDescriptor
protocolSupportEnumeration="urn:oasis:names:tc:SAML:1.1:protocol urn:oasis:names:tc:SAML:2.0:protocol">
            <Extensions>
                <shibmd:Scope regexp="false">nii.ac.jp</shibmd:Scope>      ←スコープ
            </Extensions>

            <KeyDescriptor>
                <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                    <ds:X509Data>
                        <ds:X509Certificate>
MIIFIjCCBAqgAwIBAgIERcdlfzANBgkqhkiG9w0BAQUFADB3MQswCQYDVQQGEwJK
UDEQMA4GA1UEBxMHQWNhZGVtZTEqMCgGA1UEChMhTmF0aW9uYWwgSW5zdGl0dXRl
IG9mIEluZm9ybWF0aWNzMQ0wCwYDVQQLEwRVUEtJMRswGQYDVQQLExJOSUkgT3Bl
biBEb21haW4gQ0EwHhcNMDkwMTI4MDUwNzU0WhcNMTAwNjMwMTQ1OTU5WjCBizEL
MAkGA1UEBhMCSlAxEDAOBgNVBAcTB0FjYWRlbWUxKjAoBgNVBAoTIU5hdGlvbmFs
IEluc3RpdHV0ZSBvZiBJbmZvcm1hdGljczEmMCQGA1UECxMdSW5mb3JtYXRpb24g
VGVjaG5vbG9neSBDZW50ZXIxFjAUBgNVBAMTDWlkcC5uaWkuYWMuanAwgZ8wDQYJ
                                    ********************
                                    ********************
kXqlMB0GA1UdDgQWBBREGVw1FHm9O0DToStMhpa0DMKcITANBgkqhkiG9w0BAQUF
AAOCAQEAV1t3KEunnhNThspCiiJQyVAk7MHztneZZ6BCaSvWAhqOmF9hHhIpZFut
4VwalcATGmiPKoj4bDxOrUPKYmxVK9elvvWEaT4yPnluuUv5taviwufgsgnfDCIQ
di/ORpGBDzxbHz6DPnljhFykXxnIjg7gmGeKIPSlwCCZKZkazkVWq2Gj7MfsADow
ZAJrq12ERt/yqfvytNSI3iFSONxn3+GsQcjKDhUczi3mHGu21xKLAK2bWRdHRMYB
Gjt0HyJRosJOSZY79puyOLEKKTs4lRfce9dWzTfMGPna53hFGXlavOWejlh5+v16
srflHJUIcX/aCPSTd+LJw1IEP6ivjw==
                        </ds:X509Certificate>
                    </ds:X509Data>
                </ds:KeyInfo>
            </KeyDescriptor>

            <AttributeService Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
Location="https://upki-test-idpvm.nii.ac.jp:8443/idp/profile/SAML1/SOAP/AttributeQuery"/>      ←ホスト名

            <AttributeService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP"
Location="https://upki-test-idpvm.nii.ac.jp:8443/idp/profile/SAML2/SOAP/AttributeQuery"/>      ←ホスト名

            <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
            <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>

        </AttributeAuthorityDescriptor>

        <!-- This is just information about the entity in human terms. -->
        <Organization>
                <OrganizationName xml:lang="en">Your IdP</OrganizationName>          ←組織名
                <OrganizationDisplayName xml:lang="en">Your IdP</OrganizationDisplayName>      ←組織表示名
                <OrganizationURL xml:lang="en">http://Your HomePage/</OrganizationURL>          ←組織 URL
        </Organization>
        <ContactPerson contactType="Your ContactType">          ←管理者名
                <GivenName>Your GivenName</GivenName>
                <SurName>Your SurName</SurName>
                <EmailAddress>Your Email Address</EmailAddress>          ←管理者メールアドレス
        </ContactPerson>
```

入手した証明書に変更

完成した新しい IdP 用のメタデータを、ヘルプデスク (upki-sso-help@nii.ac.jp) へ送付してください。
ヘルプデスクでは、送付していただいたメタデータをフェデレーション全体のメタデータ (upki-fed-metadata-signed.xml、または、upki-test-fed-metadata-signed.xml) に登録致します。

### 3-1-5. Back-Channel の設定

Shib1.3 は IdP との通信時に TLS 接続を行うため、Back-Channel が必要です。

(1) キーストアの設定

```
# cd /opt/shibboleth-idp/credentials
# keytool -importcert –trustcacerts -alias ca –keystore keystore.jks /
 -storetype jks -file CA 証明書.crt -storepass キーストアパスワード
# keytool -importcert –trustcacerts -alias interca –keystore keystore.jks /
 -storetype jks -file 中間 CA 証明書.crt -storepass キーストアパスワード
# openssl pkcs12 -export -out pkcs12.p12 -in サーバ証明書.crt -inkey サーバ秘密鍵.key /
  -name サーバ名
# keytool -importkeystore -srckeystore pkcs12.p12 -destkeystore keystore.jks /
 -srcstoretype pkcs12 -deststoretype jks -srcalias サーバ名 -destalias サーバ名 /
 –storepass キーストアパスワード
```

(2) SOAP 設定

Tomcat インストール先/conf/server.xml ファイルに以下を追加します。

```
<Connector port="8443"
        maxHttpHeaderSize="8192"
        maxSpareThreads="75"
        scheme="https"
        secure="true"
        clientAuth="want"
        SSLEnabled="true"
        sslProtocol="TLS"
        keystoreFile="/opt/shibboleth-idp/credentials/keystore.jks"
        keystorePass="キーストアパスワード"
        truststoreFile="/opt/shibboleth-idp/credentials/keystore.jks"
        truststorePass="キーストアパスワード"
        truststoreAlgorithm="DelegateToApplication"/>
```

### 3-1-6. 接続テスト

① SP にアクセスする

現段階で用意されている SP は、以下の接続確認サイトがあります。

https://upki-test-sp00.nii.ac.jp

https://upki-test-sp01.nii.ac.jp

https://upki-test-sp02.nii.ac.jp

＜セミナー注： セミナーでは下記を準備しています。

https://seminar-sp00.nii.ac.jp (Shib2.x プロトコル)

https://seminar-sp09.nii.ac.jp (Shib1.3 プロトコル)

② ID、パスワードの入力

上記 SP にアクセスすると DS の IdP 選択画面が表示されるので、構築した IdP を選択します。次に、選択した IdP のログイン画面が表示されるので、以下のアカウントでログインします。

| id | password |
|---|---|
| test001 | test001 |
| test002 | test002 |
| test003 | test003 |

③ ログインの確認

ログインに成功すると IdP が送信した属性一覧が表示されます。

また、upki-test-sp00 と upki-test-sp02 では、アクセスしたログを確認することができますので、ログインに失敗した場合は、IdP のログと合わせて解析が可能です。

④ SSO の確認

１つ目のテスト SP に接続後、そのブラウザで２つ目のテスト SP にアクセスします。この場合、SSO が動作して、DS や IdP が表示されずに、２つ目のテスト SP にログインすることを確認して下さい。

## 3-2. 構築後のカスタマイズ

### 3-2-1. 属性管理（登録、変換、リリース方法）

シボレスには、基本的には多くの属性を LDAP や DB から取得してリリースするための設定が既に入っており、これらのコメントを削除して有効化するだけで実行することがで

きます。
また、属性の変換機能として、”@nii.ac.jp”といったドメインの付与、値の変換、固定値の割り当てや、スクリプトを利用した変換等が可能です。
さらに、リリース制御では、サイトとしてのポリシー、各個人のポリシーによる制御や、各 SP に対応したリリース制御等が可能です。
以下では、新たな ID や属性の追加、そのリリースの方法について説明します。

### 3-2-2. LDAP ID の追加方法

以下の利用者を追加する例を示します。

| 属性 | ユーザ 4 |
|---|---|
| uid | test004 |
| userPassword | test004 |
| mail | test004_email@nii.ac.jp |
| sn | test004_sn |
| o | |
| ou | Test Unit4 |
| givenName | test004_givenname |
| displayName | test004_displayname |
| eduPersonAffiliation | Student |
| eduPersonPrincipalName | |
| eduPersonEntitlement | |
| eduPersonScopedAffiliation | |
| eduPersonTargetID | |
| jasn | テスト 004_sn |
| jaGivenName | テスト 004_givenname |
| jaDisplayName | テスト 004_displayname |
| jao | テスト 004_大学 |
| jaou | テスト 004_学部 4 |

① ldif ファイル(sample1.ldif)の作成

```
# uid=test004,ou=Test Unit1,o=Test Organization,dc=ac,c=JP
dn: uid=test004,ou=Test Unit1,o=Test Organization,dc=ac,c=JP
objectClass: eduPerson
objectClass: inetOrgPerson
uid: test004
o;lang-ja: テスト 004_大学
ou: Test Unit4
ou;lang-ja: テスト 004_学部 4
sn: test004_sn
sn;lang-ja: テスト 004_sn
cn: test004_cn
userPassword: test004
givenName: test004_givenname
givenName;lang-ja: テスト 004_givenname
displayName: test004_displayname
displayName;lang-ja: テスト 004_displayname
mail: test004_email@nii.ac.jp
eduPersonAffiliation: student
```

② 上記①の ldif ファイルを用いた登録

```
#ldapadd -x -h localhost -D " cn=olmgr,o=Test Organization,dc=ac,c=JP " -w csildap
-f sample1.ldif
```

### 3-2-3. 属性の追加方法

利用者に「displayName」属性を追加する例を示します。

① ldif ファイル(sample2.ldif)の作成

```
dn: uid=test004,ou=Test Unit1,o=Test Organization,dc=ac,c=JP
changetype: modify
add : displayName
displayName:Test4
```

② 上記①の ldif ファイルを用いた登録

```
# ldapmodify -x -h localhost –D " cn=olmgr,o=Test Organization,dc=ac,c=JP " -w csildap
-f example2.ldif
```

## 3-2-4. 属性のリリース方法

前章で追加した「displayName」属性を SP へリリースする例を示します。

① スキーマの確認

- /etc/openldap/schema 配下にスキーマファイルがあります。
- 「displayName」属性は、/etc/openldap/schema/inetorgperson.schema にて以下のように定義されています。

```
（中略）
# displayName
# When displaying an entry, especially within a one-line summary list, it # is useful to be able
to identify a name to be used.  Since other attri- # bute types such as 'cn' are multivalued,
an additional attribute type is # needed.  Display name is defined for this purpose.
attributetype ( 2.16.840.1.113730.3.1.241
        NAME 'displayName'
        DESC 'RFC2798: preferred name to be used when displaying entries'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
        SINGLE-VALUE )
（中略）
```

② /opt/shibboleth-idp/conf/attribute-resolver.xml への登録

```
<AttributeResolver xmlns="urn:mace:shibboleth:2.0:resolver" xmlns:resolver="urn:mace:shibboleth:2.0:resolver"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:pc="urn:mace:shibboleth:2.0:resolver:pc"
    xmlns:ad="urn:mace:shibboleth:2.0:resolver:ad" xmlns:dc="urn:mace:shibboleth:2.0:resolver:dc"
    xmlns:enc="urn:mace:shibboleth:2.0:attribute:encoder" xmlns:sec="urn:mace:shibboleth:2.0:security"
    xsi:schemaLocation="urn:mace:shibboleth:2.0:resolver classpath:/schema/shibboleth-2.0-attribute-resolver.xsd
                        urn:mace:shibboleth:2.0:resolver:pc classpath:/schema/shibboleth-2.0-attribute-resolver-pc.xsd
                        urn:mace:shibboleth:2.0:resolver:ad classpath:/schema/shibboleth-2.0-attribute-resolver-ad.xsd
                        urn:mace:shibboleth:2.0:resolver:dc classpath:/schema/shibboleth-2.0-attribute-resolver-dc.xsd
                        urn:mace:shibboleth:2.0:attribute:encoder
classpath:/schema/shibboleth-2.0-attribute-encoder.xsd
                        urn:mace:shibboleth:2.0:security classpath:/schema/shibboleth-2.0-security.xsd">
（中略）
    <resolver:AttributeDefinition id="displayName" xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="displayName">
        <resolver:Dependency ref="myLDAP" />
        <resolver:AttributeEncoder xsi:type="SAML1String" xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:mace:dir:attribute-def:displayName" />
         <resolver:AttributeEncoder xsi:type="SAML2String"
xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
             name="urn:oid:2.16.840.1.113730.3.1.241"  ←①のoid
             friendlyName="displayName" />
    </resolver:AttributeDefinition>
（中略）
</AttributeResolver>
```

③　/opt/shibboleth-idp/conf/attribute-filter.xml への登録

```
（中略）
    <!--  Release the transient ID to anyone -->
    <AttributeFilterPolicy id="releaseTransientIdToAnyone">
        <PolicyRequirementRule xsi:type="basic:ANY" />
（中略）
        <AttributeRule attributeID="displayName">
            <PermitValueRule xsi:type="basic:ANY" />
        </AttributeRule>
（中略）
    </AttributeFilterPolicy>
（中略）
```

追加

### 3-2-5. 認証方法の変更、設定（証明書による認証）

LDAP を利用した ID/パスワード認証の他に、様々な認証方法を利用することが可能です。
以下では、クライアント証明書を利用した認証の設定方法を示します。
この例では、

- クライアント証明書を発行するキャンパス認証局の CA 証明書＝Camp-CA.crt
- クライアント証明書のサブジェクト"o"の値＝ "Test_University_A"

として設定を行い、クライアント証明書が有効な証明書であり、かつ、上記の条件を満たす場合に認証を行う設定としています。
また、eduPersonPrincipalName をキーとして、クライアント証明書のサブジェクト "CN" の値により LDAP から各属性を取得しています。そのため、クライアント証明書のサブジェクト"CN"の値を、LDAP の eduPersonPrincipalName に格納しておくことが必要です。

- /opt/shibboleth-idp/conf/handler.xml の変更（オリジナルに戻す）

クライアント証明書を用いた認証のために handler.xml ファイルを変更します。

```
<!-- Login Handlers -->
<!-- -->                ←コメントアウトを閉じて、こちらを有効にします
<LoginHandler xsi:type="RemoteUser">
    <AuthenticationMethod>
        urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified</AuthenticationMethod>
</LoginHandler>
<!-- -->                ←コメントアウト

<!--  Username/password login handler -->
<!--          ← コメントアウトします
<LoginHandler xsi:type="UsernamePassword"
        jaasConfigurationLocation="file:///opt/shibboleth-idp/conf/login.config">
    <AuthenticationMethod>
        urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
    </AuthenticationMethod>
</LoginHandler>
-->          ← コメントアウトします
```

・ /etc/httpd/conf.d/ssl.conf への追加

```
（省略）
<VirtualHost _default_:443>
（省略）
ProxyPass /idp ajp://localhost:8009/idp
<Location /idp/Authn/RemoteUser>    ←追加
    SSLCACertificateFile /opt/shibboleth-idp/credentials/Camp-CA.crt    ←追加
    SSLVerifyClient require    ←追加
    SSLVerifyDepth 3    ←追加
    SSLRequireSSL    ←追加
    SSLOptions +ExportCertData +StdEnvVars    ←追加
    SSLUserName SSL_CLIENT_S_DN_CN    ←追加
    SSLRequire %{SSL_CLIENT_S_DN_O} eq "Test_University_A"    ←追加
</Location>    ←追加
（省略）
</VirtualHost>
```

・ /opt/shibboleth-idp/conf/attribute-resolver.xml の修正 1

また、クライアント証明書のサブジェクト“CN”の値を edupersonPrincipalName に設定して、これをキーとして LDAP から属性を取得するため、下記の設定を行います。

**※「eduPersonPrincipalName」を検索し、場所を特定してください。（行番号は参考です）**

```
404
405     <resolver:AttributeDefinition id="principalName" xsi:type="Scoped"
                                      xmlns="urn:mace:shibboleth:2.0:resolver:ad"
406         scope="nii.ac.jp" sourceAttributeID="eduPersonPrincipalName">
407         <resolver:Dependency ref="remoteUser" /> ←変更
408
409         <resolver:AttributeEncoder xsi:type="SAML1ScopedString"
                                      xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
410             name="urn:mace:dir:attribute-def:eduPersonPrincipalName" />
411
412         <resolver:AttributeEncoder xsi:type="SAML2ScopedString"
                                      xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
413             name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6" friendlyName="eduPersonPrincipalName" />
414     </resolver:AttributeDefinition>
415
416 <ressolver:AttributeDefinition xsi:type="PrincipalName"
                 xmlns="urn:mace:shibboleth:2.0:resolver:ad"  id="remoteUser" /> ←追加
```

/opt/shibboleth-idp/conf/attribute-resolver.xml の修正 2

**※「LDAP Connector」を検索し、場所を特定してください。(行番号は参考です)**

```
498     <!-- Example LDAP Connector -->
499     <!-- -->
500     <resolver:DataConnector id="myLDAP" xsi:type="LDAPDirectory"
                                            xmlns="urn:mace:shibboleth:2.0:resolver:dc"
501         ldapURL="ldap://localhost" baseDN="o=test_o,dc=ac,c=JP"
                                            principal="cn=olmgr,o=test_o,dc=ac,c=JP"
502         principalCredential="csildap">
503         <FilterTemplate>
504             <![CDATA[
505                 (eduPersonPrincipalName=$requestContext.principalName)  ←変更
506             ]]>
507         </FilterTemplate>
508     </resolver:DataConnector>
509     <!-- -->
```

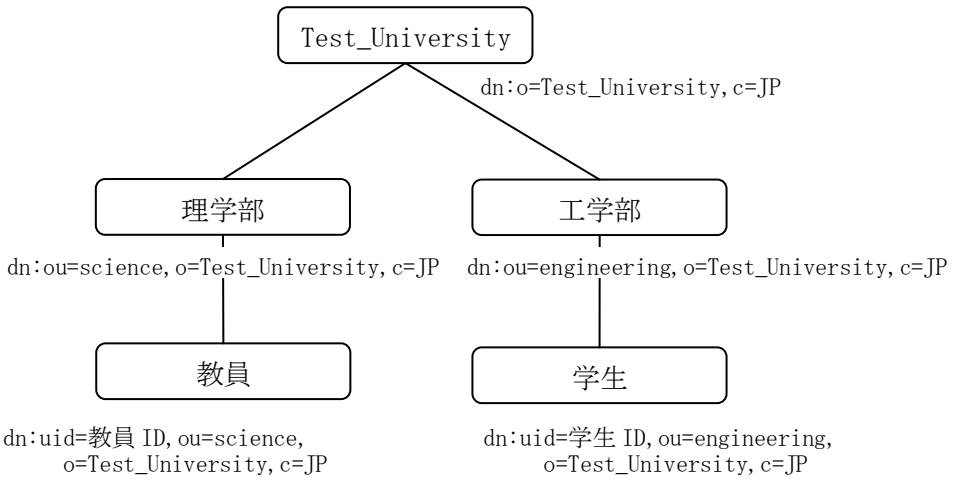
## 3-2-6. LDAP の新規作成方法

① 既存 LDAP の削除

```
#ldapdelete -x -h localhost -D "cn=olmgr,o=Test Organization,dc=ac,c=JP" -w csildap
```

② 新しい LDAP のツリー構造の設計

以下の組織構造を新規作成する例を示します。



③　ldif ファイル(sample3.ldif)の作成

```
#TOP ツリー用
dn: o= Test_University,c=JP
objectClass: organization
o: Test_University
#部署ツリー用
dn: ou= science,o= Test_University,c=JP
objectClass: organizationalUnit
ou: science
dn: ou= engineering,o= Test_University,c=JP
objectClass: organizationalUnit
ou: engineering
#利用者ツリー用
dn: uid=t001,ou= science,o= Test_University,c=JP
objectClass:inetOrgPerson
cn: takesi
sn: yamada
uid: t001
userPassword: p-yamada
dn: uid=s002,ou= engineering,o= Test_University,c=JP
objectClass:inetOrgPerson
cn: taro
sn: maeda
uid: s002
userPassword: p-maeda
```

④　/etc/openldap/slapd.conf の編集

```
(中略)
suffix      "o= Test_University,c=JP"      ←suffix
rootdn      "cn=Manager,o= Test_University,c=JP"      ←rootdn
rootpw      {SSHA}2ZbK+0IyV92py+vi67X80RNAKg066GXS    ←暗号化※したものを記載
(中略)
```

※暗号化の例　：　「csildap2」というパスワードを暗号化する

```
#slappasswd -h {SSHA} -s csildap2
80U6H/KA4fVlvC+6DzN73DTP/dEAgl76    ←これを slapd.conf の rootpw に記載
```

⑤　LDAP サービスの再起動

```
#service ldap restart
```

⑥　上記③の ldif ファイルを用いた登録

```
#ldapadd -x -h localhost -D " cn=Manager,o= Test_University,c=JP " -w csildap2 -f example3.ldif
```

⑦　/opt/shibboleth-idp/conf/attribute-resolver.xml の編集

```
<resolver:DataConnector id="myLDAP" xsi:type="LDAPDirectory" xmlns="urn:mace:shibboleth:2.0:resolver:dc"
    ldapURL="ldap://localhost" baseDN=" o= Test_University,c=JP "
    principal=" cn=Manager,o= Test_University,c=JP "
    principalCredential="csildap2">  ←LDAP のパスワードを設定
    <FilterTemplate>
        <![CDATA[
            (uid=$requestContext.principalName)
        ]]>
    </FilterTemplate>
```

⑧　tomcat サービスの再起動

```
#service tomcat stop
#service tomcat start
```

### 3-2-7. メタデータの自動更新設定方法

Shibboleth IdP 2.1 では、デフォルトでメタデータ を自動取得するための設定が記述されています。

relying-party.xml の Metadata Configuration に関する設定項目をご確認ください。

### 3-2-8. メタデータ署名の検証設定方法

① 設定方法

Shibboleth 2.1 では、デフォルトで署名付きメタデータを利用する設定が記述されています。

relying-party.xml の Metadata Configuration と Security Configurations の設定を参考にしてください。

## 3-3. 運用方法

### 3-3-1. metadata の管理方法

メタデータは新規 IdP、新規 SP の追加や、既存 IdP、既存 SP の証明書の更新等により、常に更新されます。そのため、定期的にリポジトリから最新の共通メタデータをダウンロードして IdP のメタデータを更新してください。

また、証明書の更新等、IdP のメタデータに更新があった場合は、すみやかにヘルプデスクに送付してください。

# 4. 関連 URL

■ UPKI プロジェクト（UPKI イニシアティブ）

https://upki-portal.nii.ac.jp/

■ UPKI 認証連携基盤によるシングルサインオン実証実験

https://upki-portal.nii.ac.jp/docs/fed/

■ UPKI 認証連携基盤リポジトリ

https://upki-portal.nii.ac.jp/docs/fed/technical

■ Shibboleth プロジェクト

http://shibboleth.internet2.edu/

■ Shibboleth2.0 Wiki（Shibboleth2.0 の構築、設定に関する公式サイト）

https://spaces.internet2.edu/display/SHIB2/Home

■ Switch.aai （スイスのフェデレーション）

http://www.switch.ch/aai/

■ InCommon（米国のフェデレーション）

http://www.incommonfederation.org/