uApproveJP-2.5.1 のインストールおよび設定方法

この文書にはuApprove Jet Pack (以下、「uApprove JP」) のインストールガイドと総合マニュアルが記されています。

uApprove JPはShibboleth Identity Provider 2.xを拡張するプラグインです。オリジナルのuApproveに独自の改修を施しています。これを利用することにより、利用者はIdentity Providerで認証する際に利用条件(ToU)を承諾するとともに、属性を選択的に送信することができます。 uApprove JPのコンセプトに関するより詳細な情報はこちらを参照してください。

本ガイドに関する注意事項:

- このガイドでは、uApprove JPはLinuxシステムにインストールされると仮定しています。Windows等の他のオペレーティングシステムにインストールすることも可能です。その場合は、いくつかのパスやコマンドを適切なものに置き換えてください。
- このガイドでは、パスやコマンドは $IDP_INSTALL$、 IDP_HOME、 $UAPPROVE_INSTALL$といった変数で示されます。明示的に置き換えが不要と書かれていない限りは、これらの変数は実際のパスに置き換えてください。

目次

想定

- Shibboleth Identity Provider は、 $IDP_INSTALL$ (例: /usr/local/src/shibboleth-identity-provider-#version#) に展開されているものとします。
- Shibboleth Identity Provider は、 IDP_HOME (例: /opt/shibboleth-idp) にインストールされているものとします。
- Tomcatは、 $CATALINA_HOME$ (例: /usr/java/tomcat) にインストールされているものとします。
- uApprove JP は、 $UAPPROVE_INSTALL$ (例: /usr/local/src/uApproveJP-#version#) にダウンロード、展開されているものとします。

1 インストール

1.1 前提条件

- Shibboleth Identity Provider 2.4.0 以降がインストールされている必要があります。
- MySQL 5.1 以降がインストールされている必要があります。

1.2 ライブラリのインストール

ライブラリを IdP のライブラリディレクトリにコピーします:

```
# cp $UAPPROVE_INSTALL$/lib/*.jar $IDP_INSTALL$/lib
# cp $UAPPROVE_INSTALL$/lib/jdbc/*.jar $IDP_INSTALL$/lib
```

JDBC コネクタを IdP の classpath に配置します。提供されているMySQLまたはHSQLのJDBCコネクタのいずれかを使えます:

```
# cp $UAPPROVE_INSTALL$/lib/jdbc/optional/#jdbc-connector#.jar $IDP_INSTALL$/lib
```

⚠ $IDP_INSTALL$/libにはそれぞれのライブラリの単一のバージョンのみが存在するようにしてください。

1.3 設定テンプレート

設定テンプレートを IdP の設定ディレクトリにコピーします:

```
# cp $UAPPROVE_INSTALL$/manual/configuration/uApprove.properties $IDP_HOME$/conf
# cp $UAPPROVE_INSTALL$/manual/configuration/uApprove.xml $IDP_HOME$/conf
```

1.4 Webapp ファイル

JSP や CSS といった web アプリケーションファイルおよび画像ファイルを IdP の webapp ディレクトリにコピーします:

```
# mkdir $IDP_INSTALL$/src/main/webapp/uApprove
# cp $UAPPROVE_INSTALL$/webapp/* $IDP_INSTALL$/src/main/webapp/uApprove
```

1.5 データベースの準備

⚠ 以下のデータベースパラメータは一例です。実際の値は必要に応じて変更してください。特にパスワードは安全なものを用意してください。

- “uApprove” という名前のデータベースを作成します。
- ユーザ名 “uApprove” およびパスワード “secret” でデータベースのユーザを作成します。
- INSERT, SELECT, UPDATE, DELETE の各権限を作成したユーザに付与します。
- 以下のスキーマを用いて初期テーブル構造を作成します:
    - $UAPPROVE_INSTALL$/manual/storage/terms-of-use-schema.sql
    - $UAPPROVE_INSTALL$/manual/storage/attribute-release-schema.sql
    - $UAPPROVE_INSTALL$/manual/storage/service-access-data-schema.sql

2 基本的なデプロイ

2.1 Web アプリケーションデプロイメントデスクリプタ

IdP の Web アプリケーションデプロイメントデスクリプタ ($IDP_INSTALL$/src/main/webapp/WEB-INF/web.xml) を修正します。以下に示すように、実際のファイルを反映してください。

- contextConfigLocation コンテキストパラメータに IDP_HOME/conf/uApprove.xml を追加します。 IDP_HOME の部分は変更せずにこのまま記述してください。後で行いますIdPの再デプロイの際に自動的に置き換えられます。
- 以下に示す必要となるフィルタおよびサーブレットを追加します。

```
<web-app ...>

    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>$IDP_HOME$/conf/internal.xml; $IDP_HOME$/conf/service.xml; $IDP_HOME$/conf/uApprove.xml;</param-value>
    </context-param>

    <!-- IdP Listeners, Filters and Servlets -->
    <!-- ...                                 -->

    <!-- uApprove Filter and Servlets -->

    <filter>
        <filter-name>uApprove</filter-name>
        <filter-class>ch.SWITCH.aai.uApprove.Intercepter</filter-class>
    </filter>

    <filter-mapping>
        <filter-name>uApprove</filter-name>
        <url-pattern>/profile/Shibboleth/SSO</url-pattern>
        <url-pattern>/profile/SAML1/SOAP/AttributeQuery</url-pattern>
        <url-pattern>/profile/SAML1/SOAP/ArtifactResolution</url-pattern>
        <url-pattern>/profile/SAML2/POST/SSO</url-pattern>
        <url-pattern>/profile/SAML2/POST-SimpleSign/SSO</url-pattern>
        <url-pattern>/profile/SAML2/Redirect/SSO</url-pattern>
        <url-pattern>/profile/SAML2/Unsolicited/SSO</url-pattern>
        <url-pattern>/Authn/UserPassword</url-pattern>
    </filter-mapping>

    <servlet>
        <servlet-name>uApprove - Terms Of Use</servlet-name>
        <servlet-class>ch.SWITCH.aai.uApprove.tou.ToUServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>uApprove - Terms Of Use</servlet-name>
        <url-pattern>/uApprove/TermsOfUse</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>uApprove - Attribute Release</servlet-name>
        <servlet-class>ch.SWITCH.aai.uApprove.ar.AttributeReleaseServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>uApprove - Attribute Release</servlet-name>
        <url-pattern>/uApprove/AttributeRelease</url-pattern>
    </servlet-mapping>

</web-app>
```

2.2 設定のカスタマイズ

IDP_HOME/conf/internal.xml にて以下の変更を行います。 id="shibboleth.OpensamlConfig" の Bean 定義の <constructor-arg> に、 id="uajpMetadataExtensions" の Bean 定義を追加してください:

```
<bean id="shibboleth.OpensamlConfig" class="edu.internet2.middleware.shibboleth.common.config.OpensamlConfigBean"
         depends-on="shibboleth.LogbackLogging">
    ...
    <constructor-arg>
        <list>
            <bean id="shibMetadataExtensions" class="org.opensaml.util.resource.ClasspathResource">
                <constructor-arg value="/shibboleth-saml-ext-config.xml"/>
            </bean>
            <bean id="uajpMetadataExtensions" class="org.opensaml.util.resource.ClasspathResource">
                <constructor-arg value="/uApprove-jp-metadata-config.xml"/>
            </bean>
        </list>
    </constructor-arg>
    <property name="parserPool" ref="shibboleth.ParserPool"/>
    ...
```

IDP_HOME/conf/uApprove.xml にて以下の変更を行います:

変更前:

```
<context:property-placeholder location="classpath:/configuration/uApprove.properties" />
```

変更後:

```
<context:property-placeholder location="file:$IDP_HOME$/conf/uApprove.properties" />
```

IDP_HOME は実際のパス (例: /opt/shibboleth-idp) にしてください(その結果、値全体は file:/opt/shibboleth-idp/conf/uApprove.properties のようになります)。

正しいIdPのメタデータをIDP_HOME/metadata/idp-metadata.xmlに配置してください。そのうえでIDP_HOME/metadata/idp-metadata.xmlを読み込むようにIDP_HOME/conf/relying-party.xmlを編集します:

```
    <!-- ========================================== -->
    <!--      Metadata Configuration                -->
    <!-- ========================================== -->
    <!-- MetadataProvider the combining other MetadataProviders -->
    <metadata:MetadataProvider id="ShibbolethMetadata" xsi:type="metadata:ChainingMetadataProvider">

        <!-- Load the IdP's own metadata.  This is necessary for artifact support. -->
        <metadata:MetadataProvider id="IdPMD" xsi:type="metadata:FilesystemMetadataProvider"
                                   metadataFile="$IDP_HOME$/metadata/idp-metadata.xml"
                                   maxRefreshDelay="P1D" />
```

データベースおよび必要な機能に応じて IDP_HOME/conf/uApprove.properties をカスタマイズします。設定オプションについては uApprove.properties のインライン文書を参照してください。

"Terms of Use" モジュールを有効にする場合(デフォルトで有効になっています)、所属組織に適した文書を設定する必要があります。

"Terms of Use" の HTML ファイルは、 $UAPPROVE_INSTALL$/manual/examples/terms-of-use.html に例があります。

- $UAPPROVE_INSTALL$/manual/examples/terms-of-use.html を IDP_HOME/conf/terms-of-use.html にコピーします:

  ```
  # cp $UAPPROVE_INSTALL$/manual/examples/terms-of-use.html $IDP_HOME$/conf/terms-of-use.html
  ```

- IDP_HOME/conf/terms-of-use.html を適切に修正します。
- それに従って、 IDP_HOME/conf/uApprove.properties の中の tou.resource を適切に修正します:

  ```
  tou.resource = file:$IDP_HOME$/conf/terms-of-use.html
  ```

2.3 カスタムテンプレート

テンプレートをカスタマイズしたい場合は、 テンプレートのカスタマイズ を参照してください。

少なくとも、所属機関のロゴを $IDP_INSTALL$/src/main/webapp/uApprove/logo.png にコピーする必要があります。デフォルトではこのファイルはプレースホルダのロゴになっています。
さらに、フェデレーションのロゴを $IDP_INSTALL$/src/main/webapp/uApprove/federation-logo.png にコピーすることもできます(デフォルトでは空のプレースホルダのロゴになっています)。

- SWITCHaai フェデレーションの場合は、ロゴは http://www.switch.ch/aai/design/images/switchaai-logo.png から入手できます。
- 学認の場合は、ロゴは https://www.gakunin.jp/info/logo/ から入手できます。
  大きなサイズのロゴしかないので、 $IDP_INSTALL$/src/main/webapp/uApprove/ にコピーした attribute-release.jsp および attribute-check.jsp の <img> に height, width を指定して表示する大きさを調整してください。

```
<img src="<%= request.getContextPath()%>/uApprove/federation-logo.png" alt="" class="federation_logo" height="32" width="162">
```

2.4 デプロイ

IdP で uApprove JP を有効にするには IdP を再デプロイする必要があります:

```
# cd $IDP_INSTALL$
# ./install.sh
```

idp.war を $CATALINA_HOME$/webapps にコピーします:

```
# cp $IDP_HOME$/war/idp.war $CATALINA_HOME$/webapps/
```

Tomcat を再起動します:

```
# service tomcat6 restart
```

3 アップグレード

3.1 uApprove.jp 2.2.1 からのアップグレード

本節では、すでに uApprove.jp 2.2.1 / 2.2.1a / 2.2.1b / 2.2.1c をインストール・運用しているIdPにおいて、uApprove JP 2.5.1 にアップグレードする手順を示します。

古いバージョンを完全に消去してからクリーンインストールを実施する必要があります。

インストール方法は 1 インストール を参照してください。

- デプロイされている uApprove.jp-2.2.1 を削除します。
- 設定ファイルは新しい設定ファイルを使用し、カスタマイズします。

  新しいToUはカスタマイズ可能なHTMLファイルであり、uApprove.jp 2.2.1 の ToU XML からプレーンな HTML ファイルにコピーするだけで使用できます。

- uajpmf:AttributeUapprove マッチングルールは使用できません。
  uajpmf:AttributeInMetadata を使用したルールに書き換えてください。4.7 AttributeInMetadataマッチングルールを参照してください。

1. メタデータに <RequstedAttribute> として宣言されている属性は isRequired 属性に従い、 true の場合は必須、 false の場合はオプショナルに、<RequstedAttribute> として宣言されてない属性およびそもそもメタデータに <AttributeConsumingService> がない場合は非表示とするルールは下記の通り書き換えできます。

uApprove.jp 2.2.1		uApprove JP 2.5.1
<PermitValueRule xsi:type="uajpmf:AttributeUapprove" *isApproved="true"* requestedOnly="true" />	⇒	<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" onlyIfChecked="true" onlyIfRequired="false" *matchIfMetadataSilent="false"* />

2. メタデータに <RequstedAttribute> として宣言されている属性は isRequired 属性に従い、 true の場合は必須、 false の場合はオプショナルに、メタデータに <AttributeConsumingService> がない場合もオプショナルとするルールは下記の通り書き換えできます。ただし、メタデータに <AttributeConsumingService> があるがその中で <RequstedAttribute> として宣言されていない属性は、 uajpmf:AttributeUapprove ではオプショナルとなりますが uajpmf:AttributeInMetadata では非表示となります。

uApprove.jp 2.2.1		uApprove JP 2.5.1
<PermitValueRule xsi:type="uajpmf:AttributeUapprove" *isApproved="true" requestedOnly="false"* />	⇒	<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" onlyIfChecked="true" onlyIfRequired="false" matchIfMetadataSilent="true" />

3. メタデータに <AttributeConsumingService> が含まれる場合のみに適用されるポリシーが記述されている場合、一対一の書き換えができないため、 <PolicyRequirementRule> は basic:ANY として、当該ポリシー内の各ルールでは <AttributeConsumingService> が含まれない場合の挙動も考慮し matchIfMetadataSilent に反映させてください。

uApprove.jp 2.2.1		uApprove JP 2.5.1
<PolicyRequirementRule xsi:type="uajpmf:AttributeUapprove" /> ※<AttributeConsumingService> が含まれる場合にマッチする	⇒	<PolicyRequirementRule xsi:type="basic:ANY" />
<PermitValueRule xsi:type="uajpmf:AttributeUapprove" *isApproved="true" requestedOnly="false"* /> かつ他のルールで <AttributeConsumingService> が含まれない場合に**非表示**としている場合	⇒	<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" onlyIfChecked="true" onlyIfRequired="false" *matchIfMetadataSilent="false"* /> ※ただし上記2.の場合と同じく、<AttributeConsumingService>が存在しかつ同要素中に<RequstedAttribute>として宣言されていない属性に対する挙動が変化する（オプショナル→非表示）ことにご注意ください。
<PermitValueRule xsi:type="uajpmf:AttributeUapprove" *isApproved="true"* requestedOnly="true" /> かつ他のルールで <AttributeConsumingService> が含まれない場合に**非表示**としている場合	⇒	<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" onlyIfChecked="true" onlyIfRequired="false" *matchIfMetadataSilent="false"* />
<PermitValueRule xsi:type="uajpmf:AttributeUapprove" *isApproved="true"* requestedOnly="true" /> かつ他のルールで <AttributeConsumingService> が含まれない場合に**オプショナル**としている場合	⇒	<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" onlyIfChecked="true" onlyIfRequired="false" matchIfMetadataSilent="true" />
<PermitValueRule xsi:type="uajpmf:AttributeUapprove" *isApproved="true"* requestedOnly="true" /> かつ他のルールで <AttributeConsumingService> が含まれない場合に**必須**としている場合	⇒	<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" *onlyIfChecked="false"* onlyIfRequired="false" matchIfMetadataSilent="true" /> ※ただしisRequired="false"の属性に対する挙動が変化する（オプショナル→必須）ことにご注意ください。 ※上記の場合の挙動を非表示にするには onlyIfRequied="true" としてください。

4. メタデータに <AttributeConsumingService> が含まれない場合のみに適用されるポリシーが記述されている場合も上記3.と同様の対処が必要になります。 <PolicyRequirementRule> は basic:ANY として、当該ポリシー内の各ルールでは <AttributeConsumingService> が含まれる場合の挙動も考慮し matchIfMetadataSilent に反映させてください。


上記表の斜体の属性はデフォルト値であるため実際の設定では記載していないことが多い。

- JSP は新しい JSP ファイルを使用します。
  4.1 属性送信の同意のリセットの機能を使用している場合は、 login.jsp を修正します。
- データベースのマイグレーションはできません。

4 高度なデプロイ

この節では高度な設定についてのトピックを取り上げます。

4.1 属性送信の同意のリセット

利用者がログインのフローの際に属性送信の同意をクリア [1] できる機能を提供する場合は、 $IDP_INSTALL$/src/main/webapp/login.jsp にチェックボックスを追加します:

```
<form action="<%=request.getAttribute("actionUrl")%>" method="post">
  ...
  <input id="uApprove.consent-revocation" type="checkbox" name="uApprove.consent-revocation" value="true"/>
  <label for="uApprove.consent-revocation">Clear my attribute release consent</label>
  ...
</form>
```

Shibboleth IdP 2.4.0 以降のデフォルトのログインページではCSSでラベルを非表示としています。そのため、<label>でstyle属性を変更する必要があります:

```
<form action="<%=request.getAttribute("actionUrl")%>" method="post">
  ...
  <section>
    <input id="uApprove.consent-revocation" type="checkbox" name="uApprove.consent-revocation" value="true"/>
    <label for="uApprove.consent-revocation" style="position: relative; left: 0px;">Clear my attribute release consent<
/label>
  </section>
  ...
</form>
```

[1] クリアとは、アクセスして来たRelying Partyの属性送信の同意のすべての削除が行われた際に一般的な合意も削除することを意味します。

4.2 ストレージ

ファイルのみのストレージ

シンプルなデプロイのために、ファイルのみのデータベースを使うことができます。HSQL はそのようなオプションを提供します。
uApprove.properties にデータベースプロパティを定義します。

```
database.driver             = org.hsqldb.jdbcDriver
database.url                = jdbc:hsqldb:file:/var/opt/uApprove/hsql.db
database.username           = SA
database.password           =
```

提供されているスキーマを用いてデータベースを初期化します:

```
echo "SHUTDOWN;" > /tmp/shutdown
java -jar $HSQLDB_HOME$/lib/sqltool.jar ¥
  --inlineRC=url=jdbc:hsqldb:file:/var/opt/uApprove/hsql.db,user=SA,password= ¥
  $UAPPROVE_INSTALL$/manual/storage/terms-of-use-schema.sql /tmp/shutdown
java -jar $HSQLDB_HOME$/lib/sqltool.jar ¥
  --inlineRC=url=jdbc:hsqldb:file:/var/opt/uApprove/hsql.db,user=SA,password= ¥
  $UAPPROVE_INSTALL$/manual/storage/attribute-relase-schema.sql /tmp/shutdown
java -jar $HSQLDB_HOME$/lib/sqltool.jar ¥
  --inlineRC=url=jdbc:hsqldb:file:/var/opt/uApprove/hsql.db,user=SA,password= ¥
  $UAPPROVE_INSTALL$/manual/storage/service-access-data-schema.sql /tmp/shutdown
```

⚠ $HSQLDB_HOME$ は、ダウンロードしたHSQL ディストリビューションを展開した場所を定義します。

⚠ ユーザが実行しているコンテナ (例えば Jetty) がdbディレクトリへの書き込み許可を持っていることを確認してください。

SQL文のカスタマイズ

1. $UAPPROVE_INSTALL$/manual/storage/sql-statements.properties を IDP_HOME/conf/uApprove.sql-statements.properties にコピーします。
2. IDP_HOME/conf/uApprove.sql-statements.properties をカスタマイズします。
3. カスタマイズした sql-statements.properties を $IDP_HOME/conf/uApprove.xml で有効化します:

```
<bean id="uApprove.touModule" class="ch.SWITCH.aai.uApprove.tou.ToUModule" ...>
    <!-- ... -->
    <property name="storage">
        <bean class="ch.SWITCH.aai.uApprove.tou.storage.JDBCStorage" ...
            p:sqlStatements="file:/$IDP_HOME$/conf/uApprove.sql-statements.properties" ... />
    </property>
</bean>

<!-- ... -->

<bean id="uApprove.attributeReleaseModule" class="ch.SWITCH.aai.uApprove.ar.AttributeReleaseModule" ...>
    <!-- ... -->
    <property name="storage">
        <bean class="ch.SWITCH.aai.uApprove.ar.storage.JDBCStorage" ...
            p:sqlStatements="file:/$IDP_HOME$/conf/uApprove.sql-statements.properties" ... />
    </property>
</bean>
```

JDBCコネクションの緩やかな扱い

JDBC ストレージは、データベースの一時的な問題(接続できない等)の場合に緩やかに扱うように設定することができます。通常は、例外を許さずエラーページを表示し、継続的な動作は行われません。
以前の ToU の承諾や属性送信の同意に関わらず、利用者は再度承諾/同意を行わなければならない(既に行っていた場合)か、次回のログインの際に行わなければなりません(最初の場合)。

⚠ checkoutTimeout の値は適切な小さな値に設定する必要があります。そうでなければ利用者は長い間待つことになります。

この設定は、 IDP_HOME/conf/uApprove.xml の中で定義されています:

```
<bean id="uApprove.touModule" class="ch.SWITCH.aai.uApprove.tou.ToUModule" ...>
    <!-- ... -->
    <property name="storage">
        <bean class="ch.SWITCH.aai.uApprove.tou.storage.JDBCStorage" ...
            p:graceful="true" ... />
    </property>
</bean>

<!-- ... -->

<bean id="uApprove.attributeReleaseModule" class="ch.SWITCH.aai.uApprove.ar.AttributeReleaseModule" ...>
    <!-- ... -->
    <property name="storage">
        <bean class="ch.SWITCH.aai.uApprove.ar.storage.JDBCStorage" ...
           p:graceful="true" ... />
    </property>
</bean>
```

JDBCコネクションプールのチューニング

詳細な設定オプションについては c3p0 configuration を参照してください。

この設定は、 IDP_HOME/conf/uApprove.xml の中で定義されています:

```
<bean id="uApprove.dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource" ...
    ...
    p:idleConnectionTestPeriod="300" ... />
```

他の JDBC コネクションプールライブラリ(例えば、 BoneCP)を使用することも可能です。 bean の設定で正しいデータソースのクラス名を定義し、必要なライブラリを classpath に配置します。

4.3 テンプレート

テンプレートのカスタマイズ

$IDP_INSTALL$/src/main/webapp/uApprove/ にある JSP 、 CSS や画像ファイルは自由にカスタマイズすることができます。 JSTL を用いているので容易にカスタマイズ出来るようになっています。

JSTL については JSTL reference を参照してください。

4.4 ローカライズ

カスタムメッセージ

$UAPPROVE_INSTALL$/manual/examples/messages に用意されているリソースバンドルは環境に合わせたり、修正することができます。修正後のファイルを IdP の classpath(例えば $IDP_INSTALL$/src/main/webapp/WEB-INF/classes/uApprove/messages)にコピーすることで利用可能となります。
バンドルのベースを IDP_HOME/conf/uApprove.xml で以下のように定義します:

```
<bean id="uApprove.viewHelper" class="ch.SWITCH.aai.uApprove.ViewHelper" ...
    p:messagesBase="uApprove.messages" />
```

日本語のメッセージを修正する場合は、文字コードが UTF-8 のファイル #view#_ja-UTF8.properties を編集してください。編集したファイルを native2ascii コマンドで変換したうえで$IDP_INSTALL$/src/main/webapp/WEB-INF/classes/uApprove/messages にコピーしてください。

attribute-release.jsp のメッセージを修正する手順は以下のようになります:

```
# cd $UAPPROVE_INSTALL$/manual/examples/messages
(attribute-release_ja-UTF8.properties を編集する)
# native2ascii attribute-release_ja-UTF8.properties attribute-release_ja.properties
# cp attribute-release_ja.properties $IDP_INSTALL$/src/main/webapp/WEB-INF/classes/uApprove/messages/
```

属性の名前と説明

ローカライズされた属性名および説明の設定方法については、 $UAPPROVE_INSTALL$/manual/examples/attribute-descriptions.xml を参照してください。

Relying Partyの名前と説明

現状では、ローカライズされた Relying Party の名前と説明を取得する際には、メタデータのうち <AttributeConsumingService> 要素および <mdui:UIInfo> 要素がサポートされています。

この名前と説明を使用する場合は、SPのメタデータを以下のように記述します:

```
<EntityDescriptor entityID="https://sp.example.org/shibboleth">
    <!-- ... -->
    <SPSSODescriptor>
        <Extensions>
            <mdui:UIInfo xmlns:mdui="urn:oasis:names:tc:SAML:metadata:ui">
                <mdui:DisplayName xml:lang="en">Example SP</mdui:DisplayName>
                <!-- Service names in other languages -->
                <mdui:Description xml:lang="en">Some description of Example SP</mdui:Description>
                <!-- Service descriptions in other languages -->
            </mdui:UIInfo>
        </Extensions>
        <!-- ... -->
        <AttributeConsumingService index="1">
            <ServiceName xml:lang="en">Example SP</ServiceName>
            <!-- Service names in other languages -->
            <ServiceDescription xml:lang="en">Some description of Example SP</ServiceDescription>
            <!-- Service descriptions in other languages -->
        </AttributeConsumingService>
    </SPSSODescriptor>
</EntityDescriptor>
```

両方記載されている場合には <mdui:UIInfo> 要素が優先されます。

SWITCHaaiフェデレーションのメタデータにはこの情報が含まれています。学認のSPについても海外SPの一部を除いてこの情報が含まれています。

4.5 厳密な比較

利用条件の内容の比較

デフォルトの設定では、利用者がToUを承諾したかどうかは ToU のバージョンの比較のみで行われます。 ToU の内容の比較を有効にしたい場合は、 IDP_HOME/conf/uApprove.xml にて以下のように設定します:

```
<bean id="uApprove.touModule" ... p:compareContent="true" ... />
```

4.6 監査ロギング

uApprove JP は、(IdP の監査ロギング機能を使用して) IDP_HOME/logs/idp-audit.log への監査ロギングを行うことができます。

利用条件の監査ロギングの有効化

ToU の監査ロギングを有効にしたい場合は、 IDP_HOME/conf/uApprove.xml にて以下のように設定します:

```
<bean id="uApprove.touModule" ... p:auditLogEnabled="true" ... />
```

ログ出力例:

```
20120101T010000Z|ch.SWITCH.aai.uApprove|||tou.acceptance|null|null|null|student1||1.0,5
b2ee897c08c79a09cd57e8602d605bf8c52db17de9793677c36b5c78644b2b3,|
```

属性の送信の監査ロギングの有効化

属性の送信の監査ロギングを有効にしたい場合は、 IDP_HOME/conf/uApprove.xml にて以下のように設定します:

```
<bean id="uApprove.attributeReleaseModule" ... p:auditLogEnabled="true" ... />
```

ログ出力例:

```
20120101T010000Z|ch.SWITCH.aai.uApprove||https://sp.example.org/shibboleth|ar.consent|null|null|null|student1||uid,surname,givenName,|
20120101T010000Z|ch.SWITCH.aai.uApprove||https://sp.example.org/shibboleth|ar.clearConsent|null|null|null|student1|||
20120101T010000Z|ch.SWITCH.aai.uApprove|||ar.generalConsent|null|null|null|student1|||
```

4.7 AttributeInMetadataマッチングルール

AttributeInMetadataマッチングルールの設定

このルールは、SP がその属性を必要とした場合に、そのメタデータにより属性の送信を許可します。属性は <SPSSODescriptor> 中の <AttributeConsumingService> によって示されます。 <RequestedAttribute> で isRequired="true" を記述した属性は必須とマークされ、 isRequired="false" を記述した属性はオプションとマークされます。詳細は SAML メタデータを参照してください。

以下の点に注意してください:

- このフィルタの利用には属性の要求者のメタデータがロードされ利用可能である必要があります。
- 要求者のメタデータは <SPSSODescriptor> ロールを持っている必要があります。このロールがリストされた属性を持っているためです。
- AttributeInMetadataマッチングルールは値のルールとしてのみ働き、<PermitValueRule> の場合のみ意味をなします。

名前空間の定義

属性フィルタのポリシーにおいて、このプラグイン用に名前空間の定義を加える必要があります。以下のように行います:

- ルート <AttributeFilterPolicyGroup> の xmlns:xsi 属性の前に xmlns:uajpmf="http://www.gakunin.jp/ns/uapprove-jp/afp/mf" 属性を追加します。
- xsi:schemaLocation 属性のホワイトスペースで区切られた値のリストの最後に以下を追加します:
  http://www.gakunin.jp/ns/uapprove-jp/afp/mf classpath:/schema/shibboleth-2.0-afp-mf-uapprovejp.xsd

ルールの定義

このルールは <PermitValueRule xsi:type="uajpmf:AttributeInMetadata"> のように記述します。以下のオプショナルな属性を使用できます:

onlyIfRequired	必須とマークされた属性のみ送信を許可し、オプションとマークされたものは送信しないブーリアンフラグです。 デフォルト値は **true** です。
matchIfMetadataSilent	メタデータに <AttributeConsumingService> がない場合にオプションとマークするかどうかを決定するブーリアンフラグです。 デフォルト値は **false** です。
onlyIfChecked	オプションとマークされた属性の送信を利用者が許可/拒否できるかどうかを示すブーリアンフラグです。 false の時の動作は Shibboleth IdP 2.4.0 以降の saml:AttributeInMetadata と同一になり、オプションとマークされた属性も必須とマークされた属性と同様にチェックボックスなしで表示されます。 デフォルト値は **false** です。

AttributeInMetadata マッチファンクションを使用した <PermitValueRule> の書き方は下記のようになります:

```
<PermitValueRule xsi:type="uajpmf:AttributeInMetadata" onlyIfRequired="false"
                 onlyIfChecked="true">
```

オプションとマークされた属性をチェックボックスつきで表示します。チェックボックスをチェックしたときだけ送信します。

AttributeInMetadata マッチファンクションを使用した <PermitValueRule> の設定例を示します:

```
<!--    ================================================================================
case 1: mail 属性、eduPersonPrincipalName 属性、eduPersonAffiliation 属性を、メタデータの
        定義と照合するルールです。

        メタデータで isRequired="true" が指定されている属性は、すべて必須情報になり常に
        送信されます。

        メタデータで isRequired="false" が指定されている属性は以下の通りです。
        * mail 属性は必須情報となり常に送信されます。
        * eduPersonPrincipalName 属性はオプション情報となります。属性選択画面ではチェック
          ボックスつきで表示されます。利用者がチェックボックスをチェックした場合に限り送信
          されます。
        * eduPersonAffiliation 属性は送信されません。

        メタデータに AttributeConsumingService をもたない SP の場合はどの属性も送信しません。
        ================================================================================ -->
<afp:AttributeFilterPolicy id="PolicyforSPwithAttributeConsumingService">
    <afp:PolicyRequirementRule xsi:type="basic:ANY" />

    <afp:AttributeRule attributeID="mail">
        <afp:PermitValueRule xsi:type="uajpmf:AttributeInMetadata"
                             onlyIfRequired="false" />
    </afp:AttributeRule>

    <afp:AttributeRule attributeID="eduPersonPrincipalName">
        <afp:PermitValueRule xsi:type="uajpmf:AttributeInMetadata"
                             onlyIfRequired="false"
                             onlyIfChecked="true" />
    </afp:AttributeRule>

    <afp:AttributeRule attributeID="eduPersonAffiliation">
        <afp:PermitValueRule xsi:type="uajpmf:AttributeInMetadata" />
    </afp:AttributeRule>

</afp:AttributeFilterPolicy>

<!--    ================================================================================
case 2: メタデータに AttributeConsumingService がない SP に対するルールを追加したルール
        です。

        AttributeConsumingService 要素を持たない SP の場合は以下の通りです。
        * mail 属性は必須情報となり常に送信されます。
        * eduPersonPrincipalName 属性はオプション情報となります。属性選択画面ではチェック
          ボックスつきで表示されます。利用者がチェックボックスをチェックした場合に限り送信
          されます。
        * eduPersonAffiliation 属性は送信されません。

        AttributeConsumingService 要素を持つ SP の場合は case 1 と同じです。
        ================================================================================ -->
<afp:AttributeFilterPolicy id="PolicyforSPwithoutAttributeConsumingService">
    <afp:PolicyRequirementRule xsi:type="basic:ANY" />

    <afp:AttributeRule attributeID="mail">
        <afp:PermitValueRule xsi:type="uajpmf:AttributeInMetadata"
                             matchIfMetadataSilent="true"
                             onlyIfRequired="false" />
    </afp:AttributeRule>

    <afp:AttributeRule attributeID="eduPersonPrincipalName">
        <afp:PermitValueRule xsi:type="uajpmf:AttributeInMetadata"
                             matchIfMetadataSilent="true"
                             onlyIfRequired="false"
                             onlyIfChecked="true" />
    </afp:AttributeRule>

    <afp:AttributeRule attributeID="eduPersonAffiliation">
        <afp:PermitValueRule xsi:type="uajpmf:AttributeInMetadata" />
    </afp:AttributeRule>

</afp:AttributeFilterPolicy>
```

4.8 AttributeQuery プロファイルハンドラの変更

uApprove JP は Attribute Query の SAML 応答メッセージの属性のリストを利用者の同意内容に応じて生成します。

プロファイルハンドラの設定

名前空間の定義

プロファイルハンドラファイル(例えば、IDP_HOME/conf/handler.xml) 内にこのプラグインの名前空間の定義を追加する必要があります。以下のように行います:

1. <ProfileHandlerGroup> ルートの xmlns:xsi 属性の前に xmlns:uajpph="http://www.gakunin.jp/ns/uapprove-jp/profile-handler" 属性を追加
2. xsi:schemaLocation 属性の値のリストの最後に下記を追加
   http://www.gakunin.jp/ns/uapprove-jp/profile-handler classpath:/schema/shibboleth-2.0-idp-profile-handler-uapprovejp.xsd

```
...
<ph:ProfileHandlerGroup
        xmlns:ph="urn:mace:shibboleth:2.0:idp:profile-handler"
        xmlns:uajpph="http://www.gakunin.jp/ns/uapprove-jp/profile-handler"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="urn:mace:shibboleth:2.0:idp:profile-handler classpath:/schema/shibboleth-2.0-idp-profile-handler.xsd
                            http://www.gakunin.jp/ns/uapprove-jp/profile-handler classpath:/schema/shibboleth-2.0-idp-profile-handler-
uapprovejp.xsd">
...
```

プロファイルハンドラ定義の変更

Attribute Query プロファイルハンドラを変更する必要があります。以下のように行います:

1. xsi:type 属性の値を ph:SAML1AttributeQuery から uajpph:SAML1AttributeQueryUApprove に変更
2. xsi:type 属性の値を ph:SAML2AttributeQuery から uajpph:SAML2AttributeQueryUApprove に変更

```
...
    <ph:ProfileHandler xsi:type="uajpph:SAML1AttributeQueryUApprove" inboundBinding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-
binding"
                       outboundBindingEnumeration="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding">
        <ph:RequestPath>/SAML1/SOAP/AttributeQuery</ph:RequestPath>
    </ph:ProfileHandler>
...
    <ph:ProfileHandler xsi:type="uajpph:SAML2AttributeQueryUApprove" inboundBinding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP"
                       outboundBindingEnumeration="urn:oasis:names:tc:SAML:2.0:bindings:SOAP">
        <ph:RequestPath>/SAML2/SOAP/AttributeQuery</ph:RequestPath>
    </ph:ProfileHandler>
...
```

SPがブラックリストに登録されている場合の挙動について

uApprove.propertiesのservices.blacklistプロパティが true の場合、services プロパティは SP のブラックリストになります。services プロパティに正規表現を記述すると、エンティティIDがマッチした SP には uApprove を使用せずに接続します。このとき、属性の送信情報はストレージに保存しません。

uApprove JP の AttribureQuery プロファイルハンドラを設定すると、ブラックリストに登録されている SP からの AttribureQuery には、属性情報を応答しません。ただし、SP がブラックリストに登録される前に属性の送信に同意していた場合は、「今回は情報を送信することに同意します。次のログイン時に再度チェックします」で同意した場合をのぞき、同意済みの情報を応答します。

> ⚠ uApprove JP の AttribureQuery プロファイルハンドラを設定した環境では、SAML1 SP をブラックリストに登録しないでください。登録すると、IdP はSAML1 SP に属性情報を送信できなくなります。

4.9 属性送信済み SP の一覧

Shibboleth SPのインストールと設定

属性送信済み SP の一覧のWebページは REMOTE_USER を提供する Shibboleth SP で保護されている必要があります。

IdP サーバに Shibboleth SP をインストールし、以下の設定をしてください。

以下はインストールしたShibboleth SPをIdPと連携させるための設定（および対向となるShibboleth IdPの設定）の一例です。すでにShibboleth SPを稼働させている場合は従来の挙動を壊さないように配慮してください。
また、学認の技術ガイドに従ってShibboleth SPをインストールした場合は、学認メタデータを読み込まないようにすること、DSの設定を削除することに特にご注意ください。

REMOTE_USER の設定

/etc/shibboleth/shibboleth2.xml で uid を REMOTE_USER に設定します:

```
...
    <!-- The ApplicationDefaults element is where most of Shibboleth's SAML bits are defined. -->
    <ApplicationDefaults entityID="https://idp.example.ac.jp/shibboleth-sp"
                         REMOTE_USER="uid">
...
```

SSO の設定

同じマシン上の IdP に接続するため、 /etc/shibboleth/shibboleth2.xml で SSO から DS の設定を削除します:

```
...
           <SSO entityID="https://idp.example.ac.jp/idp/shibboleth">
             SAML2 SAML1
           </SSO>
...
```

IdP のメタデータ設定

/etc/shibboleth/shibboleth2.xml で同じマシン上の IdP のメタデータの読み込むための設定を追加します:

```
...
        <!-- Example of locally maintained metadata. -->
        <MetadataProvider type="XML" file="$IDP_HOME$/metadata/idp-metadata.xml" />
...
```

uid の受信設定

/etc/shibboleth/attribute-map.xml に IdP から uid を受け取る設定を追加します:

```
...
    <Attribute name="urn:mace:dir:attribute-def:uid" id="uid"/>
    <Attribute name="urn:oid:0.9.2342.19200300.100.1.1" id="uid"/>

</Attributes>
```

shibd を再起動してください。（なお、httpd の再起動は本節の最後でまとめて行います）

IdP の設定

SP のメタデータ設定

IDP_HOME/conf/relying-party.xml で同一マシンの SP のメタデータを読み込むための設定を追加します:

```
...
        <!-- Load the local SP's metadata. -->
        <metadata:MetadataProvider id="LocalSPMD" xsi:type="metadata:FilesystemMetadataProvider"
                                   metadataFile="$IDP_HOME$/metadata/sp-metadata.xml"
                                   maxRefreshDelay="P1D" />
...
```

SP のメタデータの雛形を以下のURLから入手できます:

https://idp.example.ac.jp/Shibboleth.sso/Metadata

attribute-resolver.xml の設定

IDP_HOME/conf/attribute-resolver.xml に uid の定義を追加します:

```
<resolver:AttributeDefinition xsi:type="ad:Simple" id="uid" sourceAttributeID="uid">
    <resolver:Dependency ref="myLDAP" />
    <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:mace:dir:attribute-def:uid" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:0.9.2342.19200300.100.1.1" friendlyName="uid" />
</resolver:AttributeDefinition>
```

attribute-filter.xml の設定

IDP_HOME/conf/attribute-filter.xml で uid のみを送出するように設定します:

```
<afp:AttributeFilterPolicy id="PolicyforLocalSP">
    <afp:PolicyRequirementRule xsi:type="basic:AttributeRequesterString"
                               value="https://idp.example.ac.jp/shibboleth-sp" />
    <afp:AttributeRule attributeID="uid">
        <afp:PermitValueRule xsi:type="basic:ANY"/>
    </afp:AttributeRule>
</afp:AttributeFilterPolicy>
```

id属性には他と重複しない文字列を設定してください。

uApprove.properties の設定

IdP 上で動作する SP なので、uApprove JPによる同意をスキップするために以下の設定を推奨します。

- services にエンティティ ID の正規表現を追加します。
- services.blacklist は true を設定します。

```
services                  = ^https://idp¥.example¥.ac¥.jp/shibboleth-sp$
services.blacklist        = true
```

Web アプリケーションデプロイメントデスクリプタ に ListApprovalsServlet サーブレットについての設定を追加します:

```
<web-app ...>

    ...
    <servlet>
        <servlet-name>ListConsentedSP - List of consented SP</servlet-name>
        <servlet-class>jp.gakunin.uApprovejp.lcsp.ListApprovalsServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>ListConsentedSP - List of consented SP</servlet-name>
        <url-pattern>/uApprove/ListConsentedSP</url-pattern>
    </servlet-mapping>

</web-app>
```

IdP の再デプロイが必要です。2.4 デプロイを参照してください。

httpd の設定

/etc/httpd/conf.d/ssl.conf で以下のように設定します:

```
...
<Location /idp/uApprove/ListConsentedSP>
  AuthType shibboleth
  ShibRequestSetting requireSession 1
  require valid-user
</Location>
...
```

httpd を再起動してください。

属性送信済み SP の一覧の URL

属性送信済み SP の一覧のURLは以下のようになります:

> https://idp.example.ac.jp/idp/uApprove/ListConsentedSP

終了ページの設定

終了ボタンを押すと、https://*idp.example.ac.jp*/idp/uApprove/list-approvals-exit.html に移動します。移動先は uApprove.properties の lcsp.returnURL に URL を記述して変更できます。

設定例:

```
lcsp.returnURL = https://idp.exmaple.ac.jp/your-made-page.html
```

5 トラブルシューティング

5.1 トラブルシューティング

- ERROR ないし WARN メッセージについては、 IDP_HOME/logs/idp-process.log をチェックしてください。
- $CATALINA_HOME$/logs にある Tomcat のログファイルにエラーメッセージがないかチェックしてください。

5.2 詳細なログ設定

DEBUG ないし TRACE ログレベルを有効にしたい場合は、 IDP_HOME/conf/logging.xml にて以下のように設定します:

```
<logger name="ch.SWITCH.aai.uApprove" level="DEBUG"/>
<logger name="jp.gakunin.shibboleth" level="DEBUG"/>
<logger name="jp.gakunin.uApprovejp" level="DEBUG"/>
```

A SPでの属性使用用途通知

SP 管理者は SP のメタデータに属性の使用用途を記述することで、SP の利用者に属性の使用用途 (例えば、プロファイルの初期値として使用) を uApprove JP 内で通知することができます。

A.1 設定

属性使用用途通知機能は、 <RequestedAttribute> に uajpmd:description 属性を追加する、または、 <SPSSODescriptor> の <Extensions> に <uajpmd:RequestedAttributeExtension> を追加することで利用できます。

<uajpmd:RequestedAttributeExtension> は多言語に対応しています。また、一つの属性に両方が設定されている場合は、 <uajpmd:RequestedAttributeExtension> が優先されます。

uajpmd:description

この属性は <ResquestedAttribute> にて定義します:

uajpmd:description	属性の使用用途の文字列です。

uajpmd:descrption を使用した <RequestedAttribute> の設定例:

```
<md:RequestedAttribute FriendlyName="mail"
        Name="urn:oid:0.9.2342.19200300.100.1.3"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        uajpmd:description="The mail attribute is used as the initial value of the mail address field of the registration form."/>
```

<uajpmd:RequestedAttributeExtension>

<uajpmd:RequestedAttributeExtension> は以下の必須属性と一つ以上の <uajpmd:Description> と共に設定します:

uajpmd:FriendlyName	関連づけたい <RequestedAttribute> の FriendlyName 属性の値です。

<uajpmd:Description> には属性の使用用途を記述します。<uajpmd:RequestedAttributeExtension> は以下の必須属性と共に設定します:

xml:lang	属性の使用用途の言語です。

<uajpmd:RequestedAttributeExtension> の設定例:

```
<md:EntitiesDescriptor Name="uapprovejp-dev-metadata.xml"
                    xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
                    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
                    xmlns:shibmd="urn:mace:shibboleth:metadata:1.0"
                    xmlns:uajpmd="http://www.gakunin.jp/ns/uapprove-jp/metadata"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  ...
  <md:EntityDescriptor entityID="...">
    <md:SPSSODescriptor>
      ...

      <md:Extensions>
        ...
        <uajpmd:RequestedAttributeExtension FriendlyName="mail">
          <uajpmd:Description xml:lang="en">The mail attribute is used as the initial value of the mail address field of the
registration form.</uajpmd:Description>
          <uajpmd:Description xml:lang="ja">mail 属性を登録ページのメールアドレス欄の初期値として使用します</uajpmd:Description>
        </uajpmd:RequestedAttributeExtension>
        ...
      </md:Extensions>
      ...

      <md:AttributeConsumingService index="1">
             <md:ServiceName xml:lang="en">Sample Service</md:ServiceName>

        <md:RequestedAttribute FriendlyName="eduPersonPrincipalName"
                               Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
                               NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
                               isRequired="true"/>
        <md:RequestedAttribute FriendlyName="mail"
                               Name="urn:oid:0.9.2342.19200300.100.1.3"
                               NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
      </md:AttributeConsumingService>
      ...
    </md:SPSSODescriptor>
    ...
  </md:EntityDescriptor>
  ...
</md:EntitiesDescriptor>
```