

uApproveJP-3.2.0 のインストールおよび設定方法

この文書にはuApprove Jet Pack 3.2(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 3を拡張するプラグインです。uApprove Jet Pack 2.5で提供していた機能の一部をShibboleth Identity Provider 3でも利用できるようにすることを目的としています。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、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_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

- [目次](#)
- [想定](#)
- [前提条件](#)
- [1 基本的なデプロイ](#)
 - [1.1 ライブラリのインストール](#)
 - [1.2 Velocity テンプレートファイル](#)
 - [1.3 CSS ファイル](#)
 - [1.4 設定のカスタマイズ](#)
 - [1.5 カスタムテンプレート](#)
 - [1.6 ログの設定](#)
 - [1.7 デプロイ](#)
- [2 高度なデプロイ](#)
 - [2.1 リレーショナルデータベースを用いたユーザ同意情報の保存](#)
 - [2.2 テンプレート](#)
 - [2.3 ローカライズ](#)
 - [2.4 AttributeInMetadataマッピングルール](#)
- [3 トラブルシューティング](#)
 - [3.1 トラブルシューティング](#)
 - [3.2 詳細なログ設定](#)
- [A SPでの属性使用用途通知](#)
 - [A.1 設定](#)

想定

- 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、\$CATALINA_BASE\$ (例: \$CATALINA_HOME\$)にIdPインスタンスの設定が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。

前提条件

- Shibboleth Identity Provider 3.2.0以降3.4.0未満がインストールされている必要があります。
※ Shibboleth IdP 3.4.xをお使いの場合は[uApproveJP 3.4.0](#)をお使いください。

1 基本的なデプロイ

1.1 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 $IDP_HOME$/edit-webapp/WEB-INF/lib/
```



\$IDP_HOME\$/edit-webapp/WEB-INF/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。

1.2 Velocity テンプレートファイル

属性選択画面用のVelocityテンプレートファイルをIdPのviewsディレクトリに上書きコピーします:

```
# cp $UAPPROVE_INSTALL$/manual/examples/views/intercept/* $IDP_HOME$/views/intercept/  
以下のメッセージが表示される場合がありますが、y を入力してください。  
cp: '/opt/shibboleth-idp/views/intercept/attribute-release.vm' を上書きしますか?
```

1.3 CSS ファイル

CSSファイルをIdPのedit-webappディレクトリにコピーします:

```
# cp $UAPPROVE_INSTALL$/manual/examples/edit-webapp/css/* $IDP_HOME$/edit-webapp/css/
```

1.4 設定のカスタマイズ

\$IDP_HOME\$/conf/idp.propertiesに、以下の変更を行います。idp.consent.allowPerAttributeとidp.consent.compareValuesの値をtrueに設定してください:

\$IDP_HOME\$/conf/idp.properties

```
...  
  
# Flags controlling how built-in attribute consent feature operates  
  
#idp.consent.allowDoNotRemember = true  
  
#idp.consent.allowGlobal = true  
  
idp.consent.allowPerAttribute = true  
  
  
# Whether attribute values and terms of use text are compared  
  
idp.consent.compareValues = true  
  
...
```

\$IDP_HOME\$/conf/global.xmlに、以下の属性選択画面で使用するbean定義を追加します:



挿入する場所に制限はありませんが、よく分からなければ末尾の行に</beans>という閉じタグがあると思いますので、その直前に挿入してください。

`$IDP_HOME$/conf/global.xml`

```
...  
  
<bean id="shibboleth.FallbackLanguages" parent="shibboleth.CommaDelimStringArray" c:_0="#{'%{idp.ui.fallbackLanguages:}'.trim()}" />  
  
<util:map id="shibboleth.CustomViewContext">  
    <entry key="OptionalAttributeFunction">  
        <bean class="jp.gakunin.idp.consent.logic.impl.OptionalAttributeFunction" />  
    </entry>  
    <entry key="AttributeIntendedUseFunction">  
        <bean class="jp.gakunin.idp.consent.logic.impl.AttributeIntendedUseFunction" p:defaultLanguages-ref="shibboleth.FallbackLanguages" />  
    </entry>  
</util:map>  
  
...
```

`$IDP_HOME$/system/conf/services-system.xml`に、以下の変更を行います。id="shibboleth.AttributeFilterService"のbean定義の<constructor-arg name="strategy">を以下のように変更してください:



この変更は、Shibboleth Identity Providerを再インストール(アップグレード等)する際に上書きされるため、再インストールを行った際には、再度変更を行う必要があります。

`$IDP_HOME$/system/conf/services-system.xml`

```
...  
  
<bean id="shibboleth.AttributeFilterService" class="net.shibboleth.ext.spring.service.ReloadableSpringService"  
    depends-on="shibboleth.VelocityEngine"  
    p:serviceConfigurations-ref="#{'%{idp.service.attribute.filter.resources:shibboleth.AttributeFilterResources}'.trim()}"  
    p:failFast="%{idp.service.attribute.filter.failFast:%{idp.service.failFast:false}}"  
    p:reloadCheckDelay="%{idp.service.attribute.filter.checkInterval:PT0S}"  
    p:beanFactoryPostProcessors-ref="shibboleth.PropertySourcesPlaceholderConfigurer">  
    <constructor-arg name="claz" value="net.shibboleth.idp.attribute.filter.AttributeFilter" />  
    <constructor-arg name="strategy">  
        <bean class="jp.gakunin.idp.attribute.filter.spring.impl.AttributeFilterServiceStrategy"  
id="ShibbolethAttributeFilter"/>  
    </constructor-arg>  
</bean>  
  
...
```

`$IDP_HOME$/system/flows/intercept/attribute-release-beans.xml`に、以下の変更を行います。id="IsConsentRequiredPredicate"のbean定義のclassを変更してください:



この変更は、Shibboleth Identity Providerを再インストール(アップグレード等)する際に上書きされるため、再インストールを行った際には、再度変更を行う必要があります。

```
$IDP_HOME$/system/flows/intercept/attribute-release-beans.xml
```

```
...  
<bean id="IsConsentRequiredPredicate"  
    class="jp.gakunin.idp.consent.logic.impl.IsConsentRequiredPredicate" />  
...
```

1.5 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、所属機関のロゴを変更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。

ロゴをデフォルトのOur Identity Providerから機関のロゴに変更するには、以下の手順を行ってください。



以下の記述はShibboleth IdP 3.2.1およびそれ未満向けです。3.3.0およびそれ以降では次のリンク先を参照してください。
⇒[GakuNinShare:Shibboleth IdP 3 - ロゴの変更](#)

1. ログファイル organization-logo.pngを\$IDP_HOME\$/edit-webapp/images/以下に配置します:

```
$ ls $IDP_HOME$/edit-webapp/images/  
dummylogo-mobile.png dummylogo.png organization-logo.png
```

2. \$IDP_HOME\$/messages/error-messages.propertiesのidp.logoを上記1.で配置したファイル名に変更します。なお、ファイル名は/images/から始めます。また、idp.logo.alt-textを変更します:

```
$IDP_HOME$/messages/error-messages.properties
```

```
idp.logo = /images/organization-logo.png  
idp.logo.alt-text = Organization logo
```

3. \$IDP_HOME\$/bin/build.shを実行して、\$IDP_HOME\$/war/idp.warを作り直します:

```
$ cd $IDP_HOME$  
$ sudo -u tomcat env JAVA_HOME="$(JAVA_HOME)" bin/build.sh  
Installation Directory: [/opt/shibboleth-idp]  
  
Rebuilding /opt/shibboleth-idp/war/idp.war ...  
...done  
  
BUILD SUCCESSFUL  
Total time: 16 seconds
```

1.6 ログの設定

uApprove JPのログを出力するには、\$IDP_HOME\$/conf/logback.xmlに以下を追加します:

```
$IDP_HOMES/conf/logback.xml
```

```
...
<!-- Logging level shortcuts. -->

<variable name="idp.loglevel.uApproveJP" value="INFO" />

...

<!-- ===== -->

<!-- ===== Logging Categories and Levels ===== -->

<!-- ===== -->

<logger name="jp.gakunin.idp" level="${idp.loglevel.uApproveJP:-INFO}"/>

...
```

1.7 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_HOMES
# ./bin/build.sh
Installation Directory: [/opt/shibboleth-idp]
[Enter] ←入力なし
Rebuilding /opt/shibboleth-idp/war/idp.war ...
...done

BUILD SUCCESSFUL
Total time: 16 seconds
```

`$CATALINA_BASE/conf/Catalina/localhost/idp.xml`が存在しない場合は、追加の作業として `idp.war` を `$CATALINA_BASE/webapps` にコピーします:

```
# ls $CATALINA_BASE/conf/Catalina/localhost/idp.xml
ls: cannot access /usr/java/tomcat/conf/Catalina/localhost/idp.xml: そのようなファイルやディレクトリはありません
# cp $IDP_HOMES/war/idp.war $CATALINA_BASE/webapps/
```

Tomcatを再起動します:

 CentOS 7の場合は以下のコマンドで再起動してください。

```
# systemctl restart tomcat
```

```
# service tomcat7 restart
```

2 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。

2.1 リレーショナルデータベースを用いたユーザ同意情報の保存

リレーショナルデータベース(以下、「RDB」とします)を用いて、ユーザ同意情報の保存を行うことができます。

2.1.1. MySQLの設定



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。

MySQLの設定を行います。

1. データベース shibbolethの作成

Shibboleth IdPで使用するデータベース shibbolethを作成します:



rootにパスワードが設定してあってコマンドが以下のエラーで失敗する場合は、-pオプションを追加してください。

```
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)
```

```
db$ mysql -u root
mysql>
CREATE DATABASE shibboleth;
```

2. ユーザ作成

IdPからデータベース shibbolethにアクセスするためのユーザ shibbolethを作成し、データベース shibbolethへの権限を付与します:

```
db$ mysql -u root
mysql>
CREATE USER 'shibboleth'@'localhost' IDENTIFIED BY 'shibpassword';          # ←任意のパスワード
GRANT INSERT, SELECT, UPDATE, DELETE ON shibboleth.* TO 'shibboleth'@'localhost';
```

3. StorageRecordsテーブル作成

JPAStorageServiceが使用するテーブル StorageRecordsを作成します:

```
db$ mysql -u root
mysql>
use shibboleth;
CREATE TABLE `StorageRecords` (
  `context` varchar(255) NOT NULL,
  `id` varchar(255) NOT NULL,
  `expires` bigint(20) DEFAULT NULL,
  `value` longtext NOT NULL,
  `version` bigint(20) NOT NULL,
  PRIMARY KEY (`context`,`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

2.1.3. MySQL Connector/Jのインストール

1. MySQLへのアクセスに必要なMySQL Connector/J(mysql-connector-java.jar)をインストールします:

```
# yum install mysql-connector-java
```

2. /usr/share/java配下にインストールされているので、edit-webapp/配下のlibディレクトリにシンボリックリンクを作成します:

```
# rpm -ql mysql-connector-java
(省略)
/usr/share/java/mysql-connector-java.jar
(省略)
# ln -s /usr/share/java/mysql-connector-java.jar $IDP_HOME$/edit-webapp/WEB-INF/lib/
```

3. 1.7 デプロイの手順に従って再デプロイします。

2.1.4. idp.consent.StorageServiceの設定変更

idp.consent.StorageServiceの設定をshibboleth.JPAStorageServiceに変更します:



5行目の変更は3.1まで向けのものです。3.2以降では不要です(デフォルトでサーバサイドでは無制限に記憶されます)。

\$IDP_HOMES/conf/idp.properties

Set to "shibboleth.StorageService" or custom bean for alternate storage of consent

```
idp.consent.StorageService = shibboleth.JPASStorageService
# Maximum number of consent storage records
# Set to -1 for unlimited server-side storage
idp.consent.maxStoredRecords = -1
```

2.1.5. shibboleth.JPASStorageServiceの設定

2.1.4. [idp.consent.StorageServiceの設定変更](#)でidp.session.StorageServiceに設定したshibboleth.JPASStorageServiceを定義します。

id="Shibboleth.MySQLDataSource"のbean定義のp:url, p:username, p:passwordは、[2.1.1. MySQLの設定](#)に合わせて設定します:

\$IDP_HOMES/conf/global.xml

```
<!-- Use this file to define any custom beans needed globally. -->
<bean id="shibboleth.JPASStorageService"
      class="org.opensaml.storage.impl.JPASStorageService"
      p:cleanupInterval="{idp.storage.cleanupInterval:PT10M}"
      c:factory-ref="shibboleth.JPASStorageService.entityManagerFactory" />

<bean id="shibboleth.JPASStorageService.entityManagerFactory"
      class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">
  <property name="packagesToScan" value="org.opensaml.storage.impl" />
  <property name="dataSource" ref="shibboleth.MySQLDataSource" />
  <property name="jpaVendorAdapter" ref="shibboleth.JPASStorageService.JPAVendorAdapter" />
  <property name="jpaDialect">
    <bean class="org.springframework.orm.jpa.vendor.HibernateJpaDialect" />
  </property>
</bean>

<bean id="shibboleth.JPASStorageService.JPAVendorAdapter"
      class="org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter"
      p:database="MYSQL" />

<bean id="shibboleth.MySQLDataSource"
      class="org.apache.commons.dbcp.BasicDataSource"
      p:driverClassName="com.mysql.jdbc.Driver"
      p:url="jdbc:mysql://localhost:3306/shibboleth"
      p:username="shibboleth"
      p:password="shibpassword"
      p:maxActive="10"
      p:maxIdle="5"
      p:maxWait="15000"
      p:testOnBorrow="true"
      p:validationQuery="select 1"
      p:validationQueryTimeout="5" />
```



上記はCentOS 7（apache-commons-dbcpパッケージ）での設定例です。Tomcat 8以降をお使いの場合はclass属性を以下のようにしてください。

```
class="org.apache.tomcat.dbcp.dbcp2.BasicDataSource"
```

また、p:maxActive は p:maxTotal に、p:maxWait は p:maxWaitMillis に変更してください。

2.1.6. Tomcatの再起動

Tomcatを再起動します:



CentOS 7の場合は以下のコマンドで再起動してください。

```
# systemctl restart tomcat
```

```
# service tomcat7 restart
```

2.2 テンプレート

テンプレートのカスタマイズ

`$IDP_HOME$/views/`にある Velocityテンプレートファイル、`$IDP_HOME$/edit-webapp/`にある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。Velocity 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。

Velocity については [Velocity User Guide](#) を参照してください。

2.3 ローカライズ

カスタムメッセージ



以下の記述はShibboleth IdP 3.2.1およびそれ未満向けです。3.3.0およびそれ以降では次のリンク先を参照してください。
⇒[GakuNinShare:Shibboleth IdP 3 - メッセージの多言語化](#)

`$IDP_HOME$/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。

`$IDP_HOME$/messages/`以下のファイルを追加、または修正した場合はTomcatの再起動が必要です。

`$UAPPROVE_INSTALL$/manual/examples/messages/`に、日本語環境用のテンプレートファイルが用意されています。

`$IDP_HOME$/messages/`にファイルをコピーすることで日本語メッセージを表示できるようになります:

```
# cp $UAPPROVE_INSTALL$/manual/examples/messages/* $IDP_HOME$/messages/  
# service tomcat7 restart
```

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。

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。

2.4 AttributeInMetadataマッピングルール

AttributeInMetadataマッピングルールの設定

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<AttributeConsuming Service>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マッピ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 http://www.gakunin.jp/schema/idp/gakunin-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。以下のオプション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 3.2.0 以降の 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>

```

3 トラブルシューティング

3.1 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_BASE\$/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。

3.2 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/idp.propertiesにて以下を追加します:

```
$IDP_HOME$/conf/idp.properties
```

```
idp.loglevel.uApproveJP = 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、プロフィールの初期値として使用) をuApproveJPで表示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、<SPSS0Descriptor>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<RequestedAttribute>にて定義します:

uajpmd:description	属性の使用用途の文字列です。
---------------------------	----------------

uajpmd:description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<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>

```