旧: 貴学にてIdPv3をインストールする場合の構築手順

貴学にて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を含めShibboleth IdPならびに必要なパッケージのインストール・設定を行う手順を説明します。

---

1. Shibboleth IdP (version 3.2以降) の動作要件

以下は本技術ガイドで構築する前提となる環境です。

- メモリ3GB以上
  Java実行環境への推奨割り当てメモリ量が1.5GBですので、その動作に支障がないようにしてください。
- Apache HTTP Server 2.4 以上　と mod_ssl

以下のパッケージはインストール方法も含めて以降の手順で説明します。

- Apache Tomcat 7 or 8 or 8.5 or 9
  - 初期の8.5はセッションのバグにより不安定になるという情報があります。
  - JMXを初期化前に使うと動作がおかしくなります。
  - Tomcat 8以降idp.xmlに `unpackWAR="false"` を指定していると起動に4～5倍時間がかかりますので、気になる方は指定を解除してください。
    ※いずれも以下のShibbolethのサイト「Apache Tomcat 8」が情報源です。
- Java 8 or 11
  - Java 9および10は使用できません。Shibboleth開発元はJava 9/10はサポートしない、いわゆるLTSのみをサポートしています。
  - Java 7(OpenJDK 7)はサポートされなくなりましたので、Java 8 (Oracle JDK 8 / OpenJDK 8) およびそれ以降を使用してください。
    Java 7とJava 8でスクリプトの書き方に若干の違いがあります。（書き方の違いの例）
    詳細(Shibboleth Wiki): ScriptedAttributeDefinitionの"Java 1.8 and Nashorn"の項およびその上の記述例, ScriptedDataConnector v2版: IdPJava1.8, ResolverScriptAttributeDefinition, IdPFilterRequirementScript
    （Shibboleth Wikiでは基本的にIdPv3,4のページでの説明はJava 8(Nashorn)がメインでJava 7(Rhino)も併記、v2版は特に断りがなければJava 7(Rhino)での表記となっています）

    > ⚠ 文字列置換のためのJavaScriptメソッド `"abc".replace("a", "b")` について、Java 7では全置換されていたものがJava 8では先頭の一致した部分しか置換されなくなるという情報があります。当該メソッドを使って全置換を行っている場合は正規表現 `replace(/a/g, "b")` を使うようにしてください。

  - Oracle JDK / OpenJDK 11にてLDAPサーバへの接続にLDAPSを使う場合、以下のエラーになるという情報があります。

    ```
    java.lang.NullPointerException: Thread local SslConfig has not been set
    ```

    原因はJDKのバグであるとのことです。該当する場合、以下でUnboundIDを使う回避策が提示されています。
    https://wiki.shibboleth.net/confluence/display/IDP30/LDAPonJava>8
    詳細: https://issues.shibboleth.net/jira/browse/IDP-1357
  - Java 8およびそれ以降を使う場合エントロピー不足で起動が遅くなる場合があるという情報があります。jre/lib/security/java.securityやシステムプロパティ等で対処してください。
    確認方法および手順例: IdPのサービス動作状況の確認の「よくあるエラー」の503エラーの項
    - この問題はCentOS 7を使っている場合に顕著です。
    - VMで稼働させていてこの問題がある場合、ホストマシンでHavegedを導入しVMからこれを参照する等で十分なエントロピーを生成できる場合があるようですので、合わせてご検討ください。
  - Java 8およびそれ以降を使う場合は、/etc/sysconfig/tomcatのJAVA_OPTSに指定するオプションのうち `"-XX:MaxPermSize=256m"` は意味がありません（Java 7向けの指定です）ので削除してかまいません。
  - GNU Javaは利用できません。 OpenJDKもしくはOracleのJavaを利用してください。

他の環境および最新の情報はShibbolethのサイトでご確認ください:
全体, Jetty 9.2, Jetty 9.3, Apache Tomcat 8

---

2. OSをインストールする

1. OSでの設定

・OS（CentOS 7）インストール

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)
OpenLDAP

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。

運用フェデレーション参加後に、ホスト名を変更する場合はいくつか考慮・解決すべき点があります。ホスト名は十分ご検討いただいた上で設定してください。詳しくは IdPのホスト名変更に関する注意点 をご参照ください。
※このテキストはSELinuxはPermissiveに設定されているものとして書かれております。下記コマンドでSELinux設定を確認してください。

```
$ /usr/sbin/getenforce
```

・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。

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。

---

3. jdk 8、tomcat 7をインストールする

1. 古いtomcatの削除

tomcat 6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。

2. jdk のインストール

CentOS 7にはOpenJDKのパッケージが用意されていますので、これをyumにてインストールします。

```
# yum install java-1.8.0-openjdk
```

⚠ Oracle JDKの公開バージョンは8のみですので、その手順を示します。7と8の間でバージョンをまたぐ場合は前述の通り設定ファイルに記述するスクリプトに一部違いがありますのでご注意ください。

http://java.sun.com/javase/downloads/index.jsp にあります"Java SE 8u???"の項にある"JDK"の項より構築環境に合わせてダウンロードしたパッケージ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（???は用意されているjdkのリビジョン番号にあわせて記述して下さい）。

```
# rpm -ivh jdk-8u???-linux-x64.rpm
```

上記のようにインストールした場合、パッケージ名は jdk1.8-1.8.0_???-fcs もしくは jdk1.8.0_???-1.8.0_???-fcs 、インストールパスは /usr/java/jdk1.8.0_???-amd64/ もしくは /usr/java/jdk1.8.0_???/ になります。後述のJAVA_HOMEには /usr/java/jdk1.8.0_*/jre もしくは等価なシンボリックリンクを指定してください。

8u161およびそれ以降の場合、暗号アルゴリズムの制限は解除済みですので特に操作は不要です。念のため下記 java.security の内容を確認してください。

8u151および8u152の場合、暗号アルゴリズムの制限を解除するために、/usr/java/jdk1.8.0_???/jre/lib/security/java.security に以下のように指定してください。

```
crypto.policy=unlimited
```

再度上記URLから「Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files for JDK/JRE 8」にある jce_policy-8.zip をダウンロードし、展開したREADME.txtに従って /usr/java/jdk1.8.0_???/jre/lib/security/ に local_policy.jar と US_export_policy.jar を（オリジナルをリネームした上で）配置してください。

3. tomcat 7のインストール

CentOS 7の場合、標準パッケージにTomcat 7があるため、yumにてインストールします。

```
# yum install tomcat
```

自動起動の設定

```
# systemctl enable tomcat
```

/etc/sysconfig/tomcatを編集し、JAVA_OPTSを設定します。（以下、推奨値）

```
#JAVA_OPTS="-Xminf0.1 -Xmaxf0.3"
JAVA_OPTS="-server -Xmx1500m -XX:+UseG1GC "
```

CentOS 6の場合、標準パッケージにはTomcat 7がないため、Apache Software Foundationが配布するTomcatパッケージをダウンロードしてインストールします。
/usr/javaを作成します。https://tomcat.apache.org/download-90.cgi よりダウンロードした apache-tomcat-9.?.??.tar.gz を/usr/javaに置いて、以下のコマンドを実行してください（?は用意されているtomcatのリビジョン番号にあわせて記述して下さい）。

```
# mkdir /usr/java
# tar zxv -C /usr/java -f apache-tomcat-9.?.??.tar.gz
# ln -s /usr/java/apache-tomcat-9.?.?? /usr/java/tomcat
```

自動起動スクリプトを利用すると便利です。ZIPを解凍後にtomcat7起動スクリプトファイルをコピーします。

/etc/init.d/tomcat7 を更新する場合はTomcat停止後に行なうのがお勧めです。そうでないとPIDファイル等に不整合が生じます。

ファイル内にJAVA_HOME、CATALINA_HOMEおよびCATALINA_BASEが定義されておりますので、「4. profileの修正」を参考に環境に合わせて変更してください。

Oracle(Sun) JVM / OpenJDK 以外をご使用の方は定義されているオプションを適宜調整してください。

```
# unzip tomcat7.zip
# chmod a+x tomcat7
# cp tomcat7 /etc/rc.d/init.d/
```

自動起動の設定 (このオプション指定では マイナス ‘ー’ が２つ必要です)

```
# chkconfig --add tomcat7
# chkconfig --level 345 tomcat7 on
```

"tomcat"ユーザで起動

"root"ユーザではなく、Tomcat起動用のユーザを使用することを推奨します。
ここでは、一般的な"tomcat"ユーザを作成します。（以降、"tomcat"ユーザを使用する事が前提で説明します。）

```
# useradd -r -d /usr/java/tomcat -s /sbin/nologin -c "Tomcat daemon" tomcat
```

また、起動スクリプトを修正し、"tomcat"ユーザで起動するようにします。
ダウンロードした起動スクリプトを使用する場合は以下のように修正します。（/etc/rc.d/init.d/tomcat7）

> もしTomcatが起動していれば、修正前にstopしてください。

```
# Remove -XX:MaxPermSize=256m if you are not using Sun/Oracle JVM nor OpenJDK.
export JAVA_OPTS="-server -Xmx1500m -XX:MaxPermSize=256m -XX:+UseG1GC "
export LANG=en_US.UTF-8
TOMCAT_USER=tomcat
```

以下のコマンドでその他Tomcat関連の設定ファイルやディレクトリの所有者、パーミッションを設定します。

```
# chown -R tomcat:tomcat /usr/java/tomcat/{temp,logs,work}
# chown tomcat:tomcat /usr/java/tomcat/webapps
# chmod +t /usr/java/tomcat/webapps
# chmod go+rx /usr/java/tomcat/conf
# chgrp tomcat /usr/java/tomcat/conf/*.*
# chmod g+r /usr/java/tomcat/conf/*.*
# mkdir -p /usr/java/tomcat/conf/Catalina/localhost
# chgrp -R tomcat /usr/java/tomcat/conf/Catalina
# chmod -R g+r /usr/java/tomcat/conf/Catalina
# chmod -R +t /usr/java/tomcat/conf/Catalina
# chgrp -R tomcat /usr/java/tomcat/{bin,lib}
```

また、Tomcatのpidファイル及び保存されているディレクトリを ls -dl 等で所有者・パーミッションを確認の上、必要なら変更してください。

```
# ls -dl /var/run/tomcat
drwxr-xr-x 2 root root 4096 Jun 20 16:21 /var/run/tomcat
             ↑このディレクトリが存在し、かつ所有者・グループが tomcat tomcat でなければ以下を実行
# chown -R tomcat:tomcat /var/run/tomcat
```

4. profileの追加

/etc/profile.d/java-tomcat.sh という新規ファイルを以下の内容で作成します。

> 下記のJAVA_HOMEは、OpenJDKを使ったパスとなります。
> またCATALINA_HOMEおよびCATALINA_BASEは、Apache Software Foundationが配布するTomcatパッケージをインストールした場合のパスとなります。
> 環境に合わせて変更してください。

```
# /etc/profile.d/java-tomcat.sh
JAVA_HOME=/usr/lib/jvm/jre
#export MANPATH=$MANPATH:/usr/java/default/man
CATALINA_HOME=/usr/share/tomcat
CATALINA_BASE=$CATALINA_HOME
PATH=$JAVA_HOME/bin:$CATALINA_BASE/bin:$CATALINA_HOME/bin:$PATH
export PATH JAVA_HOME CATALINA_HOME CATALINA_BASE
```

yumでインストールした場合と、rpmからインストールした場合では、ファイルの配置が違います。

```
# /etc/profile.d/java-tomcat.sh - set Java and Tomcat stuff
JAVA_HOME=/usr/lib/jvm/jre
#export MANPATH=$MANPATH:/usr/java/default/man
CATALINA_HOME=/usr/java/tomcat
CATALINA_BASE=$CATALINA_HOME
PATH=$JAVA_HOME/bin:$CATALINA_BASE/bin:$CATALINA_HOME/bin:$PATH
export PATH JAVA_HOME CATALINA_HOME CATALINA_BASE
```

追加した環境変数を読み込みます。

```
# source /etc/profile
```

5. httpd の設定

/etc/httpd/conf/httpd.conf の修正

```
（省略）
#ServerName example-idp.nii.ac.jp:80　←ホスト名
↑コメントアウト（#）を削除
（省略）
```

/etc/httpd/conf.d/ssl.conf の修正

```
（省略）
<VirtualHost _default_:443>
（省略）
#ServerName example-idp.nii.ac.jp:443　←ホスト名
↑コメントアウト（#）を削除
ProxyPass /idp/ ajp://localhost:8009/idp/ ←追加
（省略）
```

加えて、SSL 3.0プロトコルに対する攻撃が発見されておりますので、当該プロトコルを無効化することをお勧めします。⇒SSLバージョン3の脆弱性について (CVE-2014-3566)

```
SSLProtocol all -SSLv2 -SSLv3
```

/etc/httpd/conf.d/virtualhost-localhost80.conf を以下の内容で作成してください。これはShibboleth IdPが提供するreload-metadata.sh等のコマンドを使った操作を可能にするためのものです。

すでに同一のvirtual hostを別のところで定義している場合は、そちらに含めてください。また、すでに _default_:80 のVirtualHostが定義されている場合はその中の宣言が localhost:80 に適用されなくなりますので、必要であればその宣言をこのファイルにも含めてください。

default:80 が定義されているファイルに下記ProxyPassを含める方法もありますが、外部からの通常のアクセスがセキュアでない80番ポートに対しても行えることになりますので推奨しません。（もちろん、ファイアウォール等で適切に対処されていれば問題ありません）

```
<VirtualHost localhost:80>
ProxyPass /idp/ ajp://localhost:8009/idp/
</VirtualHost>
```

6. server.xmlの修正

$CATALINA_BASE/conf/server.xmlを下記のように修正します。
他の用途で使用する予定がなければ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"
           enableLookups="false" tomcatAuthentication="false" address="127.0.0.1" maxPostSize="100000" />
```

---

4. Shibbolethのインストール

各ファイル名等の指定は，Version 3.2.1に準拠しています。

1. Shibboleth IdP パッケージのダウンロード

http://shibboleth.net/downloads/identity-provider/latest/から最新版のIdP（ shibboleth-identity-provider-3.?.?.tar.gz ）をダウンロードします。

> ダウンロードしたファイルの真正性を確かめるにはPGP署名（ダウンロードURLに".asc"を追加したもの）を確認してください。

2. インストール

shibboleth-identity-provider-3.?.?.tar.gz を適当なディレクトリに置いて、以下のコマンドを実行してください。

```
# tar xzvf shibboleth-identity-provider-3.?.?.tar.gz
# cd shibboleth-identity-provider-3.?.?
# ./bin/install.sh -Didp.conf.filemode=640
```

install.shシェルスクリプトを実行すると、以下のような問い合わせがあります。
手順に従って、進めてください。

> インストール時に入力するパスワードを本運用で使う場合は、推測されにくいものを使用してください。
> ※ここで入力したパスワードは、/opt/shibboleth-idp/conf/idp.propertiesに記載されます。（平文）

```
Source (Distribution) Directory: [/root/PKG/shibboleth-identity-provider-3.1.2]
[Enter] ←入力なし
Installation Directory: [/opt/shibboleth-idp]
[Enter] ←入力なし
Hostname: [upkishib-idp.nii.ac.jp]
[Enter] ←入力なし ※表示されたホスト名が違う場合、設定してください。
SAML EntityID: [https://upkishib-idp.nii.ac.jp/idp/shibboleth]
[Enter] ←入力なし
Attribute Scope: [nii.ac.jp]
[Enter] ←入力なし ※表示されたスコープが違う場合、設定してください。


Backchannel PKCS12 Password: backpass[Enter] ←任意のパスワード

Re-enter password: backpass[Enter]

Cookie Encryption Key Password: cookiepass[Enter] ←任意のパスワード
Re-enter password: cookiepass[Enter]

  (省略)

BUILD SUCCESSFUL
Total time: 2 minutes 9 seconds
```

上記のような質問に答えながら、インストールを行います。

3. パーミッションの調整

Tomcatが”tomcat”ユーザで起動されるので、参照や書き込みが行えるようにディレクトリの所有者を変更します。
同様に、設定ファイルやメタデータの保存ディレクトリなどの所有者・パーミッションも変更します。

> ⚠ ここで設定したパーミッションをShibboleth IdPアップデート時に変更されないよう注意が必要です。詳細は IdPv3アップデートに関する情報 をご参照ください。

```
# chown -R tomcat:tomcat /opt/shibboleth-idp/logs
# chgrp -R tomcat /opt/shibboleth-idp/conf
# chmod -R g+r /opt/shibboleth-idp/conf
# find /opt/shibboleth-idp/conf -type d -exec chmod -R g+s {} ¥;
# chgrp tomcat /opt/shibboleth-idp/metadata
# chmod g+w /opt/shibboleth-idp/metadata
# chmod +t /opt/shibboleth-idp/metadata
```

> ✓ IdPが実際に使用する証明書の秘密鍵はまだ配置されておりませんので、所有者・パーミッションは後の手順で設定します。

4. jakarta-taglibs-core.jar と jakarta-taglibs-standard.jar の配置

IdPの動作に必要なjstl.jar（jakarta-taglibs-core.jar と jakarta-taglibs-standard.jar）を配置します。
CentOS6の場合、jakarta-taglibs-standardパッケージに入っているので、 yum でインストールします。

```
# yum install jakarta-taglibs-standard
```

/usr/share/java 配下にインストールされているので、edit-webapp/ 配下にシンボリックリンクを作成し、idp.warに含めます。

```
# ln -s /usr/share/java/jakarta-taglibs-core.jar /opt/shibboleth-idp/edit-webapp/WEB-INF/lib/jakarta-taglibs-core.jar
# ln -s /usr/share/java/jakarta-taglibs-standard.jar /opt/shibboleth-idp/edit-webapp/WEB-INF/lib/jakarta-taglibs-standard.jar
# /opt/shibboleth-idp/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: 3 seconds
```

5. idp.war の登録

${CATALINA_BASE}/conf/Catalina/localhost/idp.xml という新規ファイルを以下の内容で作成し、idp.warをTomcatが認識できるようにします。

```
<Context docBase=”/opt/shibboleth-idp/war/idp.war”
         privileged=”true”
         antiResourceLocking=”false”
         swallowOutput=”true”>

    <Manager pathname=”” />

    <!-- For Tomcat 8.0: work around lack of Max-Age support in IE/Edge -->
    <CookieProcessor alwaysAddExpires="true" />

</Context>
```

上記内容のうち<CookieProcessor>の行はTomcat 8.0.xの特殊な環境向けです。Tomcat 7では以下のようなログが残りますが実害はありません。

```
WARNING:   No rules found matching 'Context/CookieProcessor'.
```

同様にTomcat 9では以下のようなログが残りますが実害はありません。

```
03-Sep-2018 11:13:49.146 WARNING [main] org.apache.tomcat.util.digester.SetPropertiesRule.begin [SetPropertiesRule]{Context
/CookieProcessor} Setting property 'alwaysAddExpires' to 'true' did not find a matching property.
```

httpdの再起動とTomcatの起動を行います。（すでにTomcatが起動している場合はstopしてから行ってください）

```
# systemctl restart httpd
# systemctl start tomcat
```

```
# service httpd restart
# service tomcat7 start
```

Tomcatの起動後、${CATALINA_BASE}/logs/catalina.{日付}.log にエラーが出力されていない事を確認してください。

※catalina.{日付}.logにTomcat終了時（再起動時）のタイミングで以下のようなエラーが表示されることがありますが問題ありませんので無視してください。

```
致命的: A web application appears to have started a TimerThread named [Timer-0] via the java.util.Timer API but has failed to stop
it. To prevent a memory leak, the timer (and hence the associated thread) has been forcibly cancelled.
```

```
致命的: A web application created a ThreadLocal with key of type [null] (value [ch.qos.logback.core.
UnsynchronizedAppenderBase$1@XXXXXXXX]) and a value of type [java.lang.Boolean] (value [false]) but failed to remove it when the web
application was stopped. To prevent a memory leak, the ThreadLocal has been forcibly removed.
```

catalina.{日付}.outではなく、catalina.outに出力されます。

(関連するバグレポート)

---

5. サービスの起動・停止方法

サービス	起動コマンド	停止コマンド	再起動コマンド
httpd	systemctl start httpd	systemctl stop httpd	systemctl restart httpd
tomcat	systemctl start tomcat	systemctl stop tomcat	systemctl restart tomcat

サービス	起動コマンド	停止コマンド	再起動コマンド
httpd	service httpd start	service httpd stop	service httpd restart
tomcat	service tomcat7 start sh /usr/java/tomcat/bin/startup.sh (起動スクリプトを利用しない場合)	service tomcat7 stop sh /usr/java/tomcat/bin/shutdown.sh (起動スクリプトを利用しない場合)	service tomcat7 restart

インストールが完了したら，サイト情報等の設定を行って下さい．