IdPClusteringStickyLogin

1. 目的

Shibboleth IdPクラスタを構成したときに、stateless/memcached方式では、ログイン処理中（ログイン画面表示→ID/パスワード入力→チェック→認証済みcookie設定）にノードを移動することが問題となる。独自ホスト名を用いてこれを防ぐ方法を提示する。

- 高価なロードバランサを使わなくても適当な振り分けで大丈夫！
  「適当な振り分け」の例: DNSの委任(delegation)を用いたDSの広域分散

1.1. 検証環境

以下の環境で検証しました。

ソフトウェア	バージョン
OS	Scientific Linux 6.3
JDK	yumパッケージのOpenJDK
Tomcat	yumパッケージ(tomcat6-6.0.24-45.el6.noarch)
Shibboleth IdP	2.3.6

2. マシン構成

以下は2ノードでの構成を示すが、原理的にはN台で可能である。

- entityID: https://idp.example.ac.jp/idp/shibboleth
- ノード1: 以下の2つのホスト名を割り当てる
  1.idp.example.ac.jp
  idp.example.ac.jp
- ノード2: 以下の2つのホスト名を割り当てる
  2.idp.example.ac.jp
  idp.example.ac.jp

ノード1およびノード2で、上記entityIDでサービスするShibboleth IdP(2.3.6)を構築する。Shibbolethで用いる鍵および証明書は同一のものを用い、メタデータは上記entityIDを学認メタデータテンプレートに当てはめた標準的なもの（独自ホスト名を含まない）とする。

また、stateless clustering方式もしくはmemcached方式でクラスタ構成としておくこと。

各ノードのHTTPサーバが用いるサーバ証明書には、2つのホスト名（共通名(idp.example.ac.jp)とノード毎の名前(N.idp.example.ac.jp)）をsubjectAltNameに記載した1枚の証明書を使うと良い。ただし携帯電話でアクセスする可能性がある場合は別々の証明書とする（つまり、ホスト名1つが1つの証明書に対応するようにする）のがよい（携帯電話のブラウザの中には、証明書のCNしか見ない実装があるため）。
それぞれのホスト名で別々の証明書を使う場合は、Name-based virtual hostでなくIP-based virtual hostを使うこと。（SNI非対応ブラウザのことを考慮して）

3. 設定

各ノードで以下の設定を行う。

1. /usr/java/tomcat/webapps/idp/WEB-INF/web.xml
   のcookieDomainがコメントアウトされている行を探し、コメントアウトを解除した上で以下のように修正する。

   ```
   <!-- --> <context-param> <param-name>cookieDomain</param-name> <param-value>.idp.example.ac.jp</param-value> </context-param>
   <!-- -->
   ```

2. /etc/httpd/conf.d/ssl.conf
   を以下のように修正する。

```
ServerName 1.idp.example.ac.jp:443      ← 当該ノードの独自ホスト名を記述
UseCanonicalName On      ← Onにしていない場合は追加

...

ProxyPass /idp ajp://localhost:8009/idp
ProxyPassReverse /idp/ https://idp.example.ac.jp:443/idp/      ←  ProxyPassの行の後に追加
```

3. （オプショナル、Tomcat 6.0.27未満の場合効果なし）
   /usr/java/tomcat/conf/Catalina/localhost/idp.xml
   に以下の内容のファイルを作成する。

```
<Context sessionCookieName="JSESSIONID1" sessionCookieDomain=".idp.example.ac.jp" />
```

   JSESSIONIDの後にはノード番号（1もしくは2）を記述すること。もしすでにidp.xmlが存在する場合は、<Context>要素の属性としてsessionCookieNameおよびsessionCookieDomainを追加する。

4. 検証

各ノードでTomcatおよびhttpdを再起動し、それぞれにアクセスし以下を確認すること。

- cookieが.idp.example.ac.jpドメインとしてストアされること。
- ログイン処理中のリダイレクト/POST先URLがノードの独自ホスト名を使っていること。（この結果、ログイン処理の途中で別ノードに遷移することがなくなる）

当然ながらログイン処理中にノードが落ちれば続行不可能になるが、それはロードバランサを置いた場合も同じ。（おそらくロードバランサでは他方のノードにアクセスしエラーが表示される。Sticky Loginの場合はページ読み込みがタイムアウトする。）

5. 注意点

- ユーザに見えるログイン画面のURLが独自ホスト名になる。
- ブラウザにパスワードを記憶させる場合、各独自ホスト名に対して記憶させる必要があるかも。