TOTPを用いた多要素認証方式の導入

Shibboleth IdP 4.1以降では本体にTOTPプラグインがモジュールとして提供されておりますので、そちらの使用をお勧めします。本ページは本来4.0.x向けのものであり、4.1にて動作するように設定手順を修正したものです。

1. はじめに

Shibboleth IdPでの認証において、Google Authenticatorなど、TOTPに対応したデバイス/アプリのワンタイムパスワードを追加で必要とするように変更します。

動作確認にはTOTPに対応したデバイスもしくはアプリが必要です。

本ページで紹介したプラグインのより新しいバージョン(fork)が以下で公開されております。v3.4に対応しMFAログインフローと組み合わせられますので、こちらの利用もご検討ください。
https://github.com/joeFischetti/Shibboleth-IdP3-TOTP-Auth

本IdPプラグインの添付されたバージョンでは登録できるデバイスの数を1個のみに制限し二要素認証としての効力を確保するものとなります。登録手続きにはIPアドレス等による制限はかかりません。

このままでの運用に不都合がある場合は、登録手続きにアクセス制限をかけた上で個数制限を緩和することをご検討ください。

本TotpフローをExtendedフローの中で使用しようとしてもうまくいきません（Totpを選択してもID・パスワード欄が表示されません）。認証フローを組み合わせたい場合はMFAフローをご検討ください。

ldap.propertiesにLDAPの情報が記述されていると思いますが、LDAPのbindDNの権限で読み込みに加えて書き込みもできることを確認しておいてください。seedの保存をLDAPに対して行います。

---

2. 実習セミナーでは

以下の手順で作業を進めてください。

添付の totpauth-impl-0.5.1oncepatch-bin.zip をマシンにダウンロード、展開。（参考まで、添付のパッケージとオリジナルの差分はこちら）

conf/, edit-webapp/, flows/, views/ の内容をディレクトリ構造を保持したまま/opt/shibboleth-idp/以下に配置。

conf/authn/authn.propertiesのidp.authn.flowsに Totp を指定。

```
-idp.authn.flows = Password
+idp.authn.flows = Totp
```

conf/global.xmlの末尾（</beans>の上）に以下を挿入します。

```
        <bean id="authn/Totp" parent="shibboleth.AuthenticationFlow"
                p:passiveAuthenticationSupported="true"
                p:forcedAuthenticationSupported="true">
            <property name="supportedPrincipals">
                <util:list>
                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
                        c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:TimeSyncToken" />
<!--
                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
                        c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level2" />
                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
                        c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level1" />
-->
                </util:list>
            </property>
        </bean>
```

※ 他の活用編メニューにてLevel1/Level2の認証強度の設定を実施済みの場合は、コメントアウトを解除してTOTPをLevel2相当としてください。

conf/authn/authn.propertiesにパスワードを持ち回る必要があるため以下の行を修正します。

```
@@ -42,7 +42,7 @@
 # Override this and removeAfterValidation to require all validators to succeed
 #idp.authn.Password.requireAll = false
 # Override to keep the password around
-#idp.authn.Password.removeAfterValidation = true
+idp.authn.Password.removeAfterValidation = false
 # Override to store password in Java Subject
 #idp.authn.Password.retainAsPrivateCredential = false
 # Simple username transforms before validation
```

/opt/shibboleth-idp/bin/build.shを実行し、WARファイル再作成およびデプロイします。
最後にJettyの再起動を行います。

```
# /opt/shibboleth-idp/bin/build.sh
Buildfile: /opt/shibboleth-idp/bin/build.xml
build-war:
Installation Directory: [/opt/shibboleth-idp] ?
[Enter]  ←入力なし

# systemctl restart jetty
```

---

3. 手順書

以下は、利用するTOTPプラグインの開発元のURLです。詳細にご興味がある方はご参照ください。
Shibboleth IdPバージョン3.2対応のものであるため、4以降で実行するためには上記のパッケージおよび手順が必要となります。

参考: https://github.com/korteke/Shibboleth-IdP3-TOTP-Auth

---

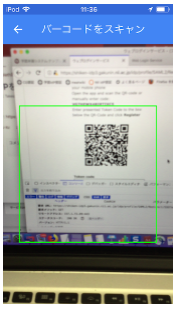
4. 動作確認

以下が登録手順です。

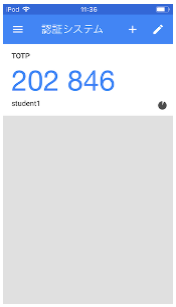
1. 接続確認用SPから各自が使用するIdPを選択します。（どのSPからの認証要求でもTOTP認証フローが実行されます）
2. 第一画面は通常と変わりませんので、通常のUsername/Passwordを入力して次に進みます。
3. "Token code"というワンタイムパスワードを入力する画面になりますが、まだ登録していませんので、画面に指示のある通り任意の6桁の数字を入力し「Login」ボタンを押してください。
4. 同じ画面の下に"Register a new Token"ボタンが現れますのでクリックします。そうすると登録のためのQRコードが表示されます。
5. TOTP対応アプリ（例: Google Authenticator）を起動します。
6. 「設定を開始」をタップします。

7. 「バーコードをスキャン」をタップします。

8. 登録画面に表示されているQRコードを読み取ります。



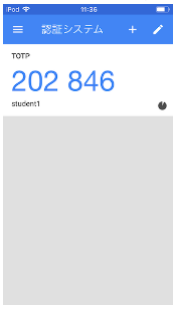
9. 登録後に数字6桁のワンタイムパスワードが表示されますので、IdPのQRコードの下にある"Token code"欄に入力し"Register"ボタンをクリックします。



以下が動作確認手順です。

1. 上記登録直後はワンタイムパスワード入力画面になりますが、入力せずに、再度接続確認用SPから各自が使用するIdPを選択します。
2. 第一画面は通常と変わりませんので、通常のUsername/Passwordを入力して次に進みます。

3. "Token code"というワンタイムパスワードを入力する画面になりますので、TOTP対応アプリ/デバイスに表示されている数字6桁のワンタイムパスワードを入力して次に進みます。



4. 通常通り送信属性同意画面および属性受信の確認ページが表示されます。
5. ページ下部の「セッション情報」をクリックして、以下のように認証手段がパスワード認証時と異なることを確認してください。

```
Authentication Context Class: urn:oasis:names:tc:SAML:2.0:ac:classes:TimeSyncToken
```

（もしくは他の活用編メニューを実施していればLevel1ないしLevel2）