MultiFactor認証フロー(MFA)を用いた認証設定

Shibboleth IdP 3.3より導入されたMultiFactor認証フロー(MFA)の認証設定についてのドキュメントです。本ドキュメントはSAML 2.0で認証の切り替えを行うことを目的としており、SAML1は対象外です（LevelXを用いた認証要求はできません）。

> ⚠ SAML1を使うことにより本設定の制約を迂回できることを避けるため、SPにおいてはshibboleth2.xmlにてSAML1の機能を無効化することをお勧めします。

MultiFactor認証フローは、シンプルないし複雑な認証シーケンスを作るために複数の認証フローを組み合わせるスクリプト記述可能な方法を提供します。

共通設定 (General Configuration)

MultiFactor認証フローの設定は、`conf/authn/mfa-authn-config.xml`で行います。

また、`conf/idp.properties`の`idp.authn.flows`でMultiFactor認証フローを有効にします。注意すべき点として、MultiFactor認証フローからルールやスクリプトを介して呼び出される認証フローについては、意図していない方法で当該認証フローが実行されるかもしれないため、`idp.authn.flows`では有効にすべきではないとされています。

conf/idp.properties

```
 # Regular expression matching login flows to enable, e.g. IPAddress|Password
-idp.authn.flows= Password
+idp.authn.flows= MFA
```

直接的なフロー選択 (Directly Selecting Flows)

もっと簡単なルールは、最初の認証フローが成功した場合に、次に実行する認証フローを指定します。

下記の例は、最初にPassword認証フローによる認証を行い、Password認証フローの認証が成功した場合にX509認証フローの認証が行われます。X509認証フローの認証が成功すると認証成功となります。

conf/authn/mfa-authn-config.xml

```
<util:map id="shibboleth.authn.MFA.TransitionMap">
    <!-- Run authn/Password first. -->
    <entry key="">
        <bean parent="shibboleth.authn.MFA.Transition" p:nextFlow="authn/Password" />
    </entry>

    <!-- If that returns "proceed", run authn/X509 next. -->
    <entry key="authn/Password">
        <bean parent="shibboleth.authn.MFA.Transition" p:nextFlow="authn/X509" />
    </entry>

    <!-- An implicit final rule will return whatever the second flow returns. -->
</util:map>
```

プログラムでのフロー選択 (Programmatically Selecting Flows)

より複雑なルールを実現するには、Script、Spring Expression、もしくはJavaで記述した関数を実行します。

下記の例は、以下の認証シーケンスを実現しています。

1. 最初にPassword認証フローを実行します。
2. ステップ1のPassword認証フローが認証成功し認証要求を満たすのに十分であればステップ3を、そうでなければステップ5に遷移します。
3. ステップ1によって識別されたユーザの属性 `allowedLoginMethods`を取得します。

4. 属性 allowedLoginMethodsが存在し、かつ属性値に**Password**が含まれていれば、Password認証フローのみで認証成功になります。そうでなければステップ5に遷移します。
5. X509認証フローを実行します。
6. X509認証フローが認証成功すれば、Password認証フローとX509認証フローの認証結果が一つにマージされます。
7. 両方の認証フローが認証成功であれば認証成功となり、そうでなければ認証失敗となります。

なお、conf/attribute-resolver.xmlに属性allowedLoginMethodsを追加する必要があります。

conf/authn/mfa-authn-config.xml

```
<util:map id="shibboleth.authn.MFA.TransitionMap">
    <!-- Run authn/Password first. -->
    <entry key="">
        <bean parent="shibboleth.authn.MFA.Transition" p:nextFlow="authn/Password" />
    </entry>

    <!--
    Second rule runs a function if authn/Password succeeds, to determine whether an additional
    factor is required.
    -->
    <entry key="authn/Password">
        <bean parent="shibboleth.authn.MFA.Transition" p:nextFlowStrategy-ref="checkSecondFactor" />
    </entry>

    <!-- An implicit final rule will return whatever the second flow returns. -->
</util:map>

<!-- Example script to see if second factor is required. -->
<bean id="checkSecondFactor" parent="shibboleth.ContextFunctions.Scripted" factory-method="inlineScript"
        p:customObject-ref="shibboleth.AttributeResolverService">
    <constructor-arg>
        <value>
        <![CDATA[
            nextFlow = "authn/X509";

            // Go straight to second factor if we have to, or set up for an attribute lookup first.
            authCtx = input.getSubcontext("net.shibboleth.idp.authn.context.AuthenticationContext");
            mfaCtx = authCtx.getSubcontext("net.shibboleth.idp.authn.context.MultiFactorAuthenticationContext");
            if (mfaCtx.isAcceptable()) {
                // Attribute check is required to decide if first factor alone is enough.
                resCtx = input.getSubcontext(
                    "net.shibboleth.idp.attribute.resolver.context.AttributeResolutionContext", true);

                // Look up the username
                usernameLookupStrategyClass = Java.type("net.shibboleth.idp.session.context.navigate.
CanonicalUsernameLookupStrategy");
                usernameLookupStrategy = new usernameLookupStrategyClass();
                resCtx.setPrincipal(usernameLookupStrategy.apply(input));


                // resolve the attribute to determine if a first factor is sufficient
                resCtx.getRequestedIdPAttributeNames().add("allowedLoginMethods");
                resCtx.resolveAttributes(custom);

                // Check for an attribute value that authorizes use of first factor.
                attribute = resCtx.getResolvedIdPAttributes().get("allowedLoginMethods");
                valueType =  Java.type("net.shibboleth.idp.attribute.StringAttributeValue");
                if (attribute != null && attribute.getValues().contains(new valueType("Password"))) {
                    nextFlow = null;
                }

                input.removeSubcontext(resCtx);   // cleanup
            }

            nextFlow;   // pass control to second factor or end with the first
        ]]>
    </value>
    </constructor-arg>
</bean>
```

遷移の完全なコントロール (Full Control Over Transitions)

最も複雑なルールを実現するために、Spring WebFlowイベントに基づいて完全に遷移を制御できます。

下記の例ではShibboleth IdPバージョン2向けにNIIと金沢大学で共同開発したGUARDプラグインと同様な認証シーケンスを実現します。GUARDプラグインについては、2015年2月に金沢大学の松平様が発表された『大学統合認証基盤における多要素認証について』の12ページをご参照ください。下記の例での認証コンテクストと認証フローの関係を以下に示します。

認証コンテクスト	略称	認証フロー
urn:mace:gakunin.jp:idprivacy:ac:classes:Level1	Level1	パスワード
urn:mace:gakunin.jp:idprivacy:ac:classes:Level2	Level2	(機関内) パスワード **OR** RemoteUser **OR** X509 (機関外) RemoteUser **OR** X509
urn:mace:gakunin.jp:idprivacy:ac:classes:Level3	Level3	RemoteUser **AND** X509

制限事項

- Level1のパスワード認証後に、Level2のパスワード認証はSSOされません。
- Level2のRemoteUser認証後に、Level3のRemoteUser認証はSSOされません。(ただし、BASIC認証に関して言えばブラウザが自動的にユーザー名とパスワードを送信するためエンドユーザーがユーザー名やパスワードを再度入力する必要はありません)

設定

1. conf/authn/mfa-authn-config.xmlで各Levelに応じた認証設定を行います。

conf/authn/mfa-authn-config.xml

```
<util:map id="shibboleth.authn.MFA.TransitionMap">
    <!-- First rule calls a flow to display a view to select a method to run. -->
    <entry key="">
        <bean parent="shibboleth.authn.MFA.Transition" p:nextFlow="custom/methodChooser" />
    </entry>

    <!-- Second rule decides what to call based on event signaled by the view. -->
    <entry key="custom/methodChooser">
        <bean parent="shibboleth.authn.MFA.Transition">
            <property name="nextFlowStrategyMap">
                <map>
                    <!-- Maps event to a flow -->
                    <!-- Level1 -->
                    <entry key="ChooseLevel1" value="authn/Level1" />

                    <!-- Level2 -->
                    <entry key="ChoosePassword"   value="authn/Password" />
                    <entry key="ChooseRemoteUser" value="authn/RemoteUser" />
                    <entry key="ChooseX509"       value="authn/X509" />


                    <!-- Level3 -->
                    <entry key="ChooseLevel3" value="authn/RemoteUser4Level3" />
                </map>
            </property>
        </bean>
    </entry>

    <!-- Level3 -->
    <entry key="authn/RemoteUser4Level3">
        <bean parent="shibboleth.authn.MFA.Transition" p:nextFlow="authn/X509" />
    </entry>

    <!-- An implicit final rule will return whatever the final flow returns. -->
</util:map>
```

   Level2, Level3で異なる認証フローを使用したい場合は16行目から23行目、および31行目の<entry>を変更します。
2. 下記ファイルをダウンロードして配置します。

ファイル名	配置先ディレクトリ
methodChooser-flow.xml	flow/custom/methodChooser/
chooser.vm	views/

Level2, Level3で異なる認証フローを使用したい場合は、methodChooser-flow.xmlの30行目の Password|RemoteUser|X509 の部分と、37行目から40行目の<transition>、および47行目から49行目の<end-state>を変更します。

methodChooser-flow.xml

```
    <view-state id="DisplayChooserWebViewForLevel2" view="chooser">
        <on-render>
            <evaluate expression="environment" result="viewScope.environment" />
            <evaluate expression="opensamlProfileRequestContext" result="viewScope.profileRequestContext" />
            <evaluate expression="opensamlProfileRequestContext.getSubcontext(T(net.shibboleth.idp.authn.context.
AuthenticationContext))" result="viewScope.authenticationContext" />
            <evaluate expression="authenticationContext.getAvailableFlows().values().?[id matches 'authn/
(Password|RemoteUser|X509)']" result="viewScope.availableAuthenticationFlows" />
            <evaluate expression="authenticationContext.getSubcontext(T(net.shibboleth.idp.ui.context.
RelyingPartyUIContext))" result="viewScope.rpUIContext" />
            <evaluate expression="T(net.shibboleth.utilities.java.support.codec.HTMLEncoder)" result="viewScope.encoder" />
            <evaluate expression="flowRequestContext.getExternalContext().getNativeRequest()" result="viewScope.request" />
            <evaluate expression="flowRequestContext.getExternalContext().getNativeResponse()" result="viewScope.response" />
            <evaluate expression="flowRequestContext.getActiveFlow().getApplicationContext().containsBean('shibboleth.
CustomViewContext') ? flowRequestContext.getActiveFlow().getApplicationContext().getBean('shibboleth.CustomViewContext') :
null" result="viewScope.custom" />
        </on-render>

        <transition on="ChoosePassword"   to="ChoosePassword" />
        <transition on="ChooseRemoteUser" to="ChooseRemoteUser" />
        <transition on="ChooseX509"       to="ChooseX509" />
    </view-state>

    <!-- Level1 -->
    <end-state id="ChooseLevel1" />

    <!-- Level2 -->
    <end-state id="ChoosePassword" />
    <end-state id="ChooseRemoteUser" />
    <end-state id="ChooseX509" />

    <!-- Level3 -->
    <end-state id="ChooseLevel3" />
```

3. Level1のパスワード認証を用意します。

```
# mkdir -p flows/authn/Level1
# cp system/flows/authn/password-authn-flow.xml Level1-flow.xml
# cp system/flows/authn/password-authn-beans.xml Level1-beans.xml
# sed -i 's/password-authn-beans.xml/Level1-beans.xml/' Level1-flow.xml
```

4. Level3のRemoteUser認証を用意します。

```
# mkdir -p flows/authn/RemoteUser4Level3
# cp system/flows/authn/remoteuser-authn-flow.xml flows/authn/RemoteUser4Level3/RemoteUser4Level3-flow.xml
# cp system/flows/authn/remoteuser-authn-beans.xml flows/authn/RemoteUser4Level3/RemoteUser4Level3-beans.xml
# sed -i 's/remoteuser-authn-beans.xml/RemoteUser4Level3-beans.xml/' flows/authn/RemoteUser4Level3/RemoteUser4Level3-flow.xml
# sed -i 's/remoteuser-authn-config.xml/RemoteUser4Level3-config.xml/' flows/authn/RemoteUser4Level3/RemoteUser4Level3-beans.
xml

# cp conf/authn/remoteuser-authn-config.xml conf/authn/RemoteUser4Level3-config.xml
# sed -i 's,Authn/RemoteUser,Authn/RemoteUser4Level3,' conf/authn/remoteuser-authn-config.xml

# cp webapp/WEB-INF/web.xml edit-webapp/WEB-INF/web.xml
```

edit-webapp/WEB-INF/web.xml

```
 @@ -113,4 +113,15 @@
     </servlet-mapping>

+    <!-- Servlet protected by container used for RemoteUser authentication (Level3) -->
+    <servlet>
+        <servlet-name>RemoteUser4Level3AuthHandler</servlet-name>
+        <servlet-class>net.shibboleth.idp.authn.impl.RemoteUserAuthServlet</servlet-class>
+        <load-on-startup>2</load-on-startup>
+    </servlet>
+    <servlet-mapping>
+        <servlet-name>RemoteUser4Level3AuthHandler</servlet-name>
+        <url-pattern>/Authn/RemoteUser4Level3</url-pattern>
+    </servlet-mapping>
+
     <!-- Servlet protected by container used for X.509 authentication -->
     <servlet>
```

```
# bin/build.sh
```

5. Shibboleth IdP 3の高度な認証設定に従い、conf/authn/general-authn.xmlの設定をします。

conf/authn/general-authn.xml

```
@@ -54,21 +54,89 @@
          <bean id="authn/External" parent="shibboleth.AuthenticationFlow"
             p:nonBrowserSupported="false" />

+        <bean id="authn/Level1" parent="shibboleth.AuthenticationFlow">
+            <property name="supportedPrincipals">
+                <list>
+                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
+                        c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level1" />
+                </list>
+            </property>
+        </bean>
+
+        <bean id="authn/Level2" parent="shibboleth.AuthenticationFlow">
+            <property name="supportedPrincipals">
+                <list>
+                    <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" />
+                </list>
+            </property>
+        </bean>
+
+        <bean id="authn/Level3" parent="shibboleth.AuthenticationFlow">
+            <property name="supportedPrincipals">
+                <list>
+                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
+                        c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level3" />
+                    <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" />
+                </list>
+            </property>
+        </bean>
+
+        <bean id="authn/Password" parent="shibboleth.AuthenticationFlow"
+                p:passiveAuthenticationSupported="true"
+                p:forcedAuthenticationSupported="true">
+            <property name="supportedPrincipals">
+                <list>
```

```
+                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
+                        c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport" />
+                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
+                        c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:Password" />
+                    <bean parent="shibboleth.SAML1AuthenticationMethod"
+                        c:method="urn:oasis:names:tc:SAML:1.0:am:password" />
+                    <!-- GUARD -->
+                    <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" />
+                </list>
+            </property>
+        </bean>
+
         <bean id="authn/RemoteUser" parent="shibboleth.AuthenticationFlow"
-            p:nonBrowserSupported="false" />
+            p:nonBrowserSupported="false">
+            <property name="supportedPrincipals">
+                <list>
+                    <!-- GUARD -->
+                    <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" />
+                </list>
+            </property>
+        </bean>

         <bean id="authn/RemoteUserInternal" parent="shibboleth.AuthenticationFlow" />

         <bean id="authn/X509" parent="shibboleth.AuthenticationFlow"
                 p:nonBrowserSupported="false">
             <property name="supportedPrincipals">
                 <list>
                     <bean parent="shibboleth.SAML2AuthnContextClassRef"
                         c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:X509" />
                     <bean parent="shibboleth.SAML2AuthnContextClassRef"
                         c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:TLSClient" />
                     <bean parent="shibboleth.SAML1AuthenticationMethod"
                         c:method="urn:ietf:rfc:2246" />
+                    <!-- GUARD -->
+                    <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" />
                 </list>
             </property>
         </bean>

@@ -89,3 +157,14 @@
-        <bean id="authn/Password" parent="shibboleth.AuthenticationFlow"
-                p:passiveAuthenticationSupported="true"
-                p:forcedAuthenticationSupported="true" />
+        <bean id="authn/RemoteUser4Level3" parent="shibboleth.AuthenticationFlow"
+                p:nonBrowserSupported="false">
+            <property name="supportedPrincipals">
+                <list>
+                    <!-- GUARD -->
+                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
+                        c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level3" />
+                    <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" />
+                </list>
+            </property>
+        </bean>
@@ -112,22 +191,29 @@
         <bean id="authn/MFA" parent="shibboleth.AuthenticationFlow"
                 p:passiveAuthenticationSupported="true"
```

```
                p:forcedAuthenticationSupported="true">
            <!--
            The list below almost certainly requires changes, and should generally be the
            union of any of the separate factors you combine in your particular MFA flow
            rules. The example corresponds to the example in mfa-authn-config.xml that
            combines IPAddress with Password.
            -->
            <property name="supportedPrincipals">
                <list>
                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
                        c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocol" />
                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
                        c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport" />
                    <bean parent="shibboleth.SAML2AuthnContextClassRef"
                        c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:Password" />
                    <bean parent="shibboleth.SAML1AuthenticationMethod"
                        c:method="urn:oasis:names:tc:SAML:1.0:am:password" />
+                   <!-- GUARD -->
+                   <bean parent="shibboleth.SAML2AuthnContextClassRef"
+                       c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level1" />
+                   <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:Level3" />
                </list>
            </property>
        </bean>
```

6. Level2のPassword認証フローに機関のIPアドレスレンジ(下記例では203.0.113.0/24)を設定します。

conf/authn/general-authn.xml

```
@@ -90,3 +90,4 @@
        <bean id="authn/Password" parent="shibboleth.AuthenticationFlow"
+               p:activationCondition-ref="authn.PasswordActivationCondition"
                p:passiveAuthenticationSupported="true"
                p:forcedAuthenticationSupported="true">
@@ -241,1 +242,9 @@
+
+    <!--
+    Activation Condition
+    -->
+    <bean id="authn.PasswordActivationCondition" class="org.opensaml.profile.logic.IPRangePredicate"
+            p:httpServletRequest-ref="shibboleth.HttpServletRequest"
+            p:ranges="#{ '203.0.113.0/24' }" >
+    </bean>
 </beans>
```

参考

- [Shibboleth wiki] MultiFactorAuthnConfiguration