2024年11月8日

セキュリティレベルを設定した

SPに対する認証﻿

国立情報学研究所

教育研修事業

**目次**

[**1. はじめに 2**](#_heading=h.ylkx8xjzwzn8)

[**2. 実習セミナーでは 3**](#_heading=h.3znysh7)

[**3. 手順書 8**](#_heading=h.3dy6vkm)

[**4. 動作確認 10**](#_heading=h.1t3h5sf)

# **1. はじめに**

※前提として、活用編メニュー「クライアント証明書認証を使った認証」が実施済みであるとします。

本メニューでは、IdPとSPを共にカスタマイズします。

IdPには、ID/パスワード認証とクライアント証明書認証を使って、セキュリティレベルに応じた認証方式を設定します。

またSPには、認証のセキュリティレベルを設定します。

上記の設定により、セキュリティレベルが低くて問題ないSPにはID/パスワード認証でアクセスが行えるが、セキュリティレベルが高いSPにアクセスした場合、認証済みであってもクライアント証明書認証などセキュアな認証方式を行わなければアクセスできないようになります。

なお、本メニュー（認証フローの階層化）に依らなくても、SP側でIdPのauthn.propertiesに記載されているURIを直接指定する（SP設定の ShibRequestSetting authnContectClassRef ... に指定する）ことによってIdPが定義している特定の認証手段を用いることを強制することは可能です。ただしSP側がIdPの設定内容を熟知している必要があります。

# 

# **2. 実習セミナーでは**

以下のような設定で行います。

手順書と照らし合わせながら、作業を進めてください。

＜IdP側の設定＞

* MFAモジュールの有効化  
  以下のコマンドを実行してMultiFactor認証フローのIdPモジュールを有効化します。

|  |
| --- |
| # /opt/shibboleth-idp/bin/module.sh -t idp.authn.MFA || /opt/shibboleth-idp/bin/module.sh -e idp.authn.MFA |

また、以下のコマンドを実行してJavaScriptエンジンをプラグインとしてインストールします。

|  |
| --- |
| # /opt/shibboleth-idp/bin/plugin.sh -I net.shibboleth.idp.plugin.nashorn  # /opt/shibboleth-idp/bin/plugin.sh -l |

* /opt/shibboleth-idp/conf/authn/authn.properties の変更

idp.authn.flows にMFAを設定して、MFAフローを有効化します。

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

* /opt/shibboleth-idp/conf/authn/authn.properties の変更

パスワード認証、クライアント認証、MFA認証フローの設定を変更します。

|  |
| --- |
| （省略）  Password認証がlevel1を満たすとする設定  idp.authn.Password.supportedPrincipals = \←#を外す  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport, \←#を外す  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:Password, \←#を外す  saml2/urn:mace:gakunin.jp:idprivacy:ac:classes:Level1, \←挿入  saml1/urn:oasis:names:tc:SAML:1.0:am:password←#を外す  （省略）  クライアント証明書認証がlevel1, 2を満たすとする設定  <赤字の部分を挿入>  idp.authn.RemoteUser.supportedPrincipals = \  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:X509, \  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:TLSClient, \  saml2/urn:mace:gakunin.jp:idprivacy:ac:classes:Level2, \←挿入  saml2/urn:mace:gakunin.jp:idprivacy:ac:classes:Level1, \←挿入  saml1/urn:ietf:rfc:2246  （省略）  MFAフローがlevel1, 2を満たすとする設定  idp.authn.MFA.supportedPrincipals = \  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:InternetProtocol, \  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport, \  saml2/urn:oasis:names:tc:SAML:2.0:ac:classes:Password, \  saml2/urn:mace:gakunin.jp:idprivacy:ac:classes:Level2, \←挿入  saml2/urn:mace:gakunin.jp:idprivacy:ac:classes:Level1, \←挿入  saml1/urn:oasis:names:tc:SAML:1.0:am:password  （省略） |

* /opt/shibboleth-idp/conf/relying-party.xml の変更

SAML2.SSOにdefaultAuthenticationMethodsプロパティを設定します。

|  |
| --- |
| （省略）  <bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">  <property name="profileConfigurations">  <list>  <!-- SAML 1.1 and SAML 2.0 AttributeQuery are disabled by default. -->  <!--  <ref bean="Shibboleth.SSO" />  <ref bean="SAML1.AttributeQuery" />  <ref bean="SAML1.ArtifactResolution" />  -->  <bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release">  <property name="defaultAuthenticationMethods">  <list>  <bean parent="shibboleth.SAML2AuthnContextClassRef"  c:classRef="urn:mace:gakunin.jp:idprivacy:ac:classes:Level1" />  </list>  </property>  </bean>  <ref bean="SAML2.ECP" />  <ref bean="SAML2.Logout" />  <!--  <ref bean="SAML2.AttributeQuery" />  -->  <ref bean="SAML2.ArtifactResolution" />  </list>  </property>  </bean>  （省略） |

* /opt/shibboleth-idp/conf/authn/mfa-authn-config.xml の変更

MFA認証フローの動作を設定します。

* 「<util:map id="shibboleth.authn.MFA.TransitionMap">」と「<bean id="checkSecondFactor" parent="shibboleth.ContextFunctions.Scripted" factory-method="inlineScript">」のブロックを以下の通り置き換えます。

|  |
| --- |
| <util:map id="shibboleth.authn.MFA.TransitionMap">  <entry key="">  <bean parent="shibboleth.authn.MFA.Transition" p:nextFlowStrategy-ref="checkAuthnFactor" />  </entry>    </util:map>    <bean id="checkAuthnFactor" parent="shibboleth.ContextFunctions.Scripted" factory-method="inlineScript">  <constructor-arg>  <value>  <![CDATA[  nextFlow = "authn/Password";    authnContextClassPrincipal = Java.type("net.shibboleth.idp.saml.authn.principal.AuthnContextClassRefPrincipal");  level2 = new authnContextClassPrincipal("urn:mace:gakunin.jp:idprivacy:ac:classes:Level2");    rpid = input.getSubcontext("net.shibboleth.idp.profile.context.RelyingPartyContext").relyingPartyId;  authCtx = input.getSubcontext("net.shibboleth.idp.authn.context.AuthenticationContext");  rprinCtx = authCtx.getSubcontext("net.shibboleth.idp.authn.context.RequestedPrincipalContext");  rpLevel = rprinCtx && rprinCtx.getRequestedPrincipals();    if(rpLevel && rpLevel.contains(level2)) {  nextFlow = "authn/RemoteUser";  }    nextFlow;      ]]>  </value>  </constructor-arg>  </bean> |

＜SP側の設定＞

* SP毎にセキュリティレベルを設定

各自作成したSPの認証をセキュリティレベル２に設定し、クライアント証明書を使った認証を要求するよう設定します。

以下のように/etc/httpd/conf.d/shib.confを変更します。

「/secure」に設定している内容に、IdPで設定したセキュリティレベル2を指定する設定を追加します。

|  |
| --- |
| <Location /secure>  AuthType shibboleth  ShibRequestSetting requireSession 1  ShibRequestSetting authnContextClassRef urn:mace:gakunin.jp:idprivacy:ac:classes:Level2  require shib-session  </Location> |

# 

# **3. 手順書**

以下は、MultiFactor認証フロー(MFA)を用いた認証設定のURLです。手順の詳細にご興味がある方はご参照ください。

[設定手順書](https://meatwiki.nii.ac.jp/confluence/pages/viewpage.action?pageId=26186832)

参考：認証フローを実習課題に対応させた一般的説明は以下の通りです。

authn.propertiesの「supportedPrincipalsプロパティ」の設定

|  |  |
| --- | --- |
| 認証フロー | supportedPrincipalsプロパティ |
| Password | Level1 |
| RemoteUser | Level2, Level1 |

SPが要求する認証レベルが次のようになっているとして、認証の挙動を説明します。

|  |  |
| --- | --- |
| SP | 要求する認証 |
| SPa | なし |
| SPb | Level1 |
| SPc | Level2 |

**認証の挙動**

* 未認証の場合

1. SPaもしくは SPbからIdPにリダイレクトされると、Level1のPassword認証フローのログインページが表示されます。

2. SPcからIdPにリダイレクトされると、Level2のRemoteUser認証フローのためのログインページやダイアログが表示されます。

* Level1が認証済みの場合

1. SPaもしくはSPbからIdPにリダイレクトされると、Level1のPassword認証フローが認証済みのためユーザ同意画面もしくは認証後のSPの画面が表示されます。

2. SPcからIdPにリダイレクトされると、Level2のRemoteUser認証フローのためのログインページやダイアログが表示されます。

* Level2が認証済みの場合

1. SPaもしくはSPbからIdPにリダイレクトされると、Level2のRemoteUser認証フローが認証済みのためユーザ同意画面もしくは認証後のSPの画面が表示されます。

2. SPcからIdPにリダイレクトされると、Level2のRemoteUser認証フローが認証済みのためユーザ同意画面もしくはSPの画面が表示されます。

# **4. 動作確認**

※確認手順の説明に記載されている「動作確認用のSP」は、現在使用しているフェデレーションによってアクセス先が変わります。（どちらかのフェデレーションに参加して利用できる状態にしておいてください。）

|  |  |
| --- | --- |
| 実習セミナーフェデレーション | <https://ex-sp.gakunin.nii.ac.jp/> |
| テストフェデレーション | <https://test-sp1.gakunin.nii.ac.jp/> |

① 設定後、IdPはJetty、SPはApacheの再起動を行ってない場合は行ってください。

|  |
| --- |
| IdP側  systemctl restart jetty  SP側  systemctl restart httpd |

② 各自が使用するSPの接続確認用ページにアクセスします。

|  |
| --- |
| 例）1番を割り振られた場合  　https://ex-sp-test01.gakunin.nii.ac.jp/ |

③ ログインボタンをクリックします。

④ DSの設定を行っている場合、所属機関の選択画面が表示されるので、各自が使用するIdPを選択します。

|  |
| --- |
| 例）1番を割り振られた場合  　実習セミナーIdPテスト01 |

⑤ 個人証明書の要求というダイアログが表示されるので、対象となるクライアント証明書を選択して、OKボタンをクリックします。

※送信属性同意画面が表示される場合は、そのまま設定値を送信します。

⑥ 正しく属性受信の確認ページが表示される事を確認してください。

※ID/パスワードを入力するログイン画面は表示されず、クライアント証明書で認証が行われ、ログインする事ができます。

⑦ 次に上記動作確認用のSPにアクセスすると、パスワード等の要求がなくシングルサインオンにてログインできます。

⑧ 一度ブラウザを閉じて、再度動作確認用のSPにアクセスします。

⑨ セキュリティレベルが動作確認用SPには設定されていないため、進めていくとデフォルト設定のLoginボタン（ID/パスワード）となります。ID/パスワードを入力してログインしてください。

⑩ 次に各自が作成したSPにアクセスします。今回はシングルサインオンとして認証済みですが、アクセスレベルが動作確認用SPよりも高いため、証明書認証が要求されます。

⑪ 認証後、正しく属性受信の確認ページが表示される事を確認してください。