2024年11月8日

# Shibbolized Tiqrを用いた認証方式の導入例

国立情報学研究所

教育研修事業

# 1．はじめに

このドキュメントではShibboleth IdPバージョン5の認証において、スマートフォンで動作するオープンソースの認証ソリューションであるtiqrアプリによる認証を必要とするようにIdPの構成を変更する手順を説明しています。

動作確認にはスマートフォン（iOS端末またはAndroid端末）で動作するtiqrアプリが必要です。

# 2．前提条件

このドキュメントで説明する検証環境はすでにShibboleth IdP と SP が構成され、SAML SSO の準備ができていることを前提としています。IdP はSP の認証要求ごとにユーザーを認証し、ユーザーの属性を適切に SP に返すことができる必要があります。

Shibbolized Tiqr では暗黙的に必要なため次の属性を SP に提供する必要があります。

* ePPN (教育者プリンシパル名)
* 表示名

また検証環境はGakuNin が提供する次のドキュメントに基づいてインストールしていることを前提としています。異なる方法でインストールした場合は、システムの要件に応じて以降の手順を変更してください。

* https://meatwiki.nii.ac.jp/confluence/pages/viewpage.action?pageId=157188958（日本語）

# 3. 検証環境

検証環境で使用しているソフトウェアのバージョンは次のとおりです。

* Rocky Linux 9.4
* Apache httpd 2.4
* PHP 5.6.4
  + 古いバージョンですがtiqrzendの動作に必要です。
* OpenJDK 17.0.12
* Jetty 11.0.20
* Shibboleth IdP 5.1.2
* Shibboleth SP 3.4.1

# 4．システム構成

検証環境のシステム構成要素は次のとおりです。

* Shibboleth IdP［FQDN: ex-idp-test??.gakunin.nii.ac.jp］
  + tiqrzendにパスワード認証フローを提供します。
  + Shibboleth SPにtiqrアプリによる認証フローを提供します。
* Shibboleth SP［FQDN: ex-sp-test??.gakunin.nii.ac.jp］
  + tiqrアプリによる認証を表示要件とするWebサービスです。検証環境ではPHPの環境変数一覧を表示するページです。
* tiqrzend［FQDN: ex-idp-test??.gakunin.nii.ac.jp］
  + tiqrアプリを用いた認証機能をShibboleth IdPの拡張認証フローを経由して提供します。tiqrzendの登録フローではShibboleth IdPのパスワード認証の成否により登録の可否を決定しているため、tiqrzendサーバ自体も１つのSPとして構成します。
  + 検証環境ではtiqrzendをShibboleth IdPと同一のサーバで動作させています。

# 5．準備

* Shibboleth IdPにShibboleth SPをインストール

tiqrzendを動作させるサーバ（ここではShibboleth IdPが動作するサーバ）に、学認技術ガイドに従ってShibboleth SPをインストールして設定してください。

* https://meatwiki.nii.ac.jp/confluence/pages/viewpage.action?pageId=12158264

例えば検証環境ではtiqrzendをShibboleth IdPに同居させていますので、同じサーバ

上で次の項目を設定しています。

［SPとしての設定例］

* 暗号化用・署名用のキーペアを作成します。

|  |
| --- |
| # cd /etc/shibboleth  # mkdir cert  # export SPURL="ex-idp-test??.gakunin.nii.ac.jp"  # ./keygen.sh -o cert -u shibd -g shibd -h $SPURL -y 10 -e https://$SPURL/shibboleth-sp -n sp-encrypt  # ./keygen.sh -o cert -u shibd -g shibd -h $SPURL -y 10 -e https://$SPURL/shibboleth-sp -n sp-signing |

* メタデータを作成します。

|  |
| --- |
| # mkdir metadata  # ./metagen.sh -c cert/sp-encrypt-cert.pem -c cert/sp-signing-cert.pem -h $SPURL -e "https://$SPURL/shibboleth-sp" > metadata/sp-metadata.xml |

* /etc/shibboleth/shibboleth2.xmlを変更します。

|  |  |
| --- | --- |
| 13  34  101  102  104 | <ApplicationDefaults entityID="https://ex-idp-test??.gakunin.nii.ac.jp/shibboleth-sp"  <SSO entityID="https://ex-idp-test??.gakunin.nii.ac.jp/idp/shibboleth"  <!-- Simple file-based resolvers for separate signing/encryption keys. -->  <CredentialResolver type="File" use="signing" key="cert/sp-signing-key.pem" certificate="cert/sp-signing-cert.pem"/>  <CredentialResolver type="File" use="encryption"  key="cert/sp-encrypt-key.pem" certificate="cert/sp-encrypt-cert.pem"/> |

* IdPのメタデータを設定します。（前項に続いて/etc/shibboleth/shibboleth2.xmlを変更します。下の例はテストフェデレーション向けです）

|  |
| --- |
| <!-- Example of remotely supplied batch of signed metadata. -->  <MetadataProvider type="XML" validate="true"  url="https://metadata.gakunin.nii.ac.jp/gakunin-test-metadata.xml"  backingFilePath="federation-metadata.xml" maxRefreshDelay="7200">  <MetadataFilter type="RequireValidUntil" maxValidityInterval="1296000"/>  <MetadataFilter type="Signature" certificate="/opt/shibboleth-idp/credentials/gakunin-test-signer-2020.cer" verifyBackup="false"/>  <DiscoveryFilter type="Exclude" matcher="EntityAttributes" trimTags="true"  attributeName="http://macedir.org/entity-category"  attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"  attributeValue="http://refeds.org/category/hide-from-discovery" />  <TransportOption provider="CURL" option="64">1</TransportOption>  <TransportOption provider="CURL" option="81">2</TransportOption>  <TransportOption provider="CURL" option="10065">/etc/pki/tls/certs/ca-bundle.crt</TransportOption>  </MetadataProvider> |

* /etc/shibboleth/attribute-map.xmlを学認技術ガイドに掲載されている次のテンプレートに置き換えます。
  + https://meatwiki.nii.ac.jp/confluence/download/attachments/12158175/attribute-map-template.xml?version=7&modificationDate=1563426445000&api=v2
* /etc/shibboleth/attribute-policy.xmlを学認技術ガイドに掲載されている次のテンプレートに置き換えます。
  + https://meatwiki.nii.ac.jp/confluence/download/attachments/12158175/attribute-policy-template.xml?version=5&modificationDate=1692924854000&api=v2

［IdPとしての設定例］

* /etc/shibboleth/metadata/sp-metadata.xmlを/opt/shibboleth-idp/metadataにコピーします。
* /opt/shibboleth-idp/conf/metadata-providers.xmlを変更します。

|  |  |
| --- | --- |
| 73  74  75 | <!-- -->  <MetadataProvider id="LocalMetadata" xsi:type="FilesystemMetadataProvider" metadataFile="%{idp.home}/metadata/sp-metadata.xml"/>  <!-- --> |

* /opt/shibboleth-idp/conf/attribute-filter.xmlを変更します。

|  |
| --- |
| <AttributeFilterPolicyGroup id="ShibbolethFilterPolicy"  xmlns="urn:mace:shibboleth:2.0:afp"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="urn:mace:shibboleth:2.0:afp http://shibboleth.net/schema/idp/shibboleth-afp.xsd">  …  <AttributeFilterPolicy id="PolicyForIDPasSP">  <PolicyRequirementRule xsi:type="Requester" value="https://ex-idp-test??.gakunin.nii.ac.jp/shibboleth-sp" />  <AttributeRule attributeID="transientId" permitAny="true" />  <AttributeRule attributeID="mail" permitAny="true" />  <AttributeRule attributeID="sn" permitAny="true" />  <AttributeRule attributeID="givenName" permitAny="true" />  <AttributeRule attributeID="displayName" permitAny="true" />  <AttributeRule attributeID="eduPersonTargetedID" permitAny="true" />  <AttributeRule attributeID="eduPersonScopedAffiliation" permitAny="true" />  <AttributeRule attributeID="eduPersonPrincipalName" permitAny="true" />  </AttributeFilterPolicy>  …  </AttributeFilterPolicyGroup> |

* Shibbolized Tiqrパッケージの展開

別途配布のShibbolized Tiqrパッケージ（tiqrshib\_202408.tar.gz）をIdPサーバ上の適当な作業ディレクトリに展開しておきます。

|  |
| --- |
| # mkdir /root/tmp  # cd /root/tmp # cp /root/PKG/tiqrshib\_202408.tar.gz /root/tmp/  # md5sum tiqrshib\_202408.tar.gz  3c584c4f1c5b0ddc680fee9783934c9d tiqrshib\_202408.tar.gz  # tar zxf tiqrshib\_202408.tar.gz  # tree --charset=ascii -L 2 /root/tmp/tiqrshib\_202408  /root/tmp/tiqrshib\_202408  |-- README.md  |-- idp  | |-- build.sh  | |-- pom.xml  | |-- src  | `-- target  `-- tiqrzend  |-- application  |-- library  |-- public  |-- resources  `-- tests |

※treeコマンドがない時は、dnf install tree でインストールできます。

# 6．tiqrzendの構築

検証環境ではtiqrzendをShibboleth IdPが動作するサーバに構築します。

1. tiqrzendの動作に必要なパッケージをインストールします。

Shibbolized Tiqrに含まれるtiqrzendはPHP 5.6で動作しますが、Shibboleth IdPが動作するRocky Linux 9では標準でPHP 5.6をインストールできません。  
このためRavenリポジトリおよびRemiリポジトリからインストールします。ただしリポジトリの性格上、無保証、自己責任で行ってください。

|  |
| --- |
| # dnf install <https://pkgs.dyn.su/el9/base/x86_64/raven-release.el9.noarch.rpm>  # dnf install <https://rpms.remirepo.net/enterprise/remi-release-9.rpm>  # dnf install gd-devel  # dnf install --enablerepo=raven-modular php56 php56-php-fpm php56-php-gd php56-php-pdo php56-php-mcrypt sqlite |

ここではバックエンド DB にsqliteを使用していますが、 MySQLの使用も可能です。

1. tiqrzendをインストールします。

* 5.で展開した/root/tmp/tiqrshib\_202408/tiqrzendディレクトリを/optディレクトリにコピーします。
* phpqrcode 1.1.4 をダウンロードして、/opt/tiqrzend/library/phpqrcode/ に解凍します。（unzipなどの解凍ツールがなければdnfコマンドでインストールしてください）
  + https://sourceforge.net/projects/phpqrcode/files/releases/phpqrcode-2010100721\_1.1.4.zip/download
* /opt/tiqrzend/application/logs を httpdプロセス実行ユーザ（apacheなど） から書き込み可能にします。

上記の手順を実行した後、tree コマンドで結果を確認できます。

|  |
| --- |
| # tree --charset=ascii -L 2 /opt/tiqrzend/  /opt/tiqrzend/  |-- application  | |-- Bootstrap.php  | |-- configs  | |-- controllers  | |-- layouts  | |-- modules  | `-- views  |-- library  | |-- phpqrcode  | |-- tiqr  | |-- tiqrshib  | |-- tiqr-zf  | |-- zend -> ZendFramework-1.12.9-minimal  | `-- ZendFramework-1.12.9-minimal  |-- public  | |-- images  | |-- index.php  | `-- scripts  |-- resources  | `-- Sample.php  `-- tests  |-- application  |-- library  `-- phpunit.xml |

1. tiqrzend を設定します。

* /opt/tiqrzend/application/configs/application.example.ini を /opt/tiqrzend/application/configs/application.ini にコピーして変更します。

|  |
| --- |
| # cp /opt/tiqrzend/application/configs/application.example.ini /opt/tiqrzend/application/configs/application.ini |

|  |  |
| --- | --- |
| 46  50  181 | resources.tiqr.identifier = "ex-idp-test??.gakunin.nii.ac.jp"  resources.tiqr.logoUrl = "https://gakunin.jp/sites/default/files/2019-10/GakuNin\_logo\_mark-S%20%281%29\_0.jpg"  resources.tiqrshib.eppnScope = 'ex-idp-test??.gakunin.nii.ac.jp'  ※/opt/shibboleth-idp/conf/idp.propertiesのidp.scopeに設定されている値と同じ値を設定してください。 |

* tiqrshib の「SecretStorage」用の DB を準備します。

|  |
| --- |
| # mkdir -p /opt/tiqrzend/db/  # sqlite3 /opt/tiqrzend/db/secrets.db  sqlite> CREATE TABLE IF NOT EXISTS tiqrshibsecrets (  ...> `id` integer NOT NULL PRIMARY KEY AUTOINCREMENT,  ...> `uid` text NOT NULL UNIQUE,  ...> `secret` text DEFAULT NULL,  ...> `isActive` integer DEFAULT NULL,  ...> `loginattempts` integer DEFAULT NULL,  ...> `isBlocked` integer DEFAULT NULL);  sqlite> .q  # chown -R apache: /opt/tiqrzend/db  # chmod -R go-rwx /opt/tiqrzend/db |

1. httpdの設定

/etc/httpd/conf.d/ssl.conf (または任意の httpd 設定ファイル) を変更して、httpd に次の設定が含まれるようにします。

|  |
| --- |
| Alias /tiqr/ "/opt/tiqrzend/public/"  <Location /tiqr>  RewriteEngine On  RewriteBase /tiqr  RewriteCond %{REQUEST\_FILENAME} -s [OR]  RewriteCond %{REQUEST\_FILENAME} -l [OR]  RewriteCond %{REQUEST\_FILENAME} -d  RewriteRule ^.\*$ - [NC,L]  RewriteRule ^.\*$ /tiqr/index.php [NC,L]  require all granted  </Location>  <Location /tiqr/shib/enroll/process>  AuthType shibboleth  ShibRequestSetting requireSession 1  ShibRequestSetting authnContextClassRef urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport  require valid-user  </Location> |

最後にhttpdを再起動してください。

# 7．Shibboleth IdPの構成

IdPの認証フローを拡張します。

1. /opt/shibboleth-idp/edit-webapp/WEB-INF/web.xmlファイルを変更します。

|  |
| --- |
| …  <session-config>  <session-timeout>15</session-timeout>  <cookie-config>  <!-- Comment name and path to revert \_\_Host- protection. -->  <name>JSESSIONID</name>  <path>/</path>  <secure>true</secure>  <http-only>true</http-only>  </cookie-config>  <tracking-mode>COOKIE</tracking-mode>  </session-config>  …  <!-- Servlets and servlet mappings -->  <servlet>  <servlet-name>TiqrShibAuthnHandler</servlet-name>  <servlet-class>jp.gakunin.tiqrshib.TiqrShibAuthnServlet</servlet-class>  <load-on-startup>4</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>TiqrShibAuthnHandler</servlet-name>  <url-pattern>/Authn/TiqrShib</url-pattern>  </servlet-mapping>  <servlet>  <servlet-name>default</servlet-name>  <servlet-class>org.eclipse.jetty.servlet.DefaultServlet</servlet-class>  <init-param>  <param-name>dirAllowed</param-name>  <param-value>false</param-value>  </init-param>  <load-on-startup>0</load-on-startup>  </servlet> |

1. /opt/shibboleth-idp/edit-webapp/WEB-INF/libディレクトリを作成して、tiqrshib\_202408に含まれるjarファイル2つをコピーしてください。

|  |
| --- |
| # mkdir /opt/shibboleth-idp/edit-webapp/WEB-INF/lib  # cp /root/tmp/tiqrshib\_202408/idp/target/java-support-8.4.2.jar /opt/shibboleth-idp/edit-webapp/WEB-INF/lib  # cp /root/tmp/tiqrshib\_202408/idp/target/tiqrshibauthn-5.0.jar /opt/shibboleth-idp/edit-webapp/WEB-INF/lib |

1. /opt/shibboleth-idp/bin/build.shを実行し、idp.warファイルをリビルドします。

|  |
| --- |
| # /opt/shibboleth-idp/bin/build.sh  INFO - net.shibboleth.idp.installer.impl.IdPBuildArguments@6833ce2c  INFO - Rebuilding /opt/shibboleth-idp/bin/../war/idp.war, Version 5.1.2  INFO - Initial populate from /opt/shibboleth-idp/bin/../dist/webapp to /opt/shibboleth-idp/bin/../webpapp.tmp  INFO - Overlay from /opt/shibboleth-idp/bin/../edit-webapp to /opt/shibboleth-idp/bin/../webpapp.tmp  INFO - Creating war file /opt/shibboleth-idp/bin/../war/idp.war |

1. /opt/shibboleth-idp/conf/authn/authn.propertiesを変更します。

|  |
| --- |
| idp.authn.flows = Password|tiqrshib  idp.authn.External.externalAuthnPath = contextRelative:/Authn/Tiqrshib |

1. /opt/shibboleth-idp/conf/authn/tiqrshib-authn-config.xmlを次の内容で作成します。

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans" xmlns:context="http://www.springframework.org/schema/context" xmlns:util="http://www.springframework.org/schema/util" xmlns:p="http://www.springframework.org/schema/p" xmlns:c="http://www.springframework.org/schema/c" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util.xsd" default-init-method="initialize" default-destroy-method="destroy">  <bean id="shibboleth.authn.External.externalAuthnPath" class="java.lang.String" c:\_0="contextRelative:Authn/TiqrShib"/>  <util:constant id="shibboleth.authn.External.addDefaultPrincipals" static-field="java.lang.Boolean.TRUE"/>  <util:map id="shibboleth.authn.External.ClassifiedMessageMap">  <entry key="UnknownUsername">  <list>  <value>UnknownUsername</value>  </list>  </entry>  <entry key="InvalidPassword">  <list>  <value>InvalidPassword</value>  </list>  </entry>  <entry key="ExpiredPassword">  <list>  <value>ExpiredPassword</value>  </list>  </entry>  <entry key="ExpiringPassword">  <list>  <value>ExpiringPassword</value>  </list>  </entry>  </util:map>  </beans> |

1. /opt/shibboleth-idp/conf/global.xmlの<beans>～</beans>の間に次の設定を追加します。

|  |
| --- |
| <beans ・・・>  <util:list id="shibboleth.AvailableAuthenticationFlows">  <bean id="authn/tiqrshib" parent="shibboleth.AuthenticationFlow" p:order="%{idp.authn.External.order:1000}" p:nonBrowserSupported="false">  <property name="supportedPrincipals">  <list>  <bean parent="shibboleth.SAML2AuthnContextClassRef" c:classRef="urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorUnregistered"/>  </list>  </property>  </bean>  </util:list>  </beans> |

1. /opt/shibboleth-idp/conf/idp.propertiesを変更します。

|  |
| --- |
| idp.csrf.enabled=false |

1. /opt/shibboleth-idp/flows/authn/tiqrshib ディレクトリを作成し、次の内容でtiqrshib-beans.xmlとtiqrshib-flow.xmlを作成します。

|  |
| --- |
| # mkdir /opt/shibboleth-idp/flows/authn/tiqrshib  # chmod 755 /opt/shibboleth-idp/flows/authn/tiqrshib |

|  |
| --- |
| tiqrshib-beans.xml |
| <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:context="http://www.springframework.org/schema/context"  xmlns:util="http://www.springframework.org/schema/util"  xmlns:p="http://www.springframework.org/schema/p"  xmlns:c="http://www.springframework.org/schema/c"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd  http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util.xsd"  default-init-method="initialize"  default-destroy-method="destroy">  <bean class="org.springframework.context.support.PropertySourcesPlaceholderConfigurer"  p:placeholderPrefix="%{" p:placeholderSuffix="}" />  <bean class="net.shibboleth.ext.spring.config.IdentifiableBeanPostProcessor" />  <bean class="net.shibboleth.idp.profile.impl.ProfileActionBeanPostProcessor" />  <bean id="shibboleth.authn.External.externalAuthnPathStrategy"  class="com.google.common.base.Functions" factory-method="constant"  c:\_0-ref="shibboleth.authn.External.externalAuthnPath" />  <import resource="../../../conf/authn/tiqrshib-authn-config.xml" />  <bean id="ValidateExternalAuthentication"  class="net.shibboleth.idp.authn.impl.ValidateExternalAuthentication" scope="prototype"  p:matchExpression="#{getObject('shibboleth.authn.External.matchExpression')}"  p:addDefaultPrincipals="#{getObject('shibboleth.authn.External.addDefaultPrincipals') ?: true}"  p:classifiedMessages-ref="shibboleth.authn.External.ClassifiedMessageMap"  p:resultCachingPredicate="#{getObject('shibboleth.authn.External.resultCachingPredicate')}" />  <bean id="PopulateSubjectCanonicalizationContext"  class="net.shibboleth.idp.authn.impl.PopulateSubjectCanonicalizationContext" scope="prototype"  p:availableFlows-ref="shibboleth.PostLoginSubjectCanonicalizationFlows" />  </beans> |

|  |
| --- |
| tiqrshib-flow.xml |
| <flow xmlns="http://www.springframework.org/schema/webflow"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://www.springframework.org/schema/webflow http://www.springframework.org/schema/webflow/spring-webflow.xsd"  parent="authn.abstract">  <view-state id="ExternalTransfer" view="externalRedirect:#{T(net.shibboleth.idp.authn.ExternalAuthentication).getExternalRedirect(flowRequestContext.getActiveFlow().getApplicationContext().getBean('shibboleth.authn.External.externalAuthnPathStrategy').apply(opensamlProfileRequestContext), flowExecutionContext.getKey().toString())}">  <on-render>  <evaluate expression="opensamlProfileRequestContext.getSubcontext(T(net.shibboleth.idp.authn.context.AuthenticationContext)).addSubcontext(new net.shibboleth.idp.authn.context.ExternalAuthenticationContext(new net.shibboleth.idp.authn.impl.ExternalAuthenticationImpl(calledAsExtendedFlow?:false)), true).setFlowExecutionUrl(flowExecutionUrl + '&amp;\_eventId\_proceed=1')" />  </on-render>  <transition to="ValidateExternalAuthentication" />  </view-state>  <action-state id="ValidateExternalAuthentication">  <evaluate expression="ValidateExternalAuthentication" />  <evaluate expression="PopulateSubjectCanonicalizationContext" />  <evaluate expression="'proceed'" />  <transition on="proceed" to="CallSubjectCanonicalization" />  </action-state>  <subflow-state id="CallSubjectCanonicalization" subflow="c14n">  <input name="calledAsSubflow" value="true" />  <transition on="proceed" to="proceed" />  <transition on="SubjectCanonicalizationError" to="ReselectFlow" />  </subflow-state>  <bean-import resource="tiqrshib-beans.xml" />  </flow> |

1. Jettyを再起動します。

|  |
| --- |
| # systemctl stop jetty  # systemctl start jetty |

# 8．動作確認

### 登録フロー

1. PCのブラウザでtiqrzendの登録ページを表示します。

検証環境ではtiqrzendはIdPが動作するサーバ（ex-idp-test??.gakunin.nii.ac.jp）と同居させているため、登録フローのURLは次のようになります。

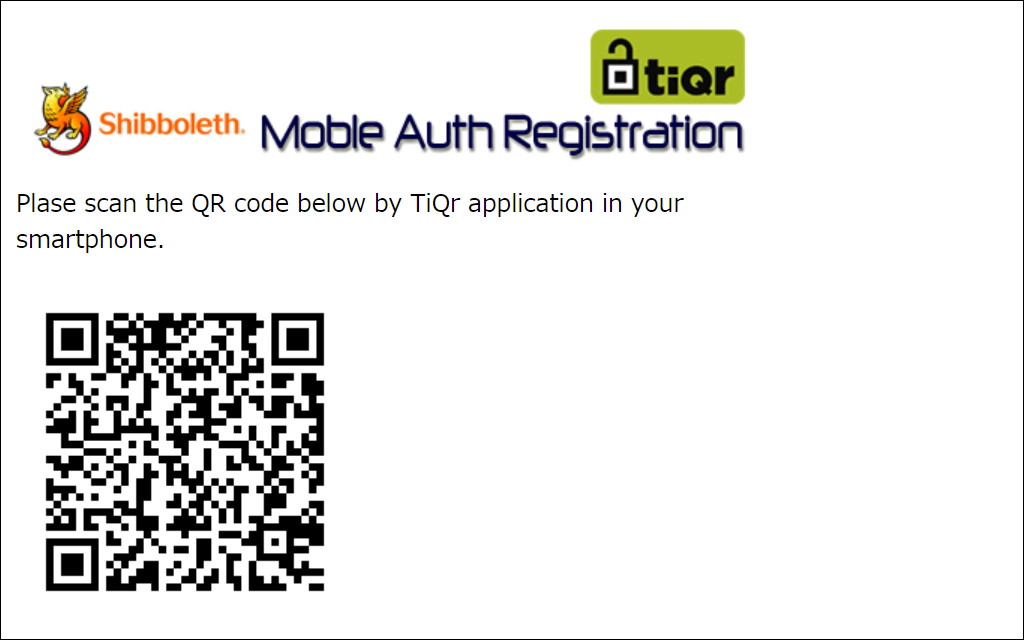
|  |
| --- |
| https://ex-idp-test??.gakunin.nii.ac.jp/tiqr/shib/enroll |



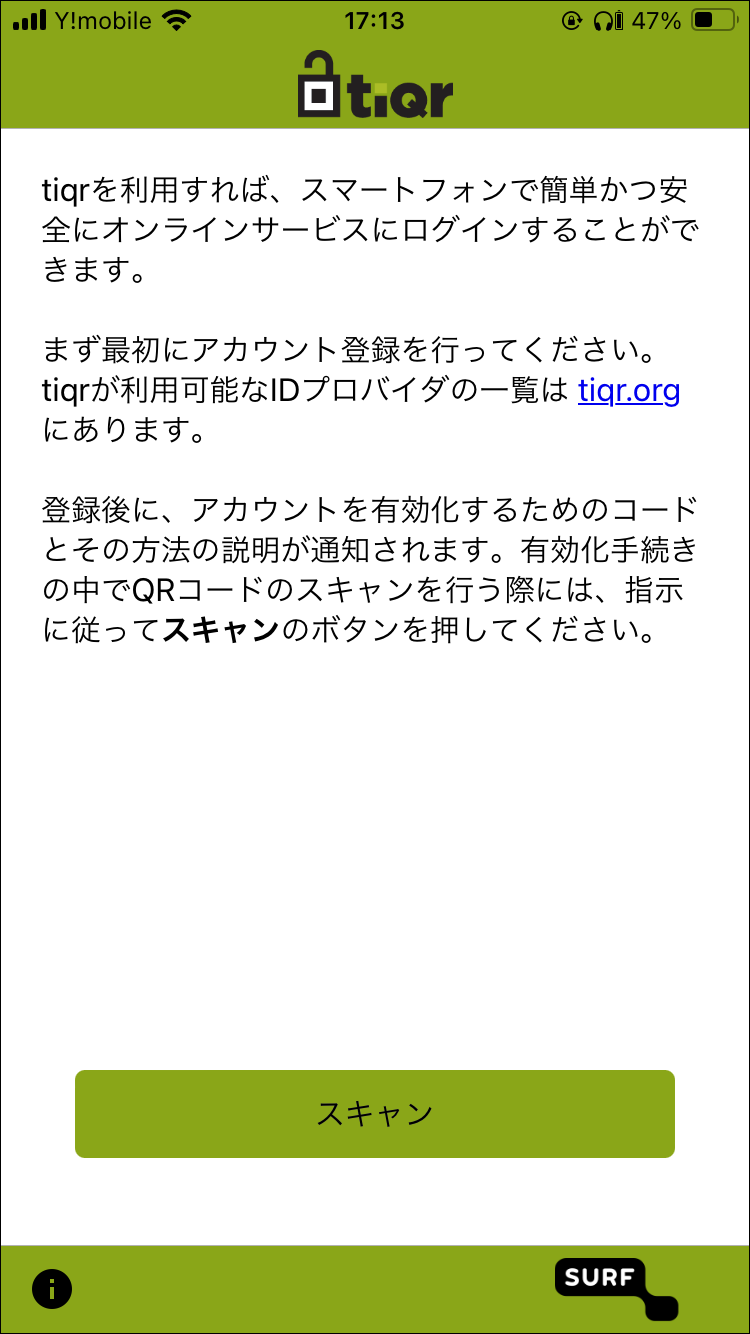
1. ［Register］ボタンを押下してIdPのパスワード認証ページを開き、［ユーザ名］と［パスワード］を入力して［Login］ボタンを押下します。



1. IdPでパスワード認証を通過すると登録用のQRコードが表示されるので、スマートフォンのtiqrアプリを起動してQRコードを読み取ります。



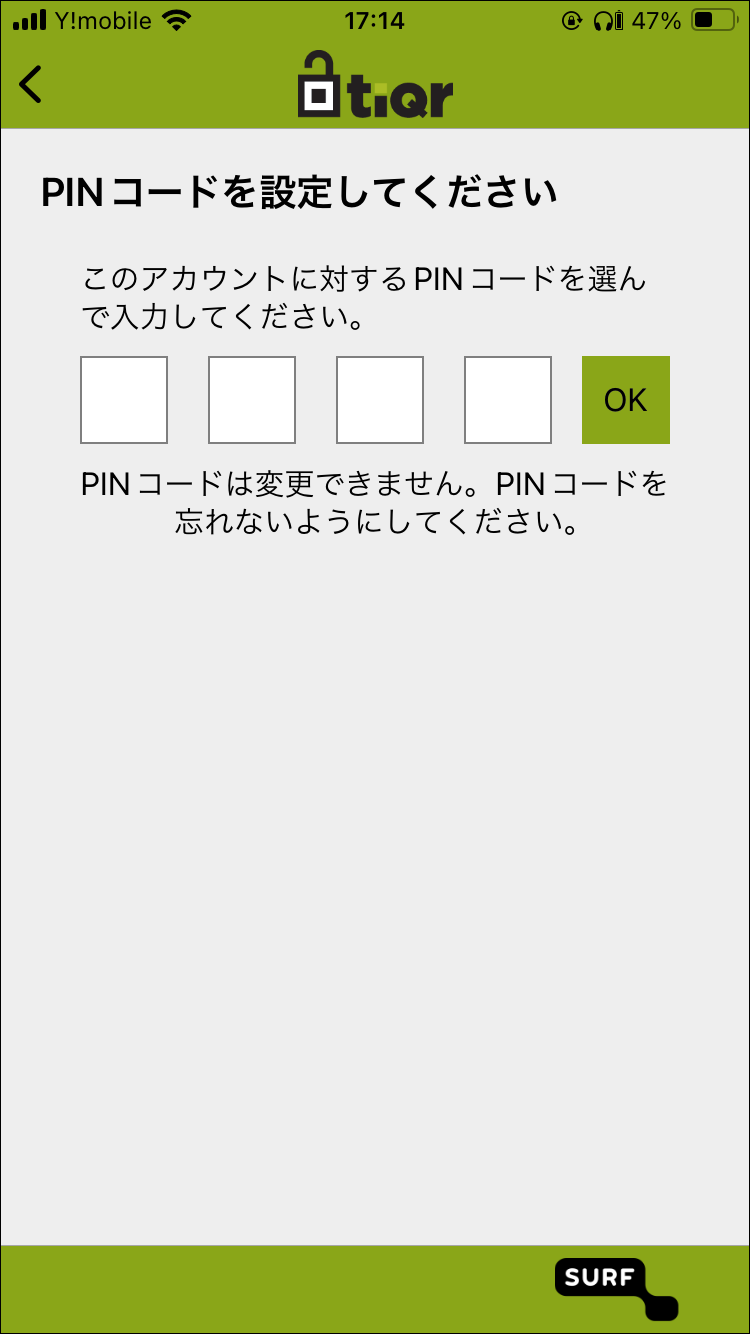
tiqrアプリを開いて［スキャン］ボタンを押下します。



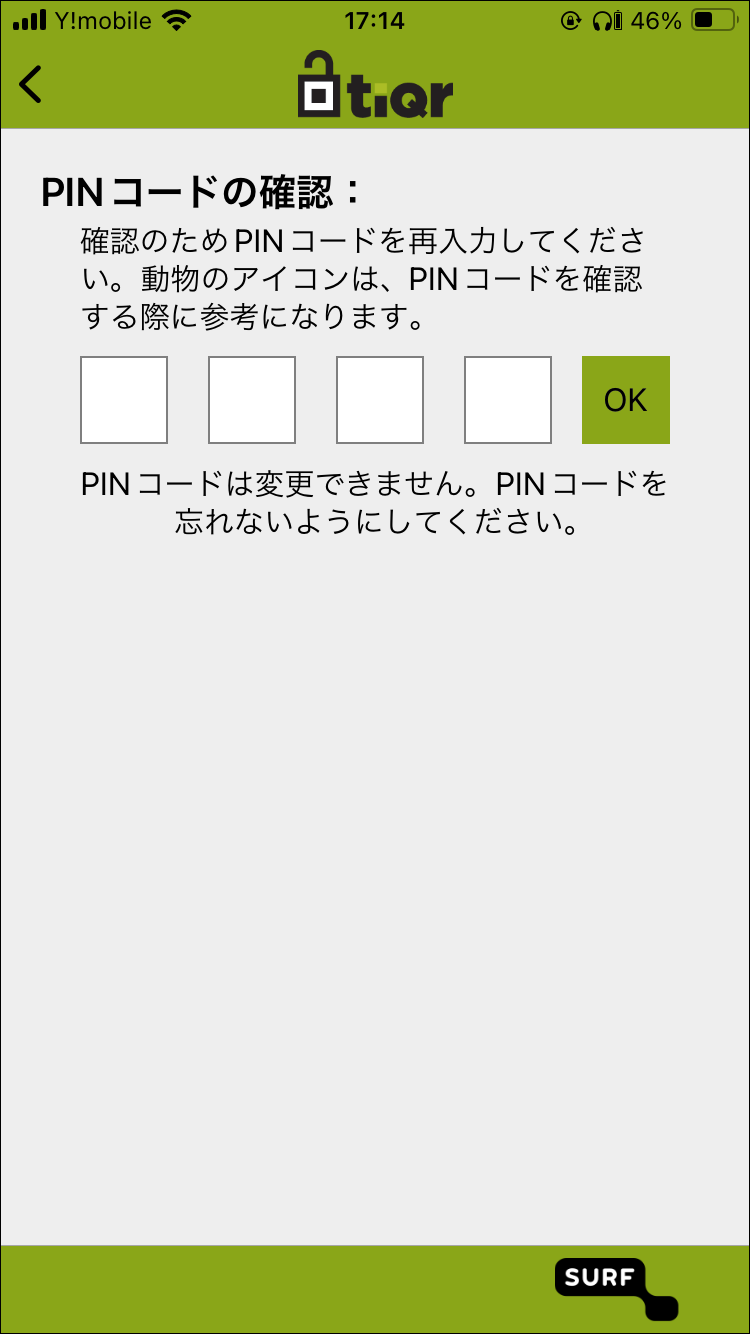
QRコードをスキャンすると、確認画面が表示されるので［OK］ボタンを押下します。



続いて、PINコードの設定画面が表示されます。4桁のPINを設定し、［OK］ボタンを押下します。



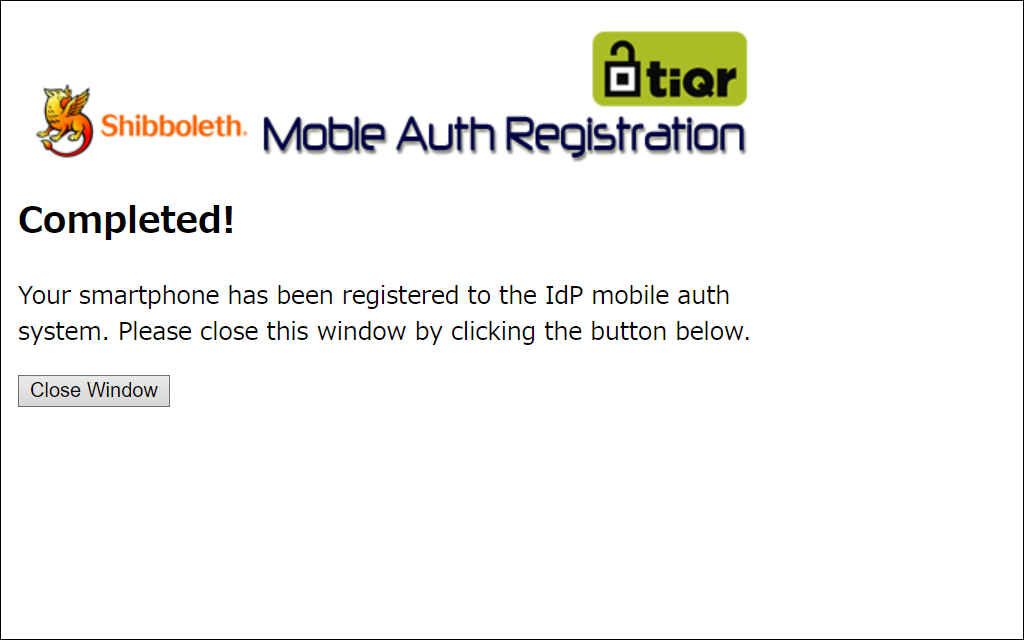
確認のためPINコードの再入力が求められますので、入力して［OK］ボタンを押下します。



tiqrzendにtiqrアプリが登録されます。

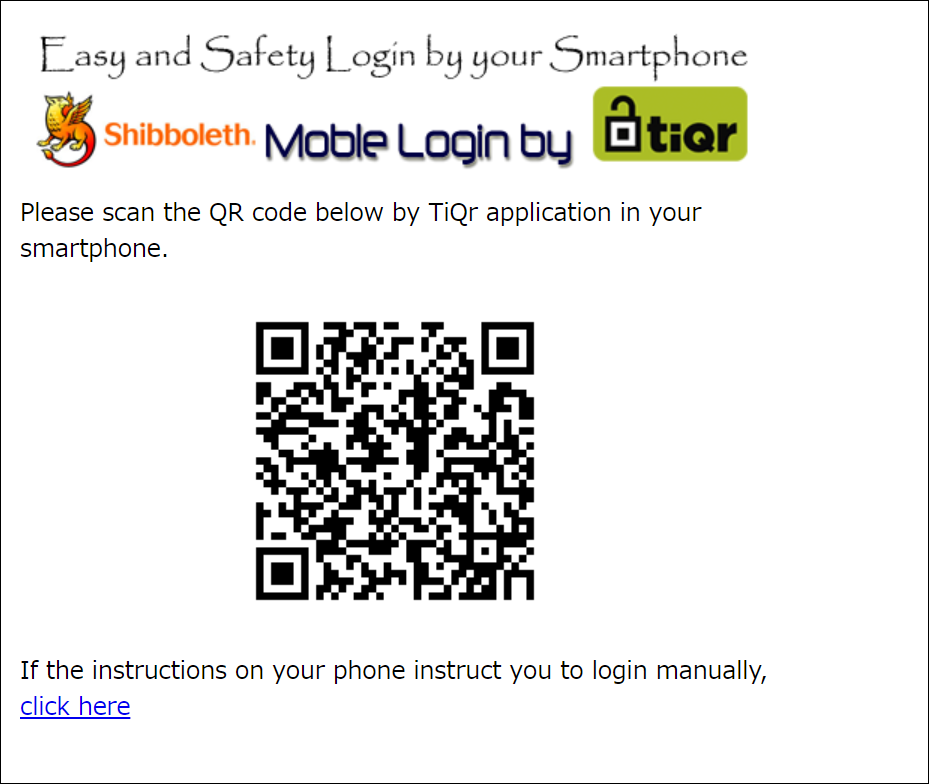


1. PCのブラウザ画面に、登録フロー完了メッセージが表示されます。



### ログインフロー

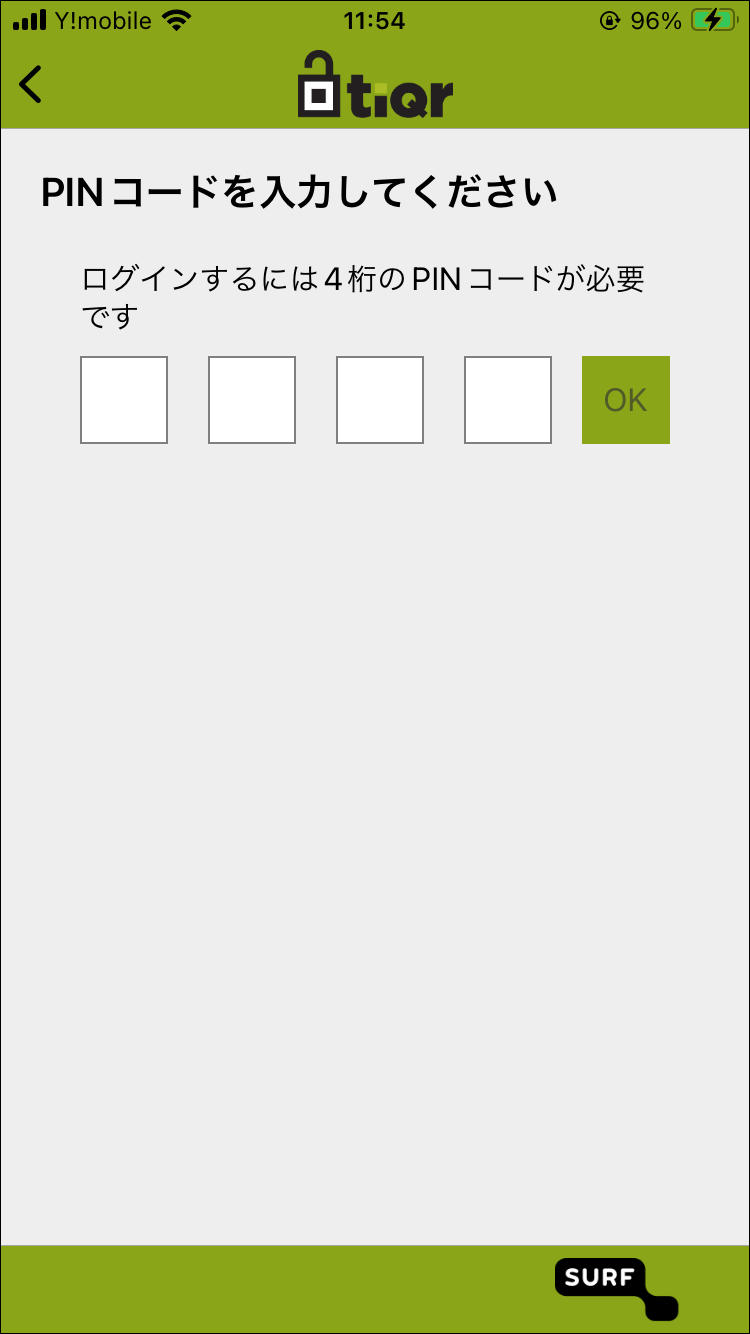
1. PCのブラウザで接続確認用SPにアクセスします。  
   テストフェデレーションの場合: <https://test-sp1.gakunin.nii.ac.jp/>
2. ログインボタンをクリックします。
3. 所属機関の選択画面が表示されるので、各自が使用するIdPを選択して、IdPのログイン画面を表示します。



1. 表示されたQRコードをスマートフォンのtiqrアプリで読み取ります。  
     
   ログインの確認画面が表示されたら、［Allow］ボタンを押下して先へ進みます。



PINコードの入力が求められます。登録フローで設定したPINコードを入力して［OK］ボタンを押下します。



PINコードの照合が成功すると、ログイン成功のメッセージが表示されます。



1. PCのブラウザの画面は認証完了後のSPの画面が表示されます。  
   