# Installing Shibbolized Tiqr

⚠ If you are using TiqrShib version 2.0, we strongly recommend you update it to 2.1 or later.

Following document instructs how to install Shibbolized Tiqr (or "tiqrshib") in the IdP environment. Please feel free to contact us (tiqr at meatmail.jp) if you have any difficulty during installation.



## About Enviroment

This instruction is for Shibbolized Tiqr 2.2. Shibbolized Tiqr is tested under the following environment.

- CentOS 7.5
- httpd 2.4
- PHP 5.4
- Oracle JDK 1.8
- Apache Maven 3.2
- Tomcat 9.0
- Shibboleth IdP 3.4.0
- Shibboleth SP 3.0

This document assumes Shibboleth IdP and SP are configured and ready for SAML SSO; IdP should be able to authenticate users per SP's authn request, and supply users' attributes back to SP appropriately. Some attributes must be supplied to SP because Shibbolized Tiqr implicitly requires them:

- ePPN (eduPersonPrincipalName)
- displayName

This document assumes they are installed based on the following document provided by GakuNin. If you installed them in a different way, modify the procedure below as your system requires.

https://meatwiki.nii.ac.jp/confluence/pages/viewpage.action?pageId=20021624 (in Japanese)

## Installation

### Install Dependent Yum Packages

To install depending packages, run the following command:

```
$ sudo yum -y install httpd php mod_ssl php-gd php-pdo php-mcrypt sqlite
```

Here, sqlite is installed as our sample configuration requires it. If you want to use MySQL for the backend DB, consider installing "mysql-connector-odbc" and "php-mysql" instead.

### Checkout Source Code for Shibbolized Tiqr

Checkout the source code from the repository.

```
$ svn co https://forge.gakunin.nii.ac.jp/anonsvn/tiqrshib/trunk tiqrshib
```

It should contain two projects, "tiqrzend" and the Shibboleth IdP extension.

## Deploy "tiqrzend" server

Here we setup "tiqrzend" first.

- Extract all the content in "tiqrzend" package under /opt/tiqrzend/.
- Download phpqrcode 1.1.4 and unarchive under /opt/tiqrzend/library/phpqrcode/: http://phpqrcode.sourceforge.net/
- Make /opt/tiqrzend/application/logs writable from your httpd.

After the procedure above, you can check the result with tree command:

```
$ 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
20 directories, 4 files
```

### Configure tiqrzend

Copy /opt/tiqrzend/application/configs/application.example.ini to /opt/tiqrzend/application/configs/application.ini.

```
$ cp /opt/tiqrzend/application/configs/application.example.ini /opt/tiqrzend/application/configs/application.ini
```

The sample configuration file should contain detailed instructions for each setting. It is recommended to read it through and modify the setting as your organization requires.

At the very least, you probably need to configure two settings show below:

- resources.tiqrshib.eppnScope must be exactly same as idp.scope setting in Shibboleth IdP's idp.properties (/opt/shibboleth-idp/conf/idp.properties)
- resources.tiqr.identifier should contain your organization identifier

Prepare DB for tiqrshib's "SecretStorage". For evaluation purpose, run the following commands:

```
$ 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> (exit)
$ sudo chown -R apache: /opt/tiqrzend/db
$ sudo chmod -R go-rwx /opt/tiqrzend/db
```

## Configure httpd

Modify /etc/httpd/conf.d/ssl.conf (or whatever httpd config file you prefer) so httpd contain the following settings:

```
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
  require valid-user
</Location>
```

## Confirm "tiqrzend" is Working

With the modifications above, tiqr authentication flow should be ready for evaluation.

Now try the following "enrollment" and "login" flow.

- Enrollment flow
    - Visit https://(your host name)/tiqr/shib/enroll
    - Check if a QR code is shown to the screen
    - Scan the QR code using Tiqr mobile application, and enter a 4-digit PIN code
    - You should be able to register the service via the mobile app.
- Login flow
    - After the enrollment, visit https://(your host name)/tiqr/shib/login
    - You will see another QR code
    - Scan the QR code with the Tiqr mobile app
    - You are asked to enter the 4-digit PIN code that you entered in the enrollment flow
    - If you enter the correct PIN, Tiqr mobile app ask you if you are ok with the login.
    - After you approve the login, the message "Login is successful" will be shown to the mobile app.
    - You will see HTTP 404 or similar error messages that happens because IdP is not ready yet.
        - At this step you can at least check enrollment and login in tiqr side is successful.

Note that web interface will show some error message at the last step. At this point it is expected. The tiqrzend's login interface is intended for IdP integration, and without IdP, it cannot correctly redirect back to the IdP.

What you should check here is if the redirect happens at this point. If the redirect does not happen and some other problem happens, you will need to trouble-shoot it here.

## Checking "tiqrzend" Log

There are multiple logs to be checked when there are some errors.

- httpd's logs in /var/log/httpd
- PHP's logs in syslog or whatever you configured at /etc/php.ini
- tiqrzend's logs /opt/tiqrzend/application/logs/

# Deploy the Shibboleth IdP extension

## Build the Extension

Building the IdP extension requires Apache Maven. You can build the extension outside where IdP resides, but you need the same Java environment to build the extension there.

Here is how to install Apache Maven 3.5.0 in /opt/ directory.

```
$ export MVN_VERSION=3.5.0
$ wget http://ftp.riken.jp/net/apache/maven/maven-3/${MVN_VERSION}/binaries/apache-maven-${MVN_VERSION}-bin.tar.gz
$ tar xf apache-maven-${MVN_VERSION}-bin.tar.gz -C /opt/
$ export PATH=/opt/apache-maven-${MVN_VERSION}/bin:$PATH
$ mvn --version
(Check Maven is installed)
```

Under the "idp/" directory in this project, run the following commands:

```
$ ls
pom.xml  src
$ mvn -DskipTests=false clean package
... (a lot of logs) ...
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1.683 s
[INFO] Finished at: 2017-09-05T16:41:39+09:00
[INFO] Final Memory: 24M/361M
[INFO] ------------------------------------------------------------------------
$ ls
pom.xml  src  target
$ ls target/
classes  maven-archiver  tiqrshibauthn-2.2.jar
```

Here, `tiqrshibauthn-2.x.jar` in target/ is what we want.

If you are outside the host where IdP resides, copy the jar file to the server.

## Install the Extension into Shibboleth IdP

From this step, we assume you logged in the host where IdP resides as root.

Before going forward, we recommend to stop Tomcat.

```
# systemctl stop tomcat
```

Place tiqrshibauthn-2.x.jar in `/opt/shibboleth-idp/edit-webapp/WEB-INF/lib/` directory.

```
# cp -i .../tiqrshibauthn-2.?.jar /opt/shibboleth-idp/edit-webapp/WEB-INF/lib/
```

Add the following XML content in `/opt/shibboleth-idp/edit-webapp/WEB-INF/web.xml`. If you don't have the web.xml file, copy from `/opt/shibboleth-idp/webapp/WEB-INF/web.xml` in advance.

```
<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>
```

Rebuild the modified IdP WAR package using bin/build.sh.

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

Depending on your Tomcat configuration, you may need to copy `idp.war` in `$CATALINA_BASE/webapps/` and remove `idp/` directory there.

```
# cp -f /opt/shibboleth-idp/war/idp.war $CATALINA_BASE/webapps/
# rm -rf $CATALINA_BASE/webapps/idp
```

Also depending on your Tomcat version, you may need to remove temporary files.

```
# rm -r $CATALINA_BASE/webapps/idp $CATALINA_BASE/work/Catalina/localhost/idp
```

## Configure IdP for the Extension

Move to `/opt/shibboleth-idp/` and modify relevant configuration files.

```
# cd /opt/shibboleth-idp
```

Create `flows/authn/tiqrshib/` directory and copy some relevant XML files. Then edit copied files so they become ready for tiqrshib authentication.

You can use the following sequence of sed commands if original XML are untouched:

```
# mkdir flows/authn/tiqrshib/
# cp system/flows/authn/external-authn-flow.xml flows/authn/tiqrshib/tiqrshib-flow.xml
# sed -i 's/external-authn-beans¥.xml/tiqrshib-beans.xml/' flows/authn/tiqrshib/tiqrshib-flow.xml
# cp system/flows/authn/external-authn-beans.xml flows/authn/tiqrshib/tiqrshib-beans.xml
# sed -i 's/external-authn-config¥.xml/tiqrshib-authn-config.xml/' flows/authn/tiqrshib/tiqrshib-beans.xml
# cp dist/conf/authn/external-authn-config.xml.dist conf/authn/tiqrshib-authn-config.xml
# sed -i 's/Authn¥/External/Authn¥/TiqrShib/' conf/authn/tiqrshib-authn-config.xml
```

All the procedures above are case-sensitive. Distinguish between "authn/tiqrshib" and "Authn/TiqrShib".

Add the following XML content in `conf/authn/general-authn.xml`.

```
<bean id="authn/tiqrshib" 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:MobileTwoFactorUnregistered" />
        </list>
    </property>
</bean>
```

You probably want to modify `classRef` settings, depending on what authnContext you want to use for the tiqrshib authn.

Modify `conf/idp.properties` so `idp.authn.flows` contain "tiqrshib"

```
idp.authn.flows=Password|tiqrshib
```

Add `sessionCookiePath="/"` in `Context` element in `$CATALINA_BASE/conf/Catalina/localhost/idp.xml` in order to be able to check JSESSIONID in PHP side. E.g.:

```
<Context docBase="/opt/shibboleth-idp/war/idp.war"
         privileged="true"
         antiResourceLocking="false"
         swallowOutput="true"
         sessionCookiePath="/">

...
</Context>
```

Finally, start Tomcat.

```
# systemctl start tomcat
```

## Make Shibboleth SP Rely on Shibbolized Tiqr

Modify SP so that it requests the authnContext you specified in general-authn.xml on certain path.

Here's an example of httpd configuration on a Shibboleth SP server, assuming it is already under a same federation with the IdP.

```
<Location /shibtiqr_protected>
    AuthType shibboleth
    ShibRequestSetting requireSession 1
    ShibRequestSetting authnContextClassRef urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorUnregistered
    Require valid-user
</Location>
```

# Configuration

Now that you confirmed Shibbolized Tiqr works on your server, we recommend reviewing your tiqrshib settings. Check your application.ini and review your settings. If you copied it from application.example.ini, you will see prenty of comments that describe each configuration and some recommendations.

In this section, we'll describe some of most important aspects of possible configurations.

## Encryption Key and IV

Change secrets shared between tiqrzend and the IdP extension.

- On tiqrzend side, modify resources.tiqrshib.encryption.key and resources.tiqrshib.encryption.iv.
- On the extension side, modify AES_KEY and AES_IV in TiqrShibConstants.java in the IdP extension.

Note that the modification requires recreating jar file and IdP package re-build.

## Metadata for Tiqr Server

Tiqr mobile application will show users basic information about the Tiqr server. The information is provided from Tiqr server on enrollment process. You probably want to change it before production use.

Check the following three settings.

- resources.tiqr.identifier
- resources.tiqr.name
- resources.tiqr.logoUrl

Consult to the official document (https://tiqr.org/tiqr-simplesaml-integration-guide) for more information.

## Storage Setting

When SQLite3 is used, all the secret for users will be stored in local storange.

You may want some redundancy or possibly more performance, in which case consider using MySQL instead of SQLite3. There are descriptions in the example config how to achieve it.