TIQR QUICK START

TiQR Quick Start

Web-Service

Start with TiQR Server

1. Introduction

TiQR is an innovative way to authenticate users to web applications. It is based on open standards for secure authentication developed by the Open Authentication Initiative. TiQR's unique user-friendly features include one-click enrollment using QR codes and secure authentication without having to re-type complicated codes by leveraging dynamic QR codes embedded in web pages. TiQR supports the OCRA suite of authentication protocols. The security is based on AES 256-bit encryption and the SHA-family hashing functions. This document is intended to provide a quick start guide to administrators who want to test and implement RCDevs TiQR Authentication Server. The reader should notice that this document is not a guide for installing and using WebADM and its applications. Specific guides are available through the RCDevs' online documentation library at RCDevs Documentation Website.

In this quick start guide, we will cover the following points:

1. How to install and configure your TiQR Authentication server in WebADM.
2. How to create a user and test the TiQR authentication.
3. How to implement TiQR in a PHP login page.

WebADM installations and configuration manuals are not covered by this guide and are documented in the WebADM Installation Guide and WebADM Administrator's Guide available through RCDevs' online documentation.

2. Install TiQR Server

2.1 Install and Configure WebADM

In order to setup RCDevs TiQR Server, you must have a working WebADM server installation. This guide assumes your target system already has a running WebADM server, configured and connected to a compatible LDAP directory. If you do not have the proper environment in place, we recommend that you first download and run one of the RCDevs' pre-installed VMWare appliances. Please go to RCDevs Downloads to get your VMWare appliance.

2.2 Download and Install the TiQR Package

If you installed a VMWare Appliance or the RCDevs webadm-all-in-one package, TiQR server is already installed. Else, you can download the TiQR Server package at RCDevs Downloads.

Copy the package file on your WebADM Linux server with WinSCP or another SSH/SCP client application and unzip it with the command:

```
gunzip tiqr-1.0.x.sh.gz
```

Then run the installer with the commands:

```
chmod 755 tiqr-1.0.x.sh
./tiqr-1.0.x.sh
```

The installer will ask you to confirm the installation or to confirm the upgrade if an older version of TiQR Server is already installed. Just say 'y' and press 'enter'. Once TiQR Server is installed, restart your WebADM server with the command:

```
/opt/webadm/bin/webadm restart
```

Your TiQR server is now installed in the `/opt/webadm/websrvs/tiqr/` directory and you need to configure the TiQR Web service settings in WebADM.

3. Configure TiQR Server

The TiQR configuration requires two things: You first need to configure the Web Service Application in WebADM and you need to set up your mobile communication endpoint.

3.1 TiQR Application Configuration

Log in the WebADM Admin Portal with your Super Administrator account and click the Applications button in the top menu bar. The TiQR Authentication Server appears in the list of installed Web Services but is not registered. Just click the `REGISTER` button to register the TiQR Web Service application in WebADM.

The TiQR application is now registered but is still not configured. The registration created a default configuration for your application. But all the mandatory configurations are not present and you need to adjust some of the default settings.

Click the `CONFIGURE` button to enter the TiQR application configuration.

We will not go in details into all the TiQR configuration as they are explained inline and most of the settings are already configured with default working values. We will start by adjusting the Default Domain setting. Domains are a very important thing in WebADM. They are required by your Web Services (ex. TiQR) to know where to search for users while processing requests. Your WebADM server should have at least one Domain already setup and your testing users must be located in an LDAP tree below the User Search Base setting of this Domain. You can check the Default Domain checkbox and select your existing Domain (here Default).

For the rest, the most important settings to be adjusted are:

- The Service Name: This is your TiQR service name as it will be displayed in the TiQR mobile applications when a user enrolls a TiQR.
- Session Timeout: This is the default TiQR authentication timeout. It can be adjusted in the API requests or on WebADM Web Services Client policies.
- QRCode Size: This is the pixel size for the generated QRCode images. 4 means a QR pixel is displayed as a 4x4 pixels square to the users.
- OCRA Suite: TiQR relies on the OATH OCRA (Challenge-Response) algorithm. The default OCRA Suite means that the QRCodes will contain a 10 characters hexadecimal challenge (QH10), that the TiQR response will be computed with a SHA256 algorithm (HOTP-SHA256-8) and the TiQR responses will be 8 digit passwords. You can adjust the challenge length, response length and algorithm in the OCRA Suite. But we recommend keeping the default values. Note: All OCRA Suites as defined in the OCRA IETF Draft specification are not supported by TiQR. Counters (C) or Timestamps (T) are forbidden and Session (S) is mandatory.
- Mobile Endpoint: This is the Web service Endpoint for mobile phones to server communications. It must be a publicly accessible URL. See in the next section for details.
- Blocking Timer: This is the delay to refuse requests for the same user after a failure. Better use the Max Tries features than the blocking timer with TiQR.
- Max Tries: This is the number of failed authentication attempts authorized before blocking the account. Please consider always using this feature for production environments. A value of 3 is recommended. With this setting, the TiQR application will prompt 3 times for a PIN Code retry before the server blocks the user account. If this setting is not used, the server will ask the user for 3 PIN Code attempts before aborting the authentication session. But the account is not blocked and the user can immediately retry.
- Block Time: With a value of 0, the account is blocked permanently until an administrator unblocks it. You can alternatively choose to block the account for a time period.

> **Note**
>
> Most of the TiQR settings can be adjusted in the SOAP requests (in the settings attribute of the tiqrStart SOAP method) or with WebADM Web Services Client policies.

Once the settings are configured, click the `Apply` button and your TiQR application is now configured. We will come back to the Mobile Endpoint setting in the next section and you will also have to re-enter the TiQR configuration to adjust the Endpoint URL.

The TiQR service is now running and the SOAP API is accessible under the listed Service URLs.

3.2 Mobile Endpoint Setup

The TiQR Server provides two distinct APIs:

1. The SOAP API accessible under the service URLs which will be used by your Web server' integrations. The SOAP API is an internal service used by your integrations (i.e. Your Web applications). It should not be accessible from the Internet.
2. The TiQR Mobile Endpoint which is the HTTP(S) URL to be used by the TiQR user mobile applications to communicate with the TiQR Server. Unlike the SOAP API, the Mobile Endpoint has to be accessible from mobile phones and also from anywhere on the Internet (via 3G / GPRS / Wifi).

The Mobile Endpoint is located under the mobile/ directory below the SOAP Web Service API URL. Yet, for security reasons, we DO NOT recommend opening the WebADM Web Services on the Internet. The WebADM SOAP ports 8080 and 8443 should be accessible from your private network only. Also, a better solution is to create a reverse-proxy URL on one of your public Web servers to let the mobile phones access the TiQR Mobile Endpoint from the Internet without exposing your WebADM server on the Internet. The reverse-proxy URL will forward the TiQR requests to your internal TiQR Server's Mobile Endpoint.

For example, let's say your internal WebADM server is accessible via the hostname mywebadmserver. Then your internal TiQR SOAP API is `http://mywebadmserver:8080/tiqr/` and `https://mywebadmserver:8443/tiqr/`. Your internal Mobile Endpoint URL will also be `http://mywebadmserver:8080/tiqr/mobile/` or `https://mywebadmserver:8443/tiqr/mobile/`. Note that you may not need to use the SSL URL for internal communications from the reverse-proxy.

Now let's say you have a public Web site hosted on an Apache server at `http://mywebsite.com/` and `https://mywebsite.com/`. Then you have two solutions to implement a TiQR Mobile Endpoint reverse-proxy URL:

a) You can set up an Apache mod_proxy reverse-proxy configuration to your Web server.

b) Or you can use a PHP/ASP TiQR endpoint reverse-proxy script.

> **Note**
>
> Proper TiQR implementation requires SSL for mobile-to-server communications. The TiQR Mobile Endpoint reverse-proxy URL should also be configured on the SSL virtual host of your Apache Web server.

3.2.1 Implementing TiQR Mobile Endpoint with Apache mod_proxy

You will have to edit the Apache httpd.conf file on your public Web server and add a configuration block like:

```
<location "/tiqr">
 ProxyPass http://mywebadmserver:8080/tiqr/mobile
 ProxyPassReverse http://mywebadmserver:8080/tiqr/mobile
</location>
```

This simple configuration will create a reverse-proxy URL that will forward the requests arriving at the public URL `http://mywebsite.com/tiqr/` or `https://mywebsite.com/tiqr/` to the internal TiQR Server endpoint at `http://mywebadmserver:8080/tiqr/mobile/`. In the TiQR Server's settings in WebADM, you need

to set the Mobile Endpoint URL to the public reverse-proxy URL: `http://mywebsite.com/tiqr/` or `https://mywebsite.com/tiqr/`.

3.2.2 Implementing TiQR Mobile Endpoint with a PHP Proxy Script

If your public Web server has PHP installed, you can use our sample PHP Mobile Endpoint reverse-proxy script. Look at Appendix C for a sample PHP reverse-proxy script. In our example, just copy the PHP reverse-proxy script in the tiqr/ directory inside the document root of your Apache Web server and rename it index.php. The reverse-proxy script is also accessible at the URL `http://mywebsite.com/tiqr/` or `https://mywebsite.com/tiqr/`.

You will need to edit the PHP reverse-proxy script and configure the `$endpoint_url` variable like:

```
$endpoint_url = http://mywebadmserver:8080/tiqr/mobile/
```

The PHP reverse-proxy script will forward the requests arriving at the public URL `http://mywebsite.com/tiqr/` or `https://mywebsite.com/tiqr/` to the internal TiQR Server endpoint at `http://mywebadmserver:8080/tiqr/mobile/`. In the TiQR Server's settings in WebADM, you need to set the Mobile Endpoint URL to the public reverse-proxy script URL: `http://mywebsite.com/tiqr/` or `https://mywebsite.com/tiqr/`.

> **Note**
>
> Your PHP installation requires the Curl extension to be installed to run our sample script.

3.2.3 SSL Mobile Endpoints

For a production system, SSL over HTTP (i.e. HTTPS) is mandatory for the TiQR mobile-to-server communications. Instead of implementing the reverse-proxy on your default HTTP Web server's configuration, you may add it in the SSL virtual host of your Web server's configuration. If you use a reverse-proxy script, you may deploy it in the document root of your SSL virtual host.

Then, the Endpoint will be `https://mywebsite.com/tiqr/` and you have to this URL in the TiQR Server's Mobile Endpoint setting.

> **IMPORTANT**
>
> The current version of the TiQR mobile application uses the default mobile phone trusted certificates store for SSL connections to the Internet. Your default WebADM self-signed server certificate may be refused by the mobile phones.

If you do not use a reverse-proxy URL, you may replace the WebADM self-signed SSL certificate with a trusted certificate as explained in Appendix C.

4. Testing your TiQR Server Installation

Your TiQR Server is now working and you can start enrolling a test user.

1. On your iPhone or Android phone, go to the AppStore and search for TiQR. Download and install the application on your mobile.
2. Create a WebADM Account test user in your LDAP tree. Go to the top menu in WebADM, and click the `Create` button. Choose the `WebADM Account` object and create a user with login name `testing`. Alternatively, you can use an existing WebADM user for your tests. Set the Container (LDAP folder) to a location below you Domain User Search Base.

3. Once the user is created, edit it and click the 'QR Login & Signing Server' button in the Application Actions box.

4. Click the 'Register / Unregister TiQR' button to start the TiQR user enrollment.

5. An enrollment QRCode is displayed and you have to start the TiQR application on your mobile phone now. Client the 'scan' button on the TiQR and scan the enrollment QRCode.

6. The TiQR will ask you to confirm the activation of your account and to enter a PIN Code. Follow the instructions and once done, the page should automatically reload and say the TiQR was successfully registered.

You can now test the TiQR user authentication with the 'Test TiQR Login' TiQR action when you edit the user in WebADM. On success, the Login Test will display the user login name and domain to the result screen. The TiQR authentication works as follows:

1. The user scans the QR Code.
2. The user confirms that he wants to log in.
3. The user enters his TiQR PIN Code.
4. The user logged in to the Web application.

Note

If things do not happen as expected, please look at the WebADM Web Services log file /opt/webadm/logs/webadm.log while you register the user or test an authentication.

5. Testing a Web Server Integration

You can download and use the RCDevs sample PHP Login Form for TiQR to experiment with a simple Web integration with TiQR. You can download the sample code archive in the Downloads section on the RCDevs Website. Go to the 'Libraries & Examples' folder and download the 'TiQR Sample PHP Login Form'. Copy the ZIP archive to your public Web server's document root and unzip it. It will create a loginform directory. The testing URL on your Web server will be `http://mywebsite.com/loginform/`. Be sure to have PHP and the PHP-SOAP extension installed on your Web server. On a RedHat server, You can check it with:

```
rpm -q php
rpm -q php-soap
```

Enter the loginform directory and edit the config.php file. You need to adjust the TiQR SOAP web service URL set in server_url. Remember that the web service URLs are displayed in the Applications menu in webADM. You can now go to the login form URL at `http://mywebsite.com/loginform/` with a Web browser to test the sample TiQR login integration.

5.1 Understanding the TiQR Integration

Look at the index.php file in the loginform folder. It is simple and self-explaining. It calls the TiQR SOAP methods to handle the user authentication session. The authentication session works as follows:

1. The page calls the tiqrStart method to get a new QRCode and a session ID. The QRCode contains a random challenge to be 'signed' by the TiQR application.

> It displays the QRCode as an inline image to the user browser. It starts an Ajax loop which will get the session status every 2 seconds by calling the check.php script. This script simply calls the tiqrCheck SOAP method and display the return code to the calling script. The Ajax script will continue looping until the tiqrCheck returns a code different than 2 or 3. Code 2 is the return code when the authentication is still pending (ie. Waiting for mobile authentication). Code 2 is the return code when mobile authentication was successful for the server and is waiting for the user LDAP password.

2. The user scans the QRCode with his TiQR and the TiQR application finally sends its OCRA response to the TiQR Server's Mobile Endpoint.

> The server validates the user authentication by checking if the TiQR OCRA response is valid for the QRCode random challenge.

3. The tiqrCheck (called from the Ajax script) returns a 1 on success or a 0 on failure. The loop stops and the Ajax script reloads the login page. It provides the session ID in a GET variable.
4. The page calls the tiqrCheck to get the authentication status.

> On success the user is authenticated and the SOAP response provides the user login name and the domain name of the authenticated user to the sample PHP script. The script can start a PHP session on the Website.

Appendix A: PHP Mobile Endpoint Reverse Proxy Script

This simple PHP script is included in the TiQR Server package and is accessible in the `/opt/webadm/websrvs/tiqr/doc/` directory.

```
<?php
/*
 * Set the URL of your TiQR server's mobile endpoint here.
 * You do not need to use the HTTPS URL for internal requests.
 */
$endpoint_url = "http://<internal_tiqr_address>:8080/tiqr/mobile/";
/*
 * Add some control here:
 * - The PHP-CURL extension must be loaded.
 * - The external mobile endpoint URL should use SSL.
 * - A TiQR action must be provided.
 */
if (!extension_loaded('curl')) {
 error_log("TiQR-Proxy Error: Curl extension not loaded!");
 header('HTTP/1.1 500 Internal Server Error');
 die;
}
if (!$_SERVER['HTTPS']) {
 error_log("TiQR-Proxy Error: HTTPS endpoint required!");
 header('HTTP/1.1 401 Unauthorized');
 die;
}
if (!isset($_GET['action']) || $_GET['action'] == NULL) {
 error_log("TiQR-Proxy Error: No action provided!");
 header('HTTP/1.1 401 Unauthorized');
 die;
}
// Forward the mobile request to the TiQR server endpoint.
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $endpoint_url.'?'.$_SERVER['QUERY_STRING']);
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, $_POST);
curl_exec($ch);
curl_close($ch);
?>
```

Appendix B: Using a Trusted SSL Certificate for WebADM Web Services

During installation, WebADM generates its own certificate authority (CA) certificate and server SSL certificates. Yet, you can use your own SSL certificates instead of the pre-generated ones.

Using a trusted certificate may be required when you use the RCDevs OpenID IDP or TiQR, and to avoid user browser warnings when accessing the WebApps.

Just replace the SSL certificate and key files in /opt/webadm/pki/httpd.crt and /opt/webadm/pki/httpd.key. WebADM will continue using its own CA certificate for issuing and validating user login certificates (with PKI-based logins) but will use your trusted certificate for the SSL on the HTTP and SOAP services. The certificate and key files must be in PEM format.

If an intermediate certificate chain is required, then just concatenate your certificate file with the chained certificates in the same PEM file.

TiQR Mobile Endpoint PHP Proxy

```
<?php

/*
 * Set the URL of your TiQR server's mobile endpoint here.
 * Typically https://<yourwebadmserver>/ws/tiqr/
 * You do not need to use the HTTPS URL for the internal requests.
 */
$endpoint_url = "https://127.0.0.1/ws/tiqr/";
$failover_url = NULL;

/*
 * Add some control here:
 * - The PHP-CURL extension must be loaded.
 * - The external mobile endpoint URL should use SSL.
 */
if ($endpoint_url == NULL) {
    error_log("TiQR Proxy Error: No endpoint configured!");
    header('HTTP/1.1 500 Internal Server Error');
    die;
}
if (!extension_loaded('curl')) {
    error_log("TiQR Proxy Error: Curl extension not loaded!");
    header('HTTP/1.1 500 Internal Server Error');
    die;
}
if (!$_SERVER['HTTPS']) {
    error_log("TiQR Proxy Error: HTTPS endpoint required!");
    header('HTTP/1.1 401 Unauthorized');
    die;
}
if (!isset($_GET['action']) || $_GET['action'] == NULL) {
    error_log("TiQR Proxy Error: No action provided!");
    header('HTTP/1.1 401 Unauthorized');
    die;
}

$headers = array("HTTP_X_FORWARDED_FOR: ".$_SERVER['REMOTE_ADDR'],
                 "HTTP_X_FORWARDED_HOST: ".$_SERVER['HTTP_HOST']);

// Forward the mobile request to the TiQR server endpoint.
$ch = init_connection($endpoint_url, $headers);
```

```
if (!$ch) {
    error_log("TiQR Proxy Error: Could not initialize Curl!");
    header('HTTP/1.1 500 Internal Server Error');
    die;
}

// try first endpoint URL
$response = curl_exec($ch);

// try second endpoint URL
if (curl_errno($ch) == CURLE_COULDNT_CONNECT && $failover_url != NULL) {
    error_log("TiQR Proxy Warning: Primary endpoint URL failed!");
    curl_close($ch);
    $ch = init_connection($failover_url, $headers);
    if (!$ch) {
 error_log("TiQR Proxy Error: Could not initialize Curl!");
 header('HTTP/1.1 500 Internal Server Error');
 die;
    }
    $response = curl_exec($ch);
}
if (curl_errno($ch) != CURLE_OK) {
    // a cURL error occurred
    error_log("TiQR Proxy Error: ".curl_error($ch));
    header('HTTP/1.1 500 Internal Server Error');
    die;
}
if (!$response) {
    error_log("TiQR Proxy Error: No response from server");
    header('HTTP/1.1 500 Internal Server Error');
    die;
}

$header_size = curl_getinfo($ch, CURLINFO_HEADER_SIZE);
$headers = substr($response, 0, $header_size);
$content = substr($response, $header_size);
curl_close($ch);

$header_array = explode("\r\n", $headers);
foreach ($header_array as $i) {
    // prevent using chunked transfer
    if (strcasecmp($i, "Transfer-Encoding: chunked") == 0) header("Content-Length:
".strlen($content));
    else header($i);
}
echo $content;

function init_connection ($url, $headers) {
    if (!$ch = curl_init()) return false;

    if (!curl_setopt($ch, CURLOPT_URL, $url.'?'.$_SERVER['QUERY_STRING'])) return
false;
    if (!curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true)) return false;
```

```
    if (!curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true)) return false;
    if (!curl_setopt($ch, CURLOPT_RETURNTRANSFER, true)) return false;
    if (!curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false)) return false;
    if (!curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false)) return false;
    if (!curl_setopt($ch, CURLOPT_HEADER, true)) return false;
    if (!curl_setopt($ch, CURLOPT_POST, count($_POST))) return false;
    if (!curl_setopt($ch, CURLOPT_POSTFIELDS, $_POST)) return false;
    if (!curl_setopt($ch, CURLOPT_HTTPHEADER, $headers)) return false;
    return $ch;
}

?>
```