API EXAMPLES

The specifications and information in this document are subject to change without notice. Companies, names, and data used in examples herein are fictitious unless otherwise noted. This document may not be copied or distributed by any means, in whole or in part, for any reason, without the express written permission of RCDevs Security.

WebADM and OpenOTP are trademarks of RCDevs. All further trademarks are the property of their respective owners.

No guarantee is given for the correctness of the information contained in this document. Please send any comments or corrections to info@rcdevs.com.

API examples

API REST API SOAP API

1. Manager API

The Manager interface provides access to some WebADM user management functions and operations exported by your registered applications. The Manager also allows external systems such as Web portals to remotely trigger user management operations and actions from the network.

The user management functions provide LDAP operations such as object creation, update, removal, WebADM settings and data management, etc… The method names for internal management functions are in the form *Manager_Method*.

The operations exported by the registered applications provide access to any feature which is accessible from the application actions in the Admin Portal. The method names for application-exported functions are in the form *Application.Manager_Method*.

The interface communication protocol is based on the JSON-RPC v2.0 specification. You can find the JSON-RPC specification at JSON-RPC 2.0 Specification.

You can go to the Manager Interface page in the WebADM Admin menu to have a full listing of the supported Manager functions and parameters. You can then navigate between applications to get the Manager functions supported by a specific registered application.

The Manager API requires authentication and a WebADM administrator account must be provided to access the interface. The authentication mechanism which is enforced is always the same as the mechanism configured for the WebADM Admin Portal (i.e. The auth_mode setting in the `webadm.conf` file).

Note

Any LDAP permission or OptionSet restriction configured in WebADM will be enforced within the Manager interface. Administrators have also the same level of access in the Manager as they have in the Admin Portal.

- With DN login mode, the administrator DN and password must be provided in the HTTP-Basic Authorization header.
- With UID login mode, the administrator user ID and password must be provided in the HTTP-Basic Authorization header.
- With PKI login mode, the administrator's user certificate must be used for establishing the HTTPs connection to the interface and the administrator password must be provided in the HTTP-Basic Authorization header.

A connection to the Manager automatically creates an Administrator session in WebADM for processing the requested methods if *manager_session* in *webadm.conf* is greater than 0. The Manager responses return a session cookie called WEBADMMANAG in the response headers. You can pass the session cookie in the next Manager requests to avoid starting new sessions.

Note that the Manager sessions have a short expiration time are automatically closed after 10 seconds of inactivity. Yet, you can force the closure of the session by passing the "Connection: close" header to the requests.

The Manager interface is accessible at the URL: `https://<yourserver>/manag/`.

All functions are described in `WebADM` > `Admin` > `Remote Manager Interface` and in following files:

```
/opt/webadm/websrvs/openotp/export.xml
/opt/webadm/websrvs/opensso/export.xml
/opt/webadm/webapps/selfreg/export.xml
/opt/webadm/websrvs/smshub/export.xml
/opt/webadm/websrvs/spankey/export.xml
/opt/webadm/lib/schemas/webadm_export.xml
```

1.1 Examples

Find below a few simple examples of the use of the WebADM Manager interface. The examples are written in PHP and use the cURL extension to send the JSON-RPC call over HTTP.

1.1.1 Resolve the DN of an Existing User

From shell with curl:

```
curl -k\
 --user "cn=admin,o=root:password"\
 --header "Content-Type: application/json"\
 --data '{"method":"Get_User_DN", "params": {"username":"test_user", "domain": "Default"}, "id":0,
"jsonrpc":"2.0"}'\
 https://localhost/manag/
```

With php:

```
<?php
$method = 'Get_User_DN';
$params = array(
 'username' => 'test_user',
 'domain' => 'Default',
);

$request = array(
 'jsonrpc' => "2.0",
 'method' => $method,
 'params' => $params,
 'id' => 0);
$json = json_encode($request);

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://localhost/manag/");
curl_setopt($ch, CURLOPT_USERPWD,"cn=admin,o=root:password");
curl_setopt($ch, CURLOPT_HTTPHEADER, array("connection: close"));
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, $json);
$out = curl_exec($ch);
curl_close($ch);

print_r(json_decode($out));
?>
```

The manager will return a structure in the form:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => cn=test_user,o=Root
    [id] => 0
)
```

If you use PKI Authentication for the manager API, the following example applies with administrator user certificate in pem format:

```php
<?php
$method = 'Get_User_DN';
$params = array(
 'username' => 'test_user',
 'domain' => 'Default',
);
# curl requires full path to certificate files
$caFile = getcwd() . '/ca.crt';
$keyFile = getcwd() . '/admin.key.pem';
$certFile = getcwd() . '/admin.crt.pem';
$certPass = "certpassword";

$request = array(
 'jsonrpc' => "2.0",
 'method' => $method,
 'params' => $params,
 'id' => 0);
$json = json_encode($request);

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://webadm.local/manag/");
curl_setopt($ch, CURLOPT_SSLKEY, $keyFile);
curl_setopt($ch, CURLOPT_CAINFO, $caFile);
curl_setopt($ch, CURLOPT_SSLCERT, $certFile);
curl_setopt($ch, CURLOPT_SSLCERTPASSWD, $certPass);
curl_setopt($ch, CURLOPT_USERPWD,"cn=admin,o=Root:password");
curl_setopt($ch, CURLOPT_HTTPHEADER, array("connection: close"));
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, $json);
$out = curl_exec($ch);
curl_close($ch);

print_r(json_decode($out));
?>
```

1.1.2 Search Email for LDAP Users with the webadmAccount Extension

```php
$method = 'Search_LDAP_Objects';
$params = array(
 'basedn' => 'o=root',
 'filter' => '(objectclass=webadmaccount)',
 'attrs' => array('mail')
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => stdClass Object
        (
            [cn=test1,o=Root] => stdClass Object
                (
                    [mail] => stdClass Object
                        (
                            [0] => test1@mycompany.com
                        )
                )
 [cn=test2,o=Root] => stdClass Object
                (
                    [mail] => stdClass Object
                        (
                            [0] => test2@mycompany.com
                        )
                )
        )
    [id] => 0
)
```

1.1.3 Set the User Mobile Number and Email Address

```
$method = 'Set_User_attrs';
$params = array(
 'dn' => 'cn=test,o=root',
 'attrs' => array('mobile' => array('12345678'), 'mail' => array('test@test.com')),
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

1.1.4 Get the User Mobile Number and Email Address

From shell with curl:

```
curl -k --user "cn=admin,o=root:password"\
 --header "Content-Type: application/json"\
 --data '{"method":"Get_User_Attrs", "params": {"dn":"cn=test,o=root","attrs":
{"0":"mobile","1":"mail"}},"id":0, "jsonrpc":"2.0"}'\
 https://localhost/manag/
```

Will return:

```
{"jsonrpc":"2.0","result":{"mail":{"0":"test@test.com"},"mobile":{"0":"12345678"}},"id":0}
```

With PHP:

```
$method = 'Get_User_attrs';
$params = array(
 'dn' => 'cn=test,o=root',
 'attrs' => array('mobile', 'mail'),
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => stdClass Object
        (
            [mobile] => Array
                (
                    [0] => 12345678
                )

            [mail] => Array
                (
                    [0] => test@test.com
                )

        )

    [id] => 0
)
```

1.1.5 Set Some User Application Settings

```
$method = 'Set_User_Settings';
$params = array(
 'dn' => 'cn=test,o=root',
 'settings' => array('OpenOTP.LoginMode' => 'LDAPOTP', 'OpenOTP.SecureMail' => false),
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

1.1.6 Register a HOTP Token with OpenOTP

```
$method = 'OpenOTP.HOTP_Register';
$params = array(
 'dn' => 'cn=test,o=root',
 'key' => base64_encode("12345678901234567890"),
 'counter' => 0
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

1.1.7 Create a WebADM-Enabled User

```
$method = 'Create_LDAP_Object';
$params = array(
            'dn' => 'cn=test_user,o=root',
            'attrs' => array('objectclass' => array('person','inetorgperson','webadmaccount'),
                             'uid' => array('test_user'),
                             'userpassword' => array('password'),
                             'sn' => array('Test User'))
 );
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

1.1.8 Create an Administrator User and Add Home to the Admin Group

In this example, we send two RPC commands in one single request.

```
$method = 'Create_LDAP_Object';
$params = array(
            'dn' => 'cn=test_admin,o=root',
            'attrs' => array('objectclass' => array('person','inetorgperson'),
                        'uid' => array('test_admin'),
                        'userpassword' => array('password'),
                        'sn' => array('Test Admin'))
);
$request1 = array(
             'jsonrpc' => "2.0",
             'method' => $method,
             'params' => $params,
             'id' => 1
);

$method = 'Set_User_Attrs';
$params = array(
            'dn' => 'cn=other_admins,dc=WebADM',
            'attrs' => array('member' => array('cn=test_admin,o=root')),
            'values' => true
);
$request2 = array(
             'jsonrpc' => "2.0",
             'method' => $method,
             'params' => $params,
             'id' => 2
);

$request = array($request1, $request2);
```

Will return:

```
Array
(
    [0] => stdClass Object
        (
            [jsonrpc] => 2.0
            [result] => 1
            [id] => 1
        )
    [1] => stdClass Object
        (
            [jsonrpc] => 2.0
            [result] => 1
            [id] => 2
        )
)
```

1.1.9 Change a User Password

```
$method = 'Set_User_Password';
$params = array(
 'dn' => 'cn=test,o=root',
 'password' => 'newpassword'
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

1.1.10 Server Status

```
$method = 'Server_Status';
$params = array(
        'servers' => true,
        'webapps' => true,
        'websrvs' => true,
);
```

Will return:

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => stdClass Object
        (
            [version] => 1.6.6-2
            [enabled] => 1
            [servers] => stdClass Object
                (
                    [ldap] => 1
                    [sql] => 1
                    [session] => 1
                    [pki] => 1
                    [push] => 1
                )

            [webapps] => stdClass Object
                (
                    [pwreset] => stdClass Object
                        (
                            [version] => 1.0.8-3
                            [enabled] => 1
                            [status] => 1
                        )

                )

            [websrvs] => stdClass Object
                (
                    [openotp] => stdClass Object
                        (
                            [version] => 1.3.11
                            [enabled] => 1
                            [status] => 1
                        )

                )

            [status] => 1
        )

    [id] => 0
)
```

1.1.11 License Status

The below examples are with curl, but the methods can be called also with PHP or any other JSON-RPC compatible language.

```
curl -k --user "CN=admin,O=root:password" --header "Content-Type: application/json" --data
'{"method":"Get_License_Details", "id":0, "jsonrpc":"2.0"}' https://localhost/manag/
```

Will return:


```
{"jsonrpc":"2.0","result":{"type":"Trial (Cloud-
based)","token_pool":"2\/2","cache_time":862022,"customer_id":"CUSTID01","instance_id":"1","valid_from":"
10-12 00:00:00","valid_to":"2019-10-13 00:00:00","products":{"OpenOTP":
{"maximum_users":"500"},"SpanKey":{"maximum_hosts":"5"},"TiQR":
{"maximum_users":"50"}},"error_message":null},"id":0})
```


1.1.12 Activated user count

```
$method = 'Count_Activated_Users';
$params = array(
);
```

```
(
    [jsonrpc] => 2.0
    [result] => 498
    [id] => 0
)
```

1.1.13 Soft Token Registration with Push

This operation is more complex because it needs a session for the registration with the push.

First, you generate a new key:

```
$method = 'Get_Random_Bytes';
$params = array(
        'length' => '20'
);
```

```
(
    [jsonrpc] => 2.0
    [result] => wU7oGD4R9lktjXtFJmGyGl0wDxE=  # -> $key
    [id] => 0
)
```

You start a new session:

```
$method = 'OpenOTP.Mobile_Session';
$params = array(
        'timeout' => '600'
);
```

```
(
    [jsonrpc] => 2.0
    [result] => aN1JBKnmEMLt3lAV.  # -> $session
    [id] => 0
)
```

You get a registration URI:

```
$method = 'OpenOTP.TOTP_URI';
$params = array(
        'name'    => 'My token',
        'key'     => $key,
        'userid'  => "john",
        'domain'  => "default",
        'session' => $session
);
```

```
stdClass Object
(
    [jsonrpc] => 2.0
    [result] => otpauth://totp/My%20token?
secret=yfhoqgb6ch3fslmnpncsmynsdjotadyr&algorithm=SHA1&digits=6&issuer=My%20Service&period=30

        #  -> $uri
    [id] => 0
)
```

You generate a QR code with that URI, the TXT format is useful for testing in a terminal (use a white screen and not a black screen with inverted text colors), but you can also use GIF or JPG:

```
$method = 'Get_QRCode';
$params = array(
        'uri'    => $uri,
        'format' => 'TXT',
        'margin' => '4',
        'size'   => '1'
);
```

```
(
    [jsonrpc] => 2.0
    [result] => # base64 encoded qrcode
    [id] => 0
)
```

You can show the qrcode with this command:

```
print(base64_decode(json_decode($out, true)['result']));
```

Now, you need to wait that the token is registered with the OpenOTP app:

```
$method = 'OpenOTP.Mobile_Response';
$params = array(
 'session'     => $session,
);
```

```
(
    [jsonrpc] => 2.0
    [result] => 2
    [id] => 0
)
```

Once the result becomes `1`, you can register the token:

```
$method = 'OpenOTP.TOTP_Register';
$params = array(
        'dn'      => 'cn=john,o=Root',
        'key'     => $key,
        'session' => $session
);
```

```
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

1.1.14 Detached Soft Token Registration with Push

In this case the QRCode can be sent separately to the user, and you don’t need to wait for the soft token registration.

First, you generate a new key:

```
$method = 'Get_Random_Bytes';
$params = array(
        'length' => '20'
);
```

```
(
    [jsonrpc] => 2.0
    [result] => wU7oGD4R9lktjXtFJmGyGl0wDxE=  # -> $key
    [id] => 0
)
```

You start a new session, you need to define a pincode to protect the QRCode and the QRCode will be usable until the end of the session:

```
$method = 'OpenOTP.Mobile_Session';
$params = array(
        'timeout' => '600'
        'pincode' => '123456',
);
```

```
(
    [jsonrpc] => 2.0
    [result] => aN1JBKnmEMLt3lAV.  # -> $session
    [id] => 0
)
```

You register the token, it will be added to the user once the mobile app scan successfully the QRCode and not before:

```
$method = 'OpenOTP.TOTP_Register';
$params = array(
        'dn'      => 'cn=john,o=Root',
        'key'     => $key,
        'session' => $session
);
```

```
(
    [jsonrpc] => 2.0
    [result] => 1
    [id] => 0
)
```

You get a registration URI:

```
$method = 'OpenOTP.TOTP_URI';
$params = array(
        'name'    => 'My token',
        'key'     => $key,
        'userid'  => "john",
        'domain'  => "default",
        'session' => $session
);
```

```
(
    [jsonrpc] => 2.0
    [result] => otpauth://totp/My%20token?
secret=yfhoqgb6ch3fslmnpncsmynsdjotadyr&algorithm=SHA1&digits=6&issuer=My%20Service&period=30

        #  -> $uri
    [id] => 0
)
```

You generate a QR code with that URI, the TXT format is useful for testing in a terminal (use a white screen and not a black screen with inverted text colors), but you can also use GIF or JPG:

```
$method = 'Get_QRCode';
$params = array(
        'uri'    => $uri,
        'format' => 'TXT',
        'margin' => '4',
        'size'   => '1'
);
```

```
(
    [jsonrpc] => 2.0
    [result] => # base64 encoded qrcode
    [id] => 0
)
```

You can show the qrcode with this command:

```
print(base64_decode(json_decode($out, true)['result']));
```

1.1.15 Signing a certificate signing request (CSR)

The manager API allows you to submit a CSR, which will signed by WebADM PKI service and a final certificate returned.

You can generate the CSR with any tool, but in this example we use OpenSSL. For example the below command will generate a private key and associated CSR for a `User certificate` for username `test-cert` in WebADM User Domain `Default`:

```
openssl req -new -newkey rsa:2048 -nodes -keyout user.key  -out user.csr  -subj '/CN=Default\\test-cert/UID=test-cert/DC=Default/description=USER/SN=test-cert'
```

To generate an `Admin` certificate which can be used for WebADM and Manager API authentication you can use the below command. The distinction between User and Admin certificate is the `description` field.

```
openssl req -new -newkey rsa:2048 -nodes -keyout admin.key -out admin.csr -subj '/CN=cn=test-cert,o=root/description=ADMIN/SN=test-cert'
```

When you have the CSR, you can have it signed with the Manager API:

```php
<?php
$method = 'Sign_certificate_Request';
$params = array(
        'request' => file_get_contents("user.csr"),
       );

$request = array(
 'jsonrpc' => "2.0",
 'method' => $method,
 'params' => $params,
 'id' => 1);
$json = json_encode($request);

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://localhost/manag/");
curl_setopt($ch, CURLOPT_USERPWD,"Default\\admin:password");
curl_setopt($ch, CURLOPT_HTTPHEADER, array("connection: close"));
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, $json);
$out = curl_exec($ch);
curl_close($ch);

print_r(json_decode($out));
?>
```

This will return the certificate in PEM format. To use the certificate for authentication, it must be registered on a user, you can use the `Set_User_attrs` method for this:

```php
$cert = file_get_contents("user.crt");
$method = 'Set_User_Attrs';
$params = array(
    'dn' => 'cn=test-cert,o=root',

    'attrs'=> array(
    'usercertificate'=> array(preg_replace ( '/(-----.*-----)|\s/','', $cert)),
  ),
    'values' => True
);
```

2. Web Services API

2.1 SOAP API

Web services are available through a SOAP API. API description and wsdl file are available here:

- openotp (Authentications) `https://<my_webadm_server>/websrvs/wsdl.php?websrv=openotp`
- smshub (Sending SMS) `https://<my_webadm_server>/websrvs/wsdl.php?websrv=smshub`
- opensso(Single Sign-On) `<my_webadm_server>/websrvs/wsdl.php?websrv=opensso`

2.1.1 Example

The example is written in PHP and use the SOAP extension.

```
<?php

$soap_client = new SoapClient("https://localhost/websrvs/wsdl.php?websrv=openotp");

$username = "test_user";
$ldapPassword = "foo";

$response = $soap_client->openotpNormalLogin($username, null, $ldapPassword);

print_r($response);

?>
```

With PHP versions later than 5 SSL peer verification is on by default and must be disabled unless the server is using CA trusted by the client.

```
<?php

$sctx = stream_context_create(array('ssl' => array('verify_peer' => false, 'verify_peer_name' => false)));

$soap_client = new SoapClient("https://localhost:8443/openotp?wsdl", array('stream_context' => $sctx));

$username = "test_user";
$ldapPassword = "foo";
$otp = "123456";

$response = $soap_client->openotpNormalLogin($username, null, $ldapPassword,$otp);

print_r($response);

?>
```

2.2 REST API

Authentication is also possible with a REST API. You can send information with GET, POST or POST-JSON. Functions and attributes are the same as with SOAP API.

If you wish to secure the access to the REST API with a certificate, you can configure this in `WebADM Applications` > `MFA Authentication server` > `CONFIGURE` > `Require Client Certificate`. Once this is enabled, you have to issue Client certificates for the API clients in `WebADM` > `Admin` > `Issue Server or Client SSL Certificate`.

2.2.1 Example with GET:

```
wget "https://localhost:8443/openotp/json/openotpNormalLogin/?
username=test_user&ldapPassword=foo"

wget "https://localhost:8443/openotp/json/?
method=openotpNormalLogin&username=test_user&ldapPassword=foo"
```

With certificate authenticating the client:

```
wget --certificate=client.crt --no-check-certificate \
"https://localhost:8443/openotp/json/openotpNormalLogin/?username=test_user&ldapPassword=foo"
```

2.2.2 Example with POST-JSON:

```
wget --post-data='{"username":"test_user","ldapPassword":"foo"}' \
"https://localhost:8443/openotp/json/openotpNormalLogin/"
```

With certificate authenticating the client:

```
wget --certificate=client.crt --no-check-certificate \
--post-data='{"username":"test_user","ldapPassword":"foo"}' \
"https://localhost:8443/openotp/json/openotpNormalLogin/"
```