Boomi Cloud™ API Management - Local Edition

Security Guide

Version 5.6.2 | November 2024

Contents

Introduction

This document describes guidelines to ensure security within the various components of Boomi Cloud™ API Management - Local Edition (CAM) cluster.

Securing the Boomi Cloud™ API Management - Local Edition Cluster

The following sections describes various ways to secure the Boomi Cloud API Management - Local Edition (Local Edition) clusters.

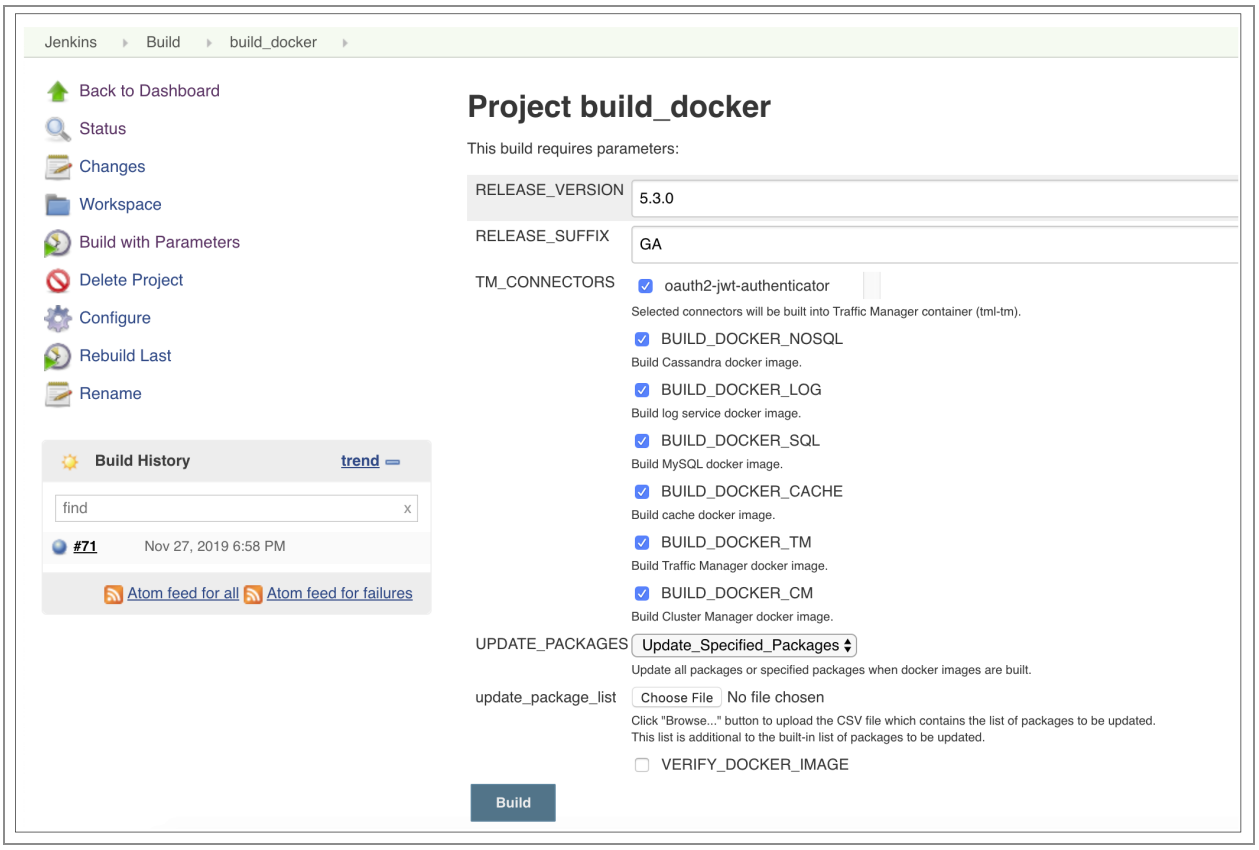
Patching Images during Image build

The image build process pulls libraries from various repositories and are installed through the yum utility.

To patch images during the build, choose the update libraries option.

Creating Local Edition Docker Images

The Jenkins job, build_docker is used to create the Local Edition Docker images.



RELEASE_VERSION

The release version, together with build number, are used to compose the docker image tag.

In the screenshot, RELEASE_VERSION is "5.3.0"; on the left pane of screenshot, it shows the build "#71" has been done, so Docker images in build #71 have tag "v5.3.0.GA.71".

RELEASE_SUFFIX

The suffix used in conjunction with the release version. For example, GA, HF1 (for Hotfix 1), and so on.

TM_CONNECTORS

In the TM_CONNECTORS section, you can choose to build "OAuth2 JWT Authenticator" into the Traffic Manager container (tml-tm).

You can also choose the following Docker images to build:

- BUILD_DOCKER_NOSQL - NoSQL container (tml-nosql)
- BUILD_DOCKER_LOG - Log container (tml-log)
- BUILD_DOCKER_SQL - SQL container (tml-sql)
- BUILD_DOCKER_CACHE - Cache container (tml-cache)
- BUILD_DOCKER_TM - Traffic Manager container (tml-tm)
- BUILD_DOCKER_CM - Cluster Manager container (tml-cm)

Note: You should verify the sizes of the images that are created. They should be approximately the following sizes after creation:

- tml-nosql - 918MB
- tml-log - 1.08GB
- tml-sql - 1.65GB
- tml-cache - 848MB
- tml-tm - 857MB
- tml-cm - 1.13GB

UPDATE_PACKAGES

The administrator can choose:

- **Update_Specified_Packages**: update specified packages during build;
- **Update_All_Packages**: all packages will be updated during build.

update_package_list:

Uploads a csv file that contains the list of packages to be updated during build. This list is applied to all Docker images.

Here is an example content of a csv file:

```
python.x86_64
libssh2
systemd
```

```
bind-license
openssl-libs
device-mapper.x86_64
device-mapper-libs.x86_64
```

Configuring AWS Access in the Installer

To configure access to AWS in the installer, run the configure_aws Jenkins job:

Complete the Project configure_aws dialog as follows:

- **aws_access_key_id** - The AWS access key ID is required.
- **aws_secret_access_key** - The AWS secret access key is required.
- **aws_default_region** - This is the region in which the AWS Kubernetes cluster and the Local Edition cluster are being created.
- **aws_role_arn** - The role Amazon Resource Name (ARN). This is required if AWS role-based access control is used. An example role ARN:
  arn:aws:iam::123456789012:role/TIBCO/Administrator

Securing Deployment

All the Boomi Cloud™ API Management - Local Edition containers run as restricted users. It is a good practice to not run them as root or users with higher permissions.

Authentication and Authorization

All the default passwords must be changed during and after deployment.

Developer Portal

All the default passwords of the developer portal must be changed.

Changing the Administrator Password

The default administrator password can be changed from the developer portal.

Before you begin

Make sure all components are active and the cluster is up and running

```
cm ls components
Using cluster [aniket-gcp-cluster-single-host-172]
Using Zone [us-central1-a]
Component ID                    Type          Name                                                                    Status      Last
Heartbeat Received     Host          Service Port(s)
--------------------------------  -----------    ------------------------------------------------------------------------ -----------
-----------------------    --------      -------------------
33319687-0658-44b9-baa4-409825c930b5  cache         cache-set-0-0.cache-svc-
0.default.svc.cluster.local               ACTIVE      May 24 2021 21:12:18 +0000  10.120.0.14
11212,11211,11213,11214,11215,11216
d5b21d3e-7d9f-4d80-80b4-2afd57177a9a  configmanager   cm-deploy-0-5ccc99468f-m6s95.cm-
svc-0.default.svc.cluster.local     ACTIVE      May 24 2021 21:12:09 +0000  10.120.0.11    7080

3f9262a3-470e-4557-806e-4a23659d5fb6  logservice     log-set-0-0.log-svc-
0.default.svc.cluster.local                ACTIVE      May 24 2021 21:12:19 +0000  10.120.0.12
24224
da4e3f02-3d0c-45ef-9595-25ac200f3992  nosql         cass-set-0-0.cass-svc-
0.default.svc.cluster.local                ACTIVE      May 24 2021 21:12:22 +0000  10.120.0.10    9042

afc6a10d-1ca6-4bd8-bd51-7dd413561efe  sql           mysql-set-0-0.mysql-svc-
0.default.svc.cluster.local               ACTIVE      May 24 2021 21:12:12 +0000  10.120.0.13    3306

0ab8c275-13e0-40d0-b486-96f7f64706a2  trafficmanager  tm-deploy-0-bc677b4c-c728g.tm-svc-
0.default.svc.cluster.local       ACTIVE      May 24 2021 21:12:15 +0000  10.120.0.15    8080
```

To change the default administrator password,

Procedure

1. Launch the developer portal.

   ```
   https://<cm_svc_ip>:8443
   ```

2. Navigate to **Localdev Admin > Profile** and click **Change Password**. Input the current and the new password
3. Navigate to the properties folder from the installation directory.

```
<root_dir>/docker-deploy/properties
```

4. Add new field by replacingpassword withnew_password as its value in the tml_cm_properties.json file

```
"maxIdleTime" : 300,
"api_debug_key" : "24NumbersAndOrCharacters",
"api_debug_secret" : "10NumChars",
"password" : "<new_password>"
```

5. Recompose the manifest file to generate the updated artifacts.

```
./compose.sh <mainfest_json_file>
```

6. Navigate to the manifest folder and undeploy the cm pod.

```
./undeploy-cm-pod.sh
secret "cm-property" deleted
secret "cm-jks" deleted
secret "cm-crt" deleted
secret "cm-key" deleted
secret "cm-resource" deleted
deployment.apps "cm-deploy-0" deleted
```

7. Redeploy the cm pod. The new password will come in effect now.

```
./deploy-cm-pod.sh
secret/cm-property created
secret/cm-jks created
secret/cm-crt created
secret/cm-key created
secret/cm-resource created
deployment.apps/cm-deploy-0 created

cm-deploy-0-5ccc99468f-gfz7r   1/1    Running   0        10s
```

What to do next

- Ensure all components are in active state

```
cm ls components
Using cluster [aniket-gcp-cluster-single-host-172]
Using Zone [us-central1-a]
Component ID                    Type          Name                                                                    Status
Last Heartbeat Received    Host          Service Port(s)
--------------------------------  -----------    ------------------------------------------------------------------------ ---
--------   ------------------------   --------      -------------------
33319687-0658-44b9-baa4-409825c930b5  cache         cache-set-0-0.cache-svc-
0.default.svc.cluster.local               ACTIVE      May 24 2021 21:12:18 +0000  10.120.0.14
11212,11211,11213,11214,11215,11216
d5b21d3e-7d9f-4d80-80b4-2afd57177a9a  configmanager   cm-deploy-0-5ccc99468f-
m6s95.cm-svc-0.default.svc.cluster.local     ACTIVE      May 24 2021 21:12:09 +0000
10.120.0.11    7080
3f9262a3-470e-4557-806e-4a23659d5fb6  logservice     log-set-0-0.log-svc-
0.default.svc.cluster.local                 ACTIVE      May 24 2021 21:12:19 +0000  10.120.0.12
 24224
da4e3f02-3d0c-45ef-9595-25ac200f3992  nosql         cass-set-0-0.cass-svc-
0.default.svc.cluster.local                 ACTIVE      May 24 2021 21:12:22 +0000  10.120.0.10
9042
afc6a10d-1ca6-4bd8-bd51-7dd413561efe  sql           mysql-set-0-0.mysql-svc-
0.default.svc.cluster.local               ACTIVE      May 24 2021 21:12:12 +0000  10.120.0.13
3306
0ab8c275-13e0-40d0-b486-96f7f64706a2  trafficmanager  tm-deploy-0-bc677b4c-c728g.tm-
svc-0.default.svc.cluster.local       ACTIVE      May 24 2021 21:12:15 +0000  10.120.0.15
8080
```

- To verify the new password, login the control center using the new password.

  ```
  https://<cm_svc_ip>:8443/admin
  ```

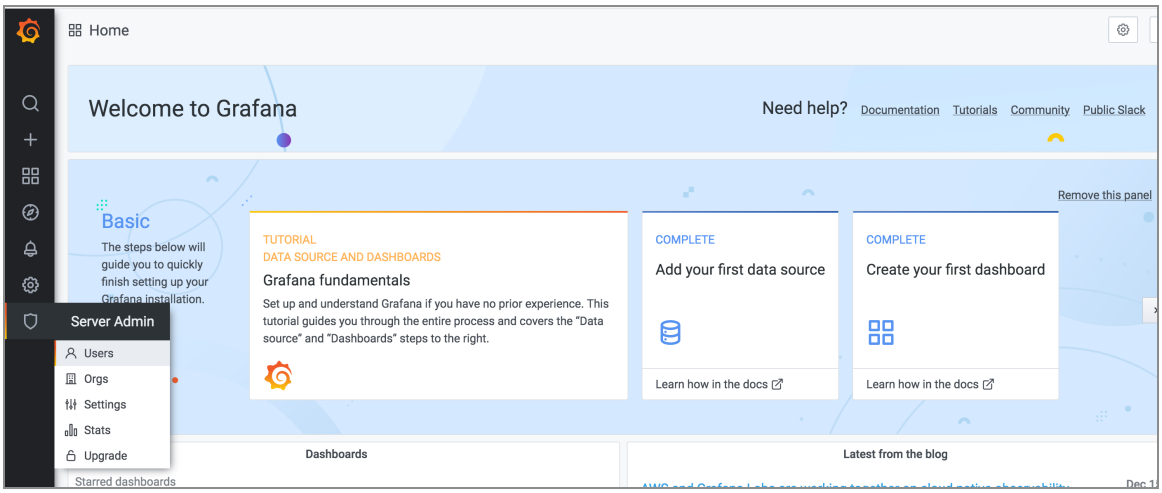
  You can carry out operations using the latest credentials.

Reporting

Managing Users and Roles

The **masheryadmin** user can create users and assign roles according to their needs.

1. Login to Grafana dashboard as the **masheryadmin** user.
2. On the left panel, navigate to **Server Admin Shield** option.



3. Follow the Grafana documentation for instructions on creating and adding a new user.
4. Assign **Roles (Permissions)** as described in the Grafana documentation on Permissions.

Port Configurations

The below are the port configurations available for deployment with options to configure https

Parameter	Port
tml_tm_http_port	80
tml_tm_https_port	443
tml_tm_mhttps_port	1443
tml_api_http_port	7080
tml_api_https_port	7443
tml_cm_http_port	8080
tml_cm_https_port	8443

Securing Cassandra

Cassandra nodes communicate with each other via the Gossip protocol. In multi zone or multi region networks, it becomes necessary to communicate via SSL.

Enable SSL on Cassandra inter node communication.

Enable encrypted inter-node communication in Cassandra cluster

Since Cassandra nodes communicate across regions via public network, it is highly recommended to encrypt inter-node communication in Cassandra cluster.

For example,

```
{
 "auto_binding":"ON",
 "dependency_health_check": "OFF",
 "server_internode_encryption": "all",
 "server_keystore_password": "changeme",
 "server_truststore_password": "changeme",
 "server_require_client_auth": true
}
```

Securing MySQL

This sections describes how to change the default passwords for MySQL.

Changing the MySQL Password of a Running Cluster

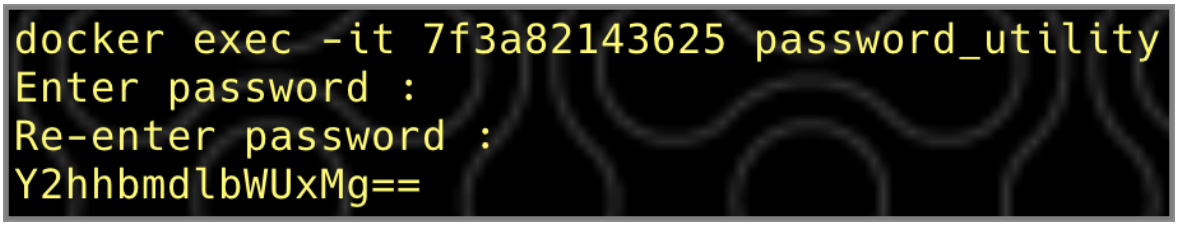
The following section describes how to change the MySQL password of a running cluster in Kubernetes.

If you have a running Local Edition cluster in a Kubernetes environment, the cluster was created using the MySQL password specified in the tml_cluster_properties.json file.

To change the MySQL password:

Procedure

1. Encrypt the password using the password utility shipped with the installer :
   docker exec -it <installer container id> password_utility.



2. After encrypting the password, change the password in the tml_cluster_properties.json file in the deployment folder.

   For example, if you deployed from the /jdoe/tml-530/docker-deploy/kubernetes/manifest folder, modify the tml_cluster_properties.json file in that folder.

   In this example, modify the below file by updating the generated encoded password as Y2hhbmdlbWUxMg==

```
{
      "mysql_root_pwd": "Y2hhbmdlbWUxMg==",
      "mysql_masheryonprem_pwd": "Y2hhbmdlbWUxMg==",
      "mysql_mashonpremrepl_pwd": "Y2hhbmdlbWUxMg==",
      "mysql_mashclient_pwd": "Y2hhbmdlbWUxMg==",
      "mysql_masherybackup_pwd": "Y2hhbmdlbWUxMg==",
      "mom_server": "https://api-mom.mashery.com",
      "mom_key": "",
      "mom_secret": ""
}
```

3. Delete the cluster-property Kubernetes secret:

```
kubectl delete secret cluster-property
```

4. Create the Kubernetes secret again using the tml_cluster_properties.json file containing the new password:

```
kubectl create secret generic cluster-property --from-file=./tml_cluster_properties.json
```

5. Log in to the MySQL container:

```
kubectl exec -it mysql-set-0-0 /bin/bash
```

6. Connect to the MySQL server with a MySQL client using the old password, then set the new password.

```
mysql -u root -p
```

Execute the following commands on the MySQL prompt:

```
use masherysolar;
FLUSH PRIVILEGES;
ALTER USER 'root'@'localhost' IDENTIFIED BY 'changeme12';
ALTER USER 'masheryonprem'@'localhost' IDENTIFIED BY 'changeme12';
ALTER USER 'masheryonprem'@'%' IDENTIFIED BY 'changeme12';
ALTER USER 'mashonpremrepl'@'%' IDENTIFIED BY 'changeme12';
ALTER USER 'mashclient'@'%' IDENTIFIED BY 'changeme12';
ALTER USER 'masherybackup'@'%' IDENTIFIED BY 'changeme12';
```

7. Exit the MySQL container.
8. Delete the MySQL pod.

```
kubectl delete pod mysql-set-0-0
```

9. Once mysql-set-0-0 is created again, wait for it to become ACTIVE. Then delete all the cache pods; for example, if you have three cache pods, delete all as follows:

```
kubectl delete pod cache-set-0-0 cache-set-0-1 cache-set-0-2
```

10. Once cache pods are created again, delete all the tm pods; for example, if you have three tm pods, delete all as follows:

```
kubectl delete pod tm-deploy-0-764874c9d8-2gl5x tm-deploy-0-4c9d764878-2gl5x tm-deploy-0-747648c9d8-2gl5
```

11. Once the tm pods are created again, delete the cm pod as follows:

```
kubectl delete pod cm-deploy-0-7b9976697c-zblbk
```

12. Once all the pods are running, you can login to the cm pod and verify that the

status is ACTIVE. Likewise, any existing service endpoint that was created before you applied the password changes should work as before.

Encrypting MySQL Replication Group

The following section describes how to enable encryption for a MySQL replication group.

Update properties/tml_sql_properties.json

Change the following property from default value "false" to "true":

```
"mysql_group_replication_encryption": true
```

Update Key and Certificates

Keys and certificates can be created using either MySQL or openssl.

Creating SSL and RSA Certificates and Keys using MySQL

For more information on how to create SSL and RSA certificates and keys using MySQL, see MySQL Reference Manual.

For example:

```
mysql_ssl_rsa_setup --datadir=${PWD}
```

This generates the following keys and certificates:

- ca-key.pem
- ca.pem
- client-cert.pem
- client-key.pem
- private_key.pem
- public_key.pem
- server-cert.pem
- server-key.pem

Creating SSL Certificates and Keys using openssl

For more information on how to create SSL certificates and keys using openssl, see MySQL Reference Manual.

Overwrite the Example Keys and Certificates

Use the generated keys and certificates to overwrite the following files:

- properties/tml-sql-ca.pem
- properties/tml-sql-server-cert.pem
- properties/tml-sql-server-key.pem

Securing Traffic

It is a good practice to use the HTTPS configuration. There are many ways in which traffic calls are secured, One way SSL, Mutual SSL, and so on.

HTTPS Configuration Overview

Non-mutual HTTPS

Message flow:

Client --(HTTPS 1)--> Customer Load Balancer --(HTTPS 2)--> Local Edition instance --(HTTPS 3)--> Backend Service

In the above flow:

1. HTTPS 1 is achieved between the Client and the Customer Load Balancer by appropriately configuring the Load Balancer. This is outside the scope of Local Edition.
2. HTTPS 2 configuration is what we refer to as the HTTPS Server feature. Since the Load Balancer and the Local Edition instance are in the customer's network, mutual SSL is currently not supported in the HTTPS server feature.
3. HTTPS 3 configuration is what we refer to as HTTPS Client feature. Since this call typically goes across networks, we support mutual SSL settings by configuring an

HTTPS Client profile with Identity and Trust settings, and associating the profile with the endpoint configuration. The required configuration is documented in this section.

Mutual HTTPS

Local Edition can be configured for mutual HTTPS authentication (server side). To accomplish this, you deploy a totally separated Local Edition cluster with mutual HTTPS authentication on it.

- In **tethered mode**, this totally separated Local Edition cluster syncs with a separated area in which all APIs to be protected by mutual HTTPS authentication are created.
- In **untethered mode**, you author all APIs to be protected by mutual HTTPS authentication using Configuration Manager. All APIs are confined in this separated Local Edition cluster.

HTTPS Server Configuration

This following section describes the HTTPS Server Configuration.

Traffic Manager as an HTTPS Server with Mutual SSL

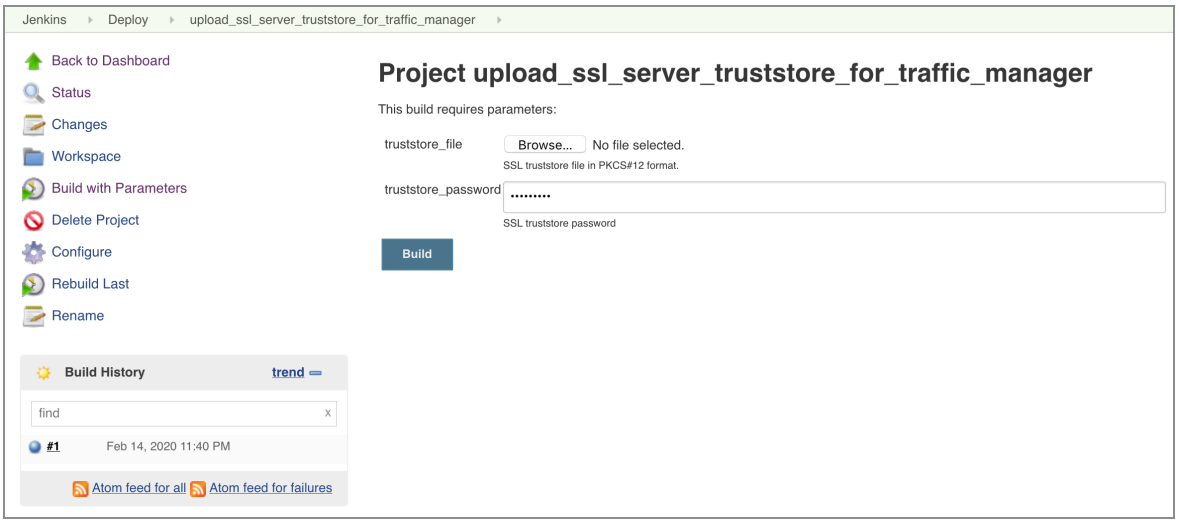
Perform the following steps to set up Traffic Manager as an HTTPS server with mutual SSL.

Procedure

1. Run the upload_ssl_server_truststore_for_traffic_manager Jenkins job in the Installer to upload the trust store.

   Complete the fields in the Jenkins job as follows:

   - truststore_file - The key store in PKCS#12 format, which holds all Certificate Authority (CA) certificates which are trusted.
   - truststore_password - The password protecting the trust store.



The upload_ssl_server_truststore_for_traffic_manager Jenkins job uploads the trust store (tml-tm-trust.jks) to the /var/jenkins_home/docker-deploy/properties folder. This trust store holds all trusted CA certificates.

> **Note:** There is a built-in sample, self-signed root CA certificate. You should upload your own trust store in the Local Edition installer

2. Configure the following property in the /var/jenkins_home/docker-deploy/properties/tml_tm_properties.json file:

   - tm_truststore_password - The password protecting the trust store.

   For example:

   ```
   "tm_truststore_password": "changeme",
   ```

3. Configure the following property in the manifest-onprem-swarm.json file:

   - tml_tm_mhttps_enabled - Set to true to turn on mutual HTTPs authentication.

   For example:

   ```
   "tml_tm_mhttps_enabled": false,
   ```

4. Verify your mutual HTTPS authentication configuration using the following example curl command:

```
curl -k -v --key PATH_TO_KEY/yam_root_.pkcs8 --cert PATH_TO_CERT/yam_
root_.cer:changeme -H 'host: calypsoqa.api.mashery.com' https://$LB:443/mock?api_
key=mycustomkey
```

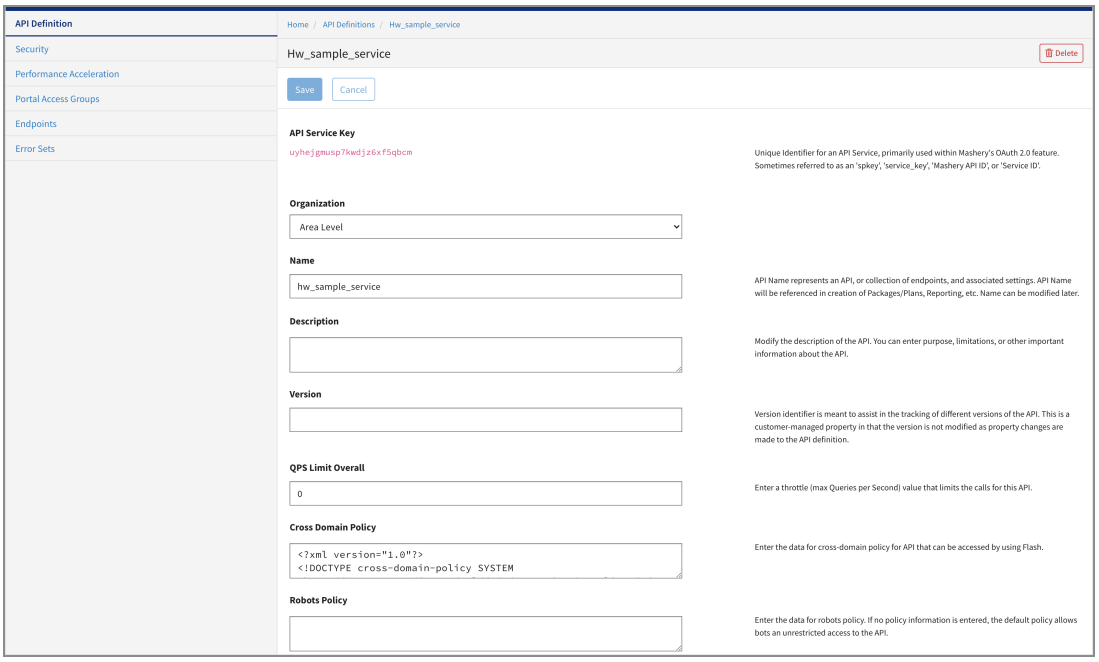
Note: In yam_root_.cer:changeme, "yam_root_.cer" is the certificate file name, and "changeme" is the trust store password protecting the private key. "LB" is the public IP address of the Load Balancer for Traffic Manager.

HTTPS Client Configuration

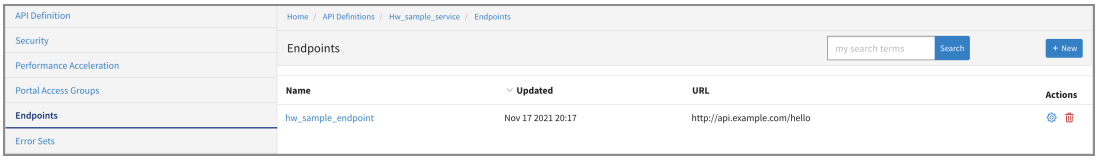
The following example walks through a use case where the CLI is used to manage the components within the cluster. The example also uses the (Untethered) Configuration Manager UI where necessary.

HTTPS Endpoint Configuration

1. Set up an HTTPS Endpoint using the HTTPS Client Profile.
    a. Open the Configuration Manager and sign in.
    b. Under **API Definitions** open the **hw_sample_service** API for edit.

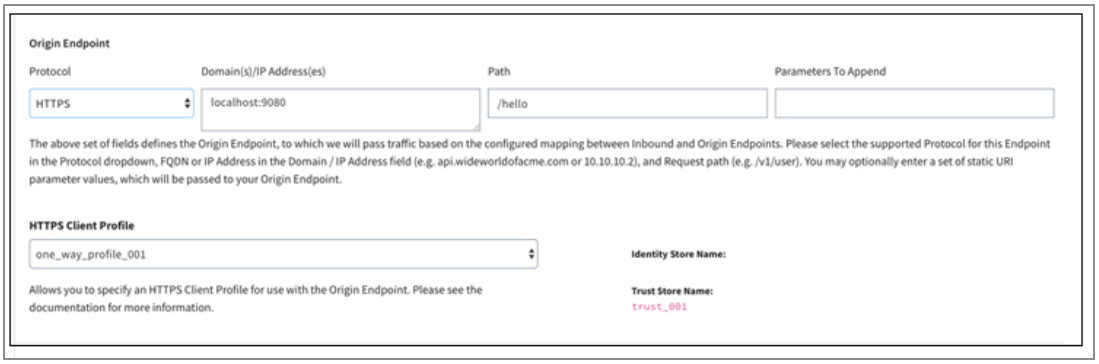


c. Click on **Endpoints** from the left pane.



d. Click on the **hw_sample_endpoint** link in the panel.

e. Select HTTPS protocol for the Origin Endpoint backend.



Testing the HTTPS Client Call

To test the HTTPS Client call, start testing traffic using CURL:

1. From within the Cluster Manager CLI container:

```
[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager ls components
Using cluster [Local Edition Reference Cluster]
Using Zone [local]
Component ID                          Component Type      Component Name      Component Status
Component Host      Component Agent Port   Component Service Port(s)
------------------------------------ ------------------- ------------------- ------------------- ------------------- -
--------------------- --------------------------
0adf6ab7-19a3-4598-9c4e-f9e20729448f  sql             sql             ACTIVE
10.244.1.5          9080            3306
bd465cb3-2c19-4b89-afc8-f4719cd43d7b  nosql           nosql           ACTIVE
10.244.1.2          9080            9042
37e880ce-6d05-4a6f-bdae-cee24a70b6a0  cache           cache           ACTIVE
10.244.1.6          9080            11212,11211,11213,11214,11215,11216
f8f6da07-6669-413c-b356-cdb26be4f4df  trafficmanager      tm              ACTIVE
10.244.1.7          9080            8080
74062084-0d77-4a43-9551-07db90e8fdd9  logservice          log             ACTIVE
10.244.1.4          9080            24224

[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# curl -H 'Host:api.example.com'
'http://x.x.x.x/hw_path_01?api_key=sra663fmnshjazx5sv2mgtmw'
{"time":"2018-11-28 20:17:02.841+0000","message":"Hello world"}
```

2. From load balanced TM service:

> **Note:** Note the Kubernetes service for the Traffic Manager ("tm-svc" in this case).

```
admin-MBP15:properties <admin>$ kubectl get svc
NAME         TYPE         CLUSTER-IP      EXTERNAL-IP       PORT(S)
AGE
cass-svc-0   ClusterIP    None            <none>            9042/TCP                        16h
kubernetes   ClusterIP    100.64.0.1      <none>            443/TCP                         17h
log-svc-0    ClusterIP    None            <none>            24224/TCP                       16h
mysql-svc-0  ClusterIP    None            <none>            3306/TCP                        16h
tm-svc       LoadBalancer   100.71.130.12   a45e45c83f2cb...
80:32299/TCP,443:32117/TCP,8083:31296/TCP   16h


admin-MBP15:properties <admin>$ kubectl describe service tm-svc|grep Ingress|awk -F' ' '
{print $3}'
a45e45c83f2cb11e8bc330609f0835b9-95933830.us-east-1.elb.amazonaws.com

admin-MBP15:properties <admin>$ curl -H 'Host:api.example.com'
```

```
'http://a45e45c83f2cb11e8bc330609f0835b9-95933830.us-east-1.elb.amazonaws.com/hw_
path_01?api_key=sra663fmnshjazx5sv2mgtmw' -k
{"time":"2018-11-28 21:15:04.540+0000","message":"Hello world"}
```

Troubleshooting HTTPS Problems

Enabling Java SSL Debug Logging

To enable Java SSL debug logging for Local Edition, follow the steps below:

Procedure

1. Add the following setting in /opt/javaproxy/proxy/proxy.ini in the ml-tm container:

   ```
   -Djavax.net.debug=all
   ```

2. Restart javaproxy:

   ```
   service javaproxy restart
   ```

3. Send requests to Local Edition and watch the log in /var/log/mashery/javaproxy-runtime.log. For example:

   ```
   tail -f /var/log/mashery/javaproxy-runtime.log
   ```

HTTPS Mutual Authentication

The following sections describe how to set up and create HTTPS mutual authentication.

- Setting up HTTPS Authentication
- Creating Mutual Authentication

Setting up HTTPS Authentication

HTTPS client profile is used to provide certificate/identity based authentication to endpoints. This can be done in two ways:

- One-way HTTPS authentication
- Mutual authentication

One-Way HTTPS Client Configuration

To create a one-way HTTPS client configuration:

Note: The following procedure is illustrated on a Kubernetes system.

1. Log in to the CLI container.

```
kubectlexec-it cm-deploy-0-76ffbdbb6b-jjb44envCOLUMNS=$COLUMNS LINES=$LINES
TERM=$TERMbash
```

2. In the command line interface, input the scope of operation as below:

```
[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager ls clusters
Cluster ID                           Cluster Name
------------------------------------ --------------------
6b08d0e3-637e-4e18-b628-2b934db0abfc  Local Edition

[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager use cluster 6b08d0e3-637e-
4e18-b628-2b934db0abfc
Using cluster [Tibco Mashery Local Reference Cluster]
[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager ls zones
Using cluster [Local Edition Reference Cluster]
Zone ID                              Zone Name
------------------------------------ --------------------
0f7c76d8-feb6-4d65-8dd9-532e43fe7eef  local
[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager use zone 0f7c76d8-feb6-
4d65-8dd9-532e43fe7eef
Using cluster [Local Edition Reference Cluster]
Using Zone [local]
```

3. Upload the trust certificate.
    a. Copy the trust certificate file to the CLI container.

```
admin-MBP15:properties admin$ kubectl get pods
NAME                        READY    STATUS    RESTARTS   AGE
cache-deploy-0-577f7d5c7b-42pfn   1/1      Running   0        47m
```

```
cass-set-0-0                  1/1     Running   0        48m
cm-deploy-0-76ffbdbb6b-jjb44     1/1     Running   0        47m
log-set-0-0                   1/1     Running   0        47m
mysql-set-0-0                 1/1     Running   0        47m
tm-deploy-0-5b584758bb-9m6cn     1/1     Running   0        47m


admin-MBP15:properties admin$ kubectl cp ~/Downloads/samplecert.cer cm-deploy-0-76ffbdbb6b-jjb44:/usr/local/bin/
```

b. Upload the trust certificate as shown:

```
[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager create certificate --file /usr/local/bin/samplecert.cer
Creating certificate for cluster ID 6b08d0e3-637e-4e18-b628-2b934db0abfc and zone ID 0f7c76d8-feb6-4d65-8dd9-532e43fe7eef
Successfully created certificate for cluster ID 6b08d0e3-637e-4e18-b628-2b934db0abfc of zone ID 0f7c76d8-feb6-4d65-8dd9-532e43fe7eef
```

c. For tethered mode, update the trust as shown:

```
[root@cm-deploy-0-76ffbdbb6b-jjb44 builder]# clustermanager set certificate --uuid cb4d2bc2-f863-440b-1001-1d11369404d8 --file /usr/local/bin/samplecert.cer
Updating certificate for cluster ID 6b08d0e3-637e-4e18-b628-2b934db0abfc and zone ID 0f7c76d8-feb6-4d65-8dd9-532e43fe7eef
Successfully updated certificate for cluster ID 6b08d0e3-637e-4e18-b628-2b934db0abfc of zone ID 0f7c76d8-feb6-4d65-8dd9-532e43fe7eef
```

4. Open the Configuration Manager and sign in. Create the new configuration.

Mutual HTTPS Client Configuration

In addition to the previous steps mentioned for one-way HTTP Client Configuration profiles, do the following:

1. List all the existing identities:

```
[root@cm-deploy-0-7d69cb94f7-g4gj9 builder]# clustermanager ls identities
clusterId [6bb97326-007d-4ba3-828e-24bb81449b94] and zoneId [5d75a537-a6b4-489c-8d5a-d97469a7390a]
Using cluster name [Raj Cluster]
Using Zone name [us-east-1e]
UUID                                 Name                                 Area ID    Expiration
------------------------------------ ------------------------------------ ---------- --------------------
cb4d2bc2-f863-440b-0001-1d11369404d8 identity_001                         5488
cb4d2bc2-f863-440b-0002-1d11369404d8 identity_002                         5488
cb4d2bc2-f863-440b-0003-1d11369404d8 identity_003                         5488
cb4d2bc2-f863-440b-0004-1d11369404d8 identity_004                         5488
cb4d2bc2-f863-440b-0005-1d11369404d8 identity_005                         5488
cb4d2bc2-f863-440b-0006-1d11369404d8 identity_006                         5488
cb4d2bc2-f863-440b-0007-1d11369404d8 identity_007                         5488
cb4d2bc2-f863-440b-0008-1d11369404d8 identity_008                         5488
cb4d2bc2-f863-440b-0009-1d11369404d8 identity_009                         5488
cb4d2bc2-f863-440b-0010-1d11369404d8 identity_010                         5488
cb4d2bc2-f863-440b-0011-1d11369404d8 identity_011                         5488
cb4d2bc2-f863-440b-0012-1d11369404d8 identity_012                         5488
cb4d2bc2-f863-440b-0013-1d11369404d8 identity_013                         5488
cb4d2bc2-f863-440b-0014-1d11369404d8 identity_014                         5488
cb4d2bc2-f863-440b-0015-1d11369404d8 identity_015                         5488
cb4d2bc2-f863-440b-0016-1d11369404d8 identity_016                         5488
cb4d2bc2-f863-440b-0017-1d11369404d8 identity_017                         5488
cb4d2bc2-f863-440b-0018-1d11369404d8 identity_018                         5488
cb4d2bc2-f863-440b-0019-1d11369404d8 identity_019                         5488
cb4d2bc2-f863-440b-0020-1d11369404d8 identity_020                         5488
```

2. Copy the Identity file to the CLI container.

```
kubectl cp ~/Downloads/sample-identity.p12 cm-deploy-0-7d69cb94f7-g4gj9:/usr/local/bin/
```

3. Use the CLI command to update the identity:

```
[root@cm-deploy-0-7d69cb94f7-g4gj9 builder]# clustermanager ls identities
clusterId [6bb97326-007d-4ba3-828e-24bb81449b94] and zoneId [5d75a537-a6b4-489c-8d5a-d97469a7390a]
```

```
Using cluster name [Raj Cluster]
Using Zone name [us-east-1e]
UUID                                   Name                                   Area ID   Expiration
------------------------------------ ------------------------------------ ---------- --------------------
cb4d2bc2-f863-440b-0001-1d11369404d8   identity_001                                 5488
cb4d2bc2-f863-440b-0002-1d11369404d8   identity_002                                 5488
cb4d2bc2-f863-440b-0003-1d11369404d8   identity_003                                 5488
cb4d2bc2-f863-440b-0004-1d11369404d8   identity_004                                 5488
cb4d2bc2-f863-440b-0005-1d11369404d8   identity_005                                 5488
cb4d2bc2-f863-440b-0006-1d11369404d8   identity_006                                 5488
cb4d2bc2-f863-440b-0007-1d11369404d8   identity_007                                 5488
cb4d2bc2-f863-440b-0008-1d11369404d8   identity_008                                 5488
cb4d2bc2-f863-440b-0009-1d11369404d8   identity_009                                 5488
cb4d2bc2-f863-440b-0010-1d11369404d8   identity_010                                 5488
cb4d2bc2-f863-440b-0011-1d11369404d8   identity_011                                 5488
cb4d2bc2-f863-440b-0012-1d11369404d8   identity_012                                 5488
cb4d2bc2-f863-440b-0013-1d11369404d8   identity_013                                 5488
cb4d2bc2-f863-440b-0014-1d11369404d8   identity_014                                 5488
cb4d2bc2-f863-440b-0015-1d11369404d8   identity_015                                 5488
cb4d2bc2-f863-440b-0016-1d11369404d8   identity_016                                 5488
cb4d2bc2-f863-440b-0017-1d11369404d8   identity_017                                 5488
cb4d2bc2-f863-440b-0018-1d11369404d8   identity_018                                 5488
cb4d2bc2-f863-440b-0019-1d11369404d8   identity_019                                 5488
cb4d2bc2-f863-440b-0020-1d11369404d8   identity_020                                 5488
```

To create a new identity, see Create Identity.

To update an existing identity, see Update Identity.

Cluster Manager CLI Commands

- Create Certificate
- Create Identity
- Update Certificate
- Update Identity
- List Certificates
- List Identities

Create Certificate

Example:

```
[root@01294de80b3d builder]# clustermanager create certificate help
Error: required flag(s) "file" not set
Usage:
clustermanager create certificate [flags]

Aliases:
certificate, cert

Flags:
--file string certificate file to create, CER type
-h, --help help for certificate

required flag(s) "file" not set

[root@01294de80b3d builder]# clustermanager create cert --file sample.cer
Creating a certificate for the topology
Successfully created certificate for the topology
```

Create Identity

Example:

```
[root@01294de80b3d builder]# clustermanager create identity help
Error: required flag(s) "file", "password" not set
Usage:
  clustermanager create identity [flags]

Aliases:
  identity, idty

Flags:
      --file string       identity file to create, P12 / JKS type
  -h, --help              help for identity
      --password string   password for the identity file

required flag(s) "file", "password" not set


[root@01294de80b3d builder]# clustermanager create identity --file sample.p12 --password
password
```

```
Creating an identity for the topology
Successfully created identity for the topology
```

Update Certificate

Example:

```
[root@01294de80b3d builder]# clustermanager set certificate help
Error: required flag(s) "file", "uuid" not set
Usage:
  clustermanager set certificate [flags]

Aliases:
  certificate, cert

Flags:
      --file string       certificate file to upload, CER type
  -h, --help              help for certificate
      --uuid string       unique ID for the existing certificate that needs update

required flag(s) "file", "uuid" not set


[root@01294de80b3d builder]# clustermanager set cert --file sample.cer --uuid f26a2422-b2cd-
4905-bbc7-5706d450b137
Updating certificate for the topology
Successfully updated certificate for the topology
```

Update Identity

Example:

```
[root@01294de80b3d builder]# clustermanager set identity help
Error: required flag(s) "file", "password", "uuid" not set
Usage:
  clustermanager set identity [flags]

Aliases:
  identity, idty

Flags:
      --file string       identity file to upload, P12 / JKS type
  -h, --help              help for identity
```

```
    --password string    password for the identity file
    --uuid string        unique ID for the existing identity that needs update
    --verifyHostname     required for mutual SSL. If set to true, hostname inside client and server
identity & trust files should be same. 'verifyHostname=false' if the flag is empty

required flag(s) "file", "password", "uuid" not set


[root@01294de80b3d builder]# clustermanager set idty --file sample.p12 --password password --
uuid a434bb96-be67-4676-a8aa-9c9fe6503d76
Updating an identity for the topology
Successfully updated identity for the topology
```

List Certificates

Example:

```
[root@01294de80b3d builder]# clustermanager list certificates --help
List of trust certificates available for the topology

Usage:
  clustermanager list certificates [flags]

Flags:
  -h, --help             help for certificates

[root@01294de80b3d builder]# clustermanager ls certificates
UUID                                 Name                                 Area ID    Expiration
------------------------------------ ------------------------------------ ---------- ----------------------
f26a2422-b2cd-4905-bbc7-5706d450b137    sample                            0          2029-03-22
```

List Identities

Example

```
[root@01294de80b3d builder]# clustermanager list identities --help
List of identities available for the topology

Usage:
  clustermanager list identities [flags]

Flags:
  -h, --help             help for identities
```

```
[root@01294de80b3d builder]# clustermanager list identities
Using Zone name [local]
UUID                                   Name                                 Area ID    Expiration
-------------------------------------- ------------------------------------ ---------- --------------------
a434bb96-be67-4676-a8aa-9c9fe6503d76   sample                               5488       2020-03-26
```

Securing Using OAuth

Endpoints can be configured as protected or unprotected i.e without any authentication. Boomi Cloud™ API Management - Local Edition provides authentication using the OAuth method.

It is advised to check the recommendations for OAuth 5.x

OAuth APIs and Authenticator Service

The OAuth Authenticator introduced in Local Edition 5.0 includes the Basic Authenticator for an OAuth API endpoint, and a Public Key Authenticator configured and targeted for an OAuth service endpoint. This page describes the configuration of the OAuth Authenticator and also provides sample calls to OAuth APIs.

OAuth Authenticator Service Configuration

Authorization for the OAuth Authenticator Service depends on the configuration of "com.mashery.service.onprem.oauth.authenticator.oauth-service-authenticator" in tml_tm_properties.json.

Note: For more information on the tml_tm_properties.json file, refer to the *tml-tm* section in the *Configuring Properties Common to All Deployments* topic.

OAuth API requests can be authenticated using Basic Authentication and/or Public Key Authentication:

- If the OAuth API endpoint is exposed or used directly, Basic Authentication should be configured.
- If a proxy OAuth API service endpoint is used, Public Key Authentication should be configured.

The following is an example of the configuration in tml_tm_properties.json for both Basic and Public Key Authentication:

```
"com.mashery.service.onprem.oauth.authenticator.oauth-service-authenticator" : {
      "publicKeyName": "public_key",
      "publicKeyValue": "3bvu8u3p8l",
      "username": "root",
      "password": "057ba03d6c44104863dc7361fe4578965d1887360f90a0895882e58a6248fc86"
},
```

- The username/password pair is to support Basic Authentication. The password can be generated by any SHA256 Hash generator. "057ba03d6c44104863dc7361fe4578965d1887360f90a0895882e58a6248fc86" is a SHA256 for "changeme".

  The encoded string "cm9vdDpjaGFuZ2VtZQ==" in the curl header -H 'Authorization: Basic cm9vdDpjaGFuZ2VtZQ==' is for "root:changeme".

- The publicKeyName/publicKeyValue pair is for public key authentication when setting up an OAuth API service endpoint.

  The public key is extracted from the request parameters when the request is received by the OAuth API service endpoint created in either tethered or untethered mode (refer to sample OAuth API service endpoint setup below for more details).

  To use the endpoint, the publicKeyName and publicKeyValue properties must be configured in tml_tm_properties.json and their values should match their corresponding parameter string in the endpoint.

  For example, if publicKeyName/publicKeyValue is set to "public_key/3bvu3p8l ", its parameter string should be "public_key=3bvu8u3p8l".

- Basic Authenticator is disabled if the username property of service "com.mashery.service.onprem.oauth.authenticator.oauth-service-authenticator" is not present in tml_tm_properties.json.

  The request must pass public key authentication when Basic Authentication is disabled.

- Public key authentication is disabled if the publicKeyName property of service "com.mashery.service.onprem.oauth.authenticator.oauth-service-authenticator" is not present in tml_tm_properties.json.

  If public key authentication is disabled, Basic Authentication should be enabled by

configuring the username and password.

Different Ways to Make OAuth API Calls

Call via Internal OAuth API Endpoint Basic Authentication

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":
{"service_key":"m9f5kyzfjjrssz6d5sfkbuw3","client":
{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_secret":"Azn4Xa"},"token_data":
{"grant_type":"implicit"},"uri":
{"redirect_uri":"https://some.com/cb"},"user_context":"testUser1455730448397"},"jsonrpc":"2.0"}'
'https://api.mashery.com:8083/v2/json-rpc' -u root:changeme -k
```

This call uses the username/password pair in tml_tm_properties.json and is for API Management - Local Edition 4.x backward compatibility.

Call via Proxied OAuth API Endpoint with Public Key Authentication

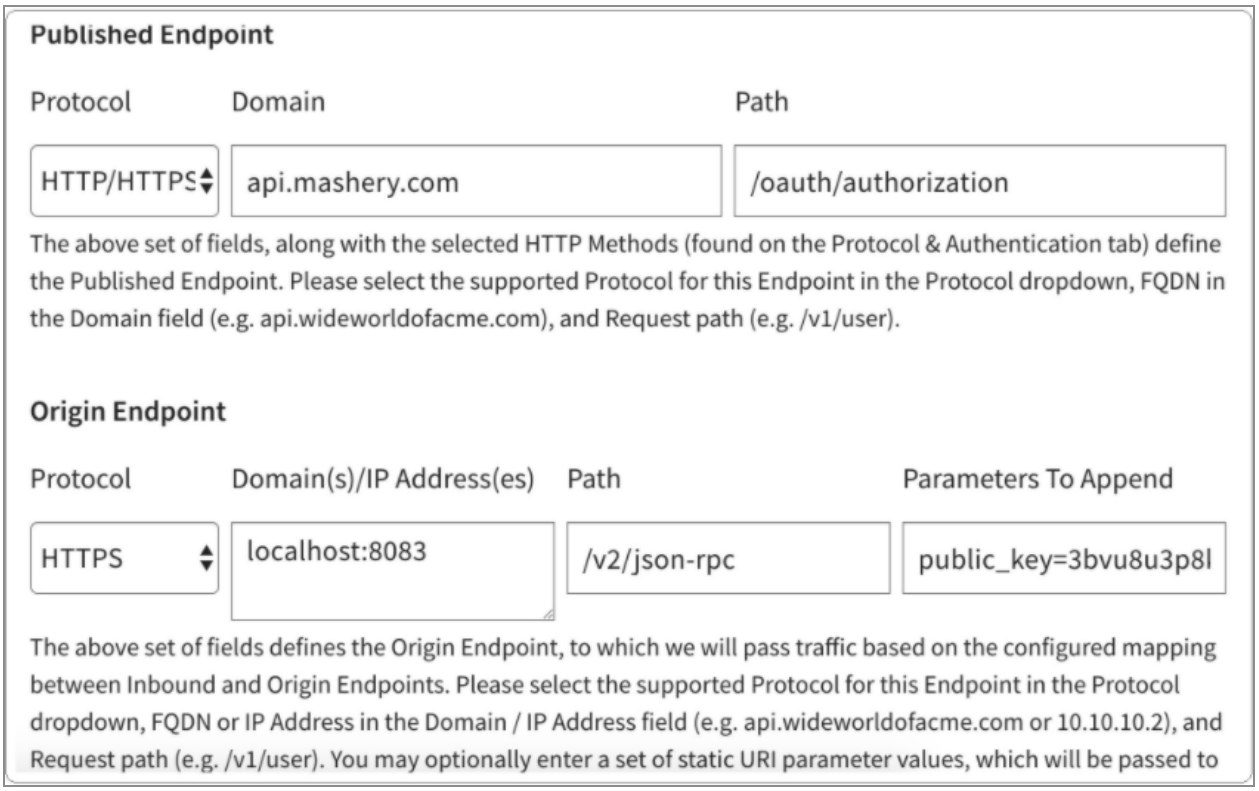
```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":
{"service_key":"m9f5kyzfjjrssz6d5sfkbuw3","client":
{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_secret":"Azn4Xa"},"token_data":
{"grant_type":"implicit"},"uri":
{"redirect_uri":"https://some.com/cb"},"user_context":"testUser1455730448397"},"jsonrpc":"2.0"}' -
H 'Content-Type: application/json' 'http://api.mashery.com/oauth/authorization/?api_
key=8qavf83pg2uzgndtxz2w4dg3'
```

This call is new in Local Edition 5.x. It uses the URL path /oauth/authorization and publicKeyName/publicKeyValue pair in tml_tm_properties.json, along with the URL parameter string setting in the proxy endpoint.

Users do not need to specify any credentials (username/password) in the curl commands.

Sample Proxy Endpoint Setup with publicKeyName/publicKeyValue Parameter String in Tethered/Untethered

The following screenshot shows the settings on "Call Transformation" of the proxied OAuth API endpoint.



In this setup:

- The service URL is set to "/oauth/authorization" and can either be sent with HTTP or HTTPS.
- The parameters include the *publicKeyName/publicKeyValu*e set in "tml_tm_ properties.json"
- The protocol and IP address of the Origin Endpoint correspond to the internal OAuth API endpoint setting in tml_tm_properties.json.

Support for HTTPS Connection

This setting is in tml_tm_properties.json to support an HTTPS connection for OAuth.

```
"com.mashery.service.onprem.oauth.api-server": {
    "http.enabled": true,
    "https.enabled": true,
    "http.port": 9083,
    "https.port": 8083,
    "ssl.keystore": "/etc/mashery-server-ssl/tml-tm.jks",
    "ssl.password": "changeit",
    "ssl.keypassword": "changeit"
```

```
        "ssl.keypassword": "changeit"
},
```

In Local Edition 5.x, only https.port is configured in the .yml file and exposed when the container is started. The http.port works only in calls within the containers.

You should always use HTTPS configuration for OAuth API calls outside the container.

In the "properties" folder in Local Edition deployment scripts, "tml-tm.jks" should be replaced with customers' keystore. This keystore is used by both regular traffic (non-OAuth requests) and OAuth API.

After "tml-tm.jks" is replaced with customers' keystore, the setting "tm_keystore_password" in "properties/tml_tm_properties.json" should also be updated.

OAuth APIs

Sample calls through proxy endpoint using above settings.

Create access token

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"implicit"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context":"testUser1455730448397"},"jsonrpc":"2.0"}' -H 'Host:
api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 310
>
* upload completely sent off: 310 out of 310 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 231
< Date: Tue, 30 Oct 2018 00:38:26 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
```

```
token":"unrct8wsweny8h3b3kw39wab","expires_in":36000,"scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Create authorization code

```
curl -v -d '{ "id":1,"method":"oauth2.createAuthorizationCode","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x"},"response_
type":"code","uri":{"redirect_uri":"https://some.com/cb"},"user_context":
"testUser1455730448397"},"jsonrpc":"2.0"}' -H 'Host: api.mashery.com' -H 'Content-Type:
application/json' 'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 277
>
* upload completely sent off: 277 out of 277 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 168
< Date: Tue, 30 Oct 2018 00:42:12 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"code":"p2kjsmn8mw2xb3d2ksryxk6z","uri":{"redirect_
uri":"https://some.com/cb?code=p2kjsmn8mw2xb3d2ksryxk6z","state":""}},"error":null}
```

Create access token with authorization code

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"authorization_
code","code":"p2kjsmn8mw2xb3d2ksryxk6z","scope":"scope1"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context": "testUser1455730448397"},"jsonrpc":"2.0"}' -H
'Content-Type: application/json'  -H 'Host: api.mashery.com'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
```

```
> Accept: */*
> Content-Type: application/json
> Content-Length: 372
>
* upload completely sent off: 372 out of 372 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Date: Tue, 30 Oct 2018 00:49:18 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"rjzb2k26cgv7tufe2j8cs7rm","expires_in":36000,"refresh_
token":"zgwyg64s2ejc6catffvg5kbx","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Update access token

```
curl -v -d '{"id":1,"method":"oauth2.updateAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}, "access_token":"rjzb2k26cgv7tufe2j8cs7rm","user_
context":"testUser1455730448397", "expires_in":600},"jsonrpc":"2.0"}' -H 'Host: api.mashery.com'  -
H 'Content-Type: application/json'  'http://X.X.99.100:80/oauth/authorization/?api_
key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 287
>
* upload completely sent off: 287 out of 287 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 322
< Date: Tue, 30 Oct 2018 01:01:50 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","access_
token":"rjzb2k26cgv7tufe2j8cs7rm","expires":"2018-10-30T11:01:50Z","refresh_token_
```

```
expires":1540864158505,"scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}
```

Fetch access token

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method": "oauth2.fetchAccessToken","params": {"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","access_token": "rjzb2k26cgv7tufe2j8cs7rm"}}' -H 'Host:
api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 157
>
* upload completely sent off: 157 out of 157 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 242
< Date: Tue, 30 Oct 2018 01:05:25 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","expires":"2018-10-
30T11:01:50Z","scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}}
```

Fetch application

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method":"oauth2.fetchApplication","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}}}' -H 'Host: api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 187
```

```
>
* upload completely sent off: 187 out of 187 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 82
< Date: Tue, 30 Oct 2018 01:10:58 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"id":251547,"name":"TestOAuthRegisterCallBack"}}
```

Fetch user application

```
curl -v -d  '{ "id":1, "method":"oauth2.fetchUserApplications", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3",  "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}' -H
'Host: api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 162
>
* upload completely sent off: 162 out of 162 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 195
< Date: Tue, 30 Oct 2018 01:12:32 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":[{"id":251547,"name":"TestOAuthRegisterCallBack","access_tokens":
["rjzb2k26cgv7tufe2j8cs7rm","unrct8wsweny8h3b3kw39wab"],"client_
id":"yq7chp6ufkeea587gp6wqa6x"}]}
```

Revoke access token

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeAccessToken", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "access_token":"rjzb2k26cgv7tufe2j8cs7rm" },"jsonrpc":"2.0"}' -H 'Host:
```

```
api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 240
>
* upload completely sent off: 240 out of 240 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 38
< Date: Tue, 30 Oct 2018 01:17:10 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":true}
```

Revoke user application

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeUserApplication", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}' -H 'Host:
api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 241
>
* upload completely sent off: 241 out of 241 bytes
< HTTP/1.1 500 Server Error
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 81
< Date: Tue, 30 Oct 2018 16:41:56 GMT
<
```

```
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"error":{"message":"Internal Server Error","code":-2001}}
```

Refresh token

```
curl -k -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"refresh_token","refresh_
token":"8f9qy6jrw3r23j2r6wqwp2jh"},"uri":{"redirect_uri":"https://some.com/cb"},"user_
context":null},"jsonrpc":"2.0"}'  -H 'Host: api.mashery.com' -H 'Content-Type: application/json'
'http://X.X.99.100:80/oauth/authorization/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying X.X.99.100...
* Connected to X.X.99.100 (X.X.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 339
>
* upload completely sent off: 339 out of 339 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Date: Tue, 30 Oct 2018 16:55:36 GMT
<
* Connection #0 to host X.X.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"79rkkkxd4rk2zzbrqj4z93vb","expires_in":36000,"refresh_
token":"7db67za3xvdg99nc7gnctkqw","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Sample direct calls with Basic Authentication

Create access token

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"implicit"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context":"testUser1455730448397"},"jsonrpc":"2.0"}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
```

```
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 310
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 310 out of 310 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 231
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"q3nxsetjry582yhq2ej3xa7j","expires_in":36000,"scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}
```

Create authorization code

```
curl -v -d '{ "id":1,"method":"oauth2.createAuthorizationCode","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x"},"response_
type":"code","uri":{"redirect_uri":"https://some.com/cb"},"user_context":
"testUser1455730448397"},"jsonrpc":"2.0"}' 'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -
k
*   Trying X.X.57.97...
```

```
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 277
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 277 out of 277 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 168
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"code":"33benr9fb9vah3g8ys7scmh5","uri":{"redirect_
uri":"https://some.com/cb?code=33benr9fb9vah3g8ys7scmh5","state":""}},"error":null}
```

Create access token with authorization code

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"authorization_
code","code":"33benr9fb9vah3g8ys7scmh5","scope":"scope1"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context": "testUser1455730448397"},"jsonrpc":"2.0"}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
```

```
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 372
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 372 out of 372 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"s4b7qs24qfns2m3ecjmuvgh4","expires_in":36000,"refresh_
token":"xekntywdyp2hg7yvfbaxac5g","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Update access token

```
curl -v -d '{"id":1,"method":"oauth2.updateAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}, "access_token":"s4b7qs24qfns2m3ecjmuvgh4","user_
```

```
context":"testUser1455730448397", "expires_in":600},"jsonrpc":"2.0"}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 287
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 287 out of 287 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 322
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","access_
token":"s4b7qs24qfns2m3ecjmuvgh4","expires":"2018-11-21T10:57:43Z","refresh_token_
expires":1542765336403,"scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}}
```

Fetch access token

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method": "oauth2.fetchAccessToken","params": {"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","access_token": "s4b7qs24qfns2m3ecjmuvgh4"}}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 157
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 157 out of 157 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 242
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","expires":"2018-11-
21T10:57:43Z","scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}}
```

Fetch application

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method":"oauth2.fetchApplication","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}}}' 'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 187
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 187 out of 187 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 82
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"id":251547,"name":"TestOAuthRegisterCallBack"}}
```

Fetch user application

```
curl -v -d  '{ "id":1, "method":"oauth2.fetchUserApplications", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3",  "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 162
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 162 out of 162 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 195
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":[{"id":251547,"name":"TestOAuthRegisterCallBack","access_tokens":
["s4b7qs24qfns2m3ecjmuvgh4","q3nxsetjry582yhq2ej3xa7j"],"client_
id":"yq7chp6ufkeea587gp6wqa6x"}]}
```

Revoke access token

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeAccessToken", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "access_token":"s4b7qs24qfns2m3ecjmuvgh4" },"jsonrpc":"2.0"}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 240
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 240 out of 240 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 38
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":true}
```

Revoke user application

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeUserApplication", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}'
'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 241
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 241 out of 241 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 38
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":true}
```

Refresh token

```
curl -k -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"refresh_token","refresh_
token":"xekntywdyp2hg7yvfbaxac5g"},"uri":{"redirect_uri":"https://some.com/cb"},"user_
context":null},"jsonrpc":"2.0"}' 'https://X.X.57.97:8083/v2/json-rpc' -u root:changeme -k
*   Trying X.X.57.97...
* Connected to X.X.57.97 (X.X.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'X.X.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v2/json-rpc HTTP/1.1
> Host: X.X.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 339
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 339 out of 339 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host X.X.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"z242k8wyshjmvjrkcckask2f","expires_in":36000,"refresh_
```

```
token":"epecdr7zzfjxwwfu35395dx9","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Making OAuth 2.0 Calls

OAuth API Supported Methods

The following methods support the OAuth 2.0 functionality:

- fetchApplication - Verifies that client has a valid identifier and fetches the application data (e.g., name) for a client requesting access to user resource.
- createAuthorizationCode - Creates authorization code that can be subsequently used to obtain an access token. Used in Authorization Code flow.
- createAccessToken - Creates access token for the flow indicated by response and grant type. Individual parameters are validated based on the indicated flow.
- fetchAccessToken - Fetches the specified access token and data associated with the token or returns that it is invalid.
- fetchUserApplications - Fetches the applications with access tokens for the given user.
- revokeAccessToken - Revokes the access token.
- revokeUserApplication - Revokes all tokens for the application for the specified user.
- updateAccessToken - Updates access token based on individual parameters specified.

Sample Calls

Sample Direct Calls using HTTPS and Basic Authentication

Create access token

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
```

```
secret":"Azn4Xa"},"token_data":{"grant_type":"implicit"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context":"testUser1455730448397"},"jsonrpc":"2.0"}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 310
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 310 out of 310 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 231
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"q3nxsetjry582yhq2ej3xa7j","expires_in":36000,"scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}
```

Create authorization code

```
curl -v -d '{ "id":1,"method":"oauth2.createAuthorizationCode","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x"},"response_
type":"code","uri":{"redirect_uri":"https://some.com/cb"},"user_context":
"testUser1455730448397"},"jsonrpc":"2.0"}' 'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 277
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 277 out of 277 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 168
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"code":"33benr9fb9vah3g8ys7scmh5","uri":{"redirect_
uri":"https://some.com/cb?code=33benr9fb9vah3g8ys7scmh5","state":""}},"error":null}
```

Create access token with authorization code

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"authorization_
code","code":"33benr9fb9vah3g8ys7scmh5","scope":"scope1"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context": "testUser1455730448397"},"jsonrpc":"2.0"}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 372
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 372 out of 372 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"s4b7qs24qfns2m3ecjmuvgh4","expires_in":36000,"refresh_
```

```
token":"xekntywdyp2hg7yvfbaxac5g","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Update access token

```
curl -v -d '{"id":1,"method":"oauth2.updateAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}, "access_token":"s4b7qs24qfns2m3ecjmuvgh4","user_
context":"testUser1455730448397", "expires_in":600},"jsonrpc":"2.0"}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 287
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 287 out of 287 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 322
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
```

{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","access_
token":"s4b7qs24qfns2m3ecjmuvgh4","expires":"2018-11-21T10:57:43Z","refresh_token_
expires":1542765336403,"scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}}

Fetch access token

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method": "oauth2.fetchAccessToken","params": {"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","access_token": "s4b7qs24qfns2m3ecjmuvgh4"}}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 157
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 157 out of 157 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 242
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
```

{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","expires":"2018-11-21T10:57:43Z","scope":null,"user_context":"testUser1455730448397","uri":null,"grant_type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}}

Fetch application

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method":"oauth2.fetchApplication","params":{"service_key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_secret":"Azn4Xa"}}}' 'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 187
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 187 out of 187 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 82
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"id":251547,"name":"TestOAuthRegisterCallBack"}}
```

Fetch user application

```
curl -v -d  '{ "id":1, "method":"oauth2.fetchUserApplications", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3",  "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 162
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 162 out of 162 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 195
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":[{"id":251547,"name":"TestOAuthRegisterCallBack","access_tokens":
["s4b7qs24qfns2m3ecjmuvgh4","q3nxsetjry582yhq2ej3xa7j"],"client_
id":"yq7chp6ufkeea587gp6wqa6x"}]}
```

Revoke access token

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeAccessToken", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "access_token":"s4b7qs24qfns2m3ecjmuvgh4" },"jsonrpc":"2.0"}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 240
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 240 out of 240 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 38
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":true}
```

Revoke user application

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeUserApplication", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}'
'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 241
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 241 out of 241 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 38
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":true}
```

Refresh token

```
curl -k -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"refresh_token","refresh_
token":"xekntywdyp2hg7yvfbaxac5g"},"uri":{"redirect_uri":"https://some.com/cb"},"user_
context":null},"jsonrpc":"2.0"}' 'https://x.x.57.97:8083/v3/json-rpc' -u root:changeme -k
*   Trying x.x.57.97...
* Connected to x.x.57.97 (x.x.57.97) port 8083 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*       server certificate verification SKIPPED
*       server certificate status verification SKIPPED
*       common name: api.example.com (does not match 'x.x.57.97')
*       server certificate expiration date OK
*       server certificate activation date OK
*       certificate public key: RSA
*       certificate version: #3
*       subject: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       start date: Wed, 14 Nov 2018 03:28:23 GMT
*       expire date: Fri, 21 Oct 2118 03:28:23 GMT
*       issuer: C=defC,ST=defST,L=defL,O=defO,OU=defOU,CN=api.example.com
*       compression: NULL
* ALPN, server did not agree to a protocol
* Server auth using Basic with user 'root'
> POST /v3/json-rpc HTTP/1.1
> Host: x.x.57.97:8083
> Authorization: Basic cm9vdDpjaGFuZ2VtZQ==
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Length: 339
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 339 out of 339 bytes
< HTTP/1.1 200 OK
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Server: Jetty(8.1.3.v20120522)
<
* Connection #0 to host x.x.57.97 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"z242k8wyshjmvjrkcckask2f","expires_in":36000,"refresh_
```

token":"epecdr7zzfjxwwfu35395dx9","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}

Sample Calls through Proxy Endpoint

Note: This section uses the settings from the *Proxy Endpoint Creation in Tethered* section in OAuth Authenticator Service Configuration.

Create access token

```
curl -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"implicit"},"uri":{"redirect_
uri":"https://some.com/cb"},"user_context":"testUser1455730448397"},"jsonrpc":"2.0"}' -H 'Host:
chainsproxy.api.mashery.com' -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 310
>
* upload completely sent off: 310 out of 310 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 231
< Date: Tue, 30 Oct 2018 00:38:26 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"unrct8wsweny8h3b3kw39wab","expires_in":36000,"scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Create authorization code

```
curl -v -d '{ "id":1,"method":"oauth2.createAuthorizationCode","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x"},"response_
type":"code","uri":{"redirect_uri":"https://some.com/cb"},"user_context":
```

```
"testUser1455730448397"},"jsonrpc":"2.0"}' -H 'Host: chainsproxy.api.mashery.com' -H 'Content-
Type: application/json' 'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 277
>
* upload completely sent off: 277 out of 277 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 168
< Date: Tue, 30 Oct 2018 00:42:12 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"code":"p2kjsmn8mw2xb3d2ksryxk6z","uri":{"redirect_
uri":"https://some.com/cb?code=p2kjsmn8mw2xb3d2ksryxk6z","state":""}},"error":null}
```

Create access token with authorization code

```
curl -v -d '{ "id":1,"method":"oauth2.createAuthorizationCode","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x"},"response_
type":"code","uri":{"redirect_uri":"https://some.com/cb"},"user_context":
"testUser1455730448397"},"jsonrpc":"2.0"}' -H 'Host: chainsproxy.api.mashery.com' -H 'Content-
Type: application/json' 'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 277
>
* upload completely sent off: 277 out of 277 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 168
< Date: Tue, 30 Oct 2018 00:42:12 GMT
```

```
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"code":"p2kjsmn8mw2xb3d2ksryxk6z","uri":{"redirect_
uri":"https://some.com/cb?code=p2kjsmn8mw2xb3d2ksryxk6z","state":""}},"error":null}
```

Update access token

```
curl -v -d '{"id":1,"method":"oauth2.updateAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}, "access_token":"rjzb2k26cgv7tufe2j8cs7rm","user_
context":"testUser1455730448397", "expires_in":600},"jsonrpc":"2.0"}' -H 'Host:
chainsproxy.api.mashery.com'  -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 287
>
* upload completely sent off: 287 out of 287 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 322
< Date: Tue, 30 Oct 2018 01:01:50 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","access_
token":"rjzb2k26cgv7tufe2j8cs7rm","expires":"2018-10-30T11:01:50Z","refresh_token_
expires":1540864158505,"scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}
```

Fetch access token

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method": "oauth2.fetchAccessToken","params": {"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","access_token": "rjzb2k26cgv7tufe2j8cs7rm"}}' -H 'Host:
chainsproxy.api.mashery.com' -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
```

```
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 157
>
* upload completely sent off: 157 out of 157 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 242
< Date: Tue, 30 Oct 2018 01:05:25 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","expires":"2018-10-
30T11:01:50Z","scope":null,"user_context":"testUser1455730448397","uri":null,"grant_
type":"authorization_code","client_id":"yq7chp6ufkeea587gp6wqa6x","extended":null}}
```

Fetch application

```
curl -v -d '{"id": 1,"jsonrpc":"2.0","method":"oauth2.fetchApplication","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"}}}' -H 'Host: chainsproxy.api.mashery.com' -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 187
>
* upload completely sent off: 187 out of 187 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 82
< Date: Tue, 30 Oct 2018 01:10:58 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"id":251547,"name":"TestOAuthRegisterCallBack"}}
```

Fetch user application

```
curl -v -d  '{ "id":1, "method":"oauth2.fetchUserApplications", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3",  "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}' -H
'Host: chainsproxy.api.mashery.com' -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 162
>
* upload completely sent off: 162 out of 162 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 195
< Date: Tue, 30 Oct 2018 01:12:32 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":[{"id":251547,"name":"TestOAuthRegisterCallBack","access_tokens":
["rjzb2k26cgv7tufe2j8cs7rm","unrct8wsweny8h3b3kw39wab"],"client_
id":"yq7chp6ufkeea587gp6wqa6x"}]}
```

Revoke access token

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeAccessToken", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "access_token":"rjzb2k26cgv7tufe2j8cs7rm" },"jsonrpc":"2.0"}' -H 'Host:
chainsproxy.api.mashery.com' -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 240
>
* upload completely sent off: 240 out of 240 bytes
< HTTP/1.1 200 OK
```

```
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 38
< Date: Tue, 30 Oct 2018 01:17:10 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":true}
```

Revoke user application

```
curl -v -d  '{ "id":1, "method":"oauth2.revokeUserApplication", "params":{ "service_
key":"m9f5kyzfjjrssz6d5sfkbuw3", "client": { "client_id":"yq7chp6ufkeea587gp6wqa6x", "client_
secret":"Azn4Xa" }, "user_context":"testUser1455730448397" },"jsonrpc":"2.0"}' -H 'Host:
chainsproxy.api.mashery.com' -H 'Content-Type: application/json'
'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 241
>
* upload completely sent off: 241 out of 241 bytes
< HTTP/1.1 500 Server Error
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 81
< Date: Tue, 30 Oct 2018 16:41:56 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"error":{"message":"Internal Server Error","code":-2001}}
```

Refresh token

```
curl -k -v -d '{"id":1,"method":"oauth2.createAccessToken","params":{"service_
key":"m9f5kyzfjjrssz6d5sfkbuw3","client":{"client_id":"yq7chp6ufkeea587gp6wqa6x","client_
secret":"Azn4Xa"},"token_data":{"grant_type":"refresh_token","refresh_
token":"8f9qy6jrw3r23j2r6wqwp2jh"},"uri":{"redirect_uri":"https://some.com/cb"},"user_
context":null},"jsonrpc":"2.0"}'  -H 'Host: chainsproxy.api.mashery.com' -H 'Content-Type:
application/json' 'http://192.168.99.100:80/v2/oauth/?api_key=8qavf83pg2uzgndtxz2w4dg3'
*   Trying 192.168.99.100...
```

```
* Connected to 192.168.99.100 (192.168.99.100) port 80 (#0)
> POST /oauth/auth/v2/?api_key=8qavf83pg2uzgndtxz2w4dg3 HTTP/1.1
> Host: chainsproxy.api.mashery.com
> User-Agent: curl/7.43.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 339
>
* upload completely sent off: 339 out of 339 bytes
< HTTP/1.1 200 OK
< X-Mashery-Responder: 8abfe284898c
< Content-Type: application/json;charset=UTF-8
< Cache-Control: no-store
< Content-Length: 274
< Date: Tue, 30 Oct 2018 16:55:36 GMT
<
* Connection #0 to host 192.168.99.100 left intact
{"jsonrpc":"2.0","id":1,"result":{"token_type":"bearer","return_type":"json","access_
token":"79rkkkxd4rk2zzbrqj4z93vb","expires_in":36000,"refresh_
token":"7db67za3xvdg99nc7gnctkqw","scope":null,"user_
context":"testUser1455730448397","uri":null,"extended":null,"state":null}}
```

Securing Config UI and Dev Portal in Untethered Setup

The Config UI and Dev Portal are UI components of Boomi Cloud™ API Management - Local Edition. They can be secured using LDAP and SAML. This section describes how to setup LDAP and SAML server and configure it with Local Edition. Before moving to Local Edition, we have to setup SAML and LDAP server on an instance.

Configuring LDAP and SAML with Local Edition

The following section provides details to configure LDAP and SAML with Local Edition.

Configuring LDAP with Local Edition

Procedure

1. Launch Configuration Manager.

   ```
   https://<ip_of_cm_service>:8443/admin
   ```

2. Navigate to **Members**.
3. Create a new user name LDAP
4. Add the appropriate role to the LDAP user by navigating to **Portal Access Group** tab. The LDAP-Authenticated users inherit the roles and permissions of this LDAP user.
5. Navigate to **Home > Zone Settings** and under **LDAP settings** input the following:
   - **LDAP Enabled**: Checked (Enabled)
   - **LDAP Login Button Text**: LDAP Login(or any other text you want on the **LDAP Login** button)
   - **LDAP URL**: URL of the LDAP Server. This server should be accessible from the Local Edition Cluster. For example,

     ```
     LDAP URL: ldap:// 34. 67.34.161.1234
     ```

   - **Bind Username and Password**: This is used to connect to the LDAP Server. For example,

     ```
     Bind Username: uid=john,ou=People,dc=nodomain
     Bind Password: johnldap
     ```

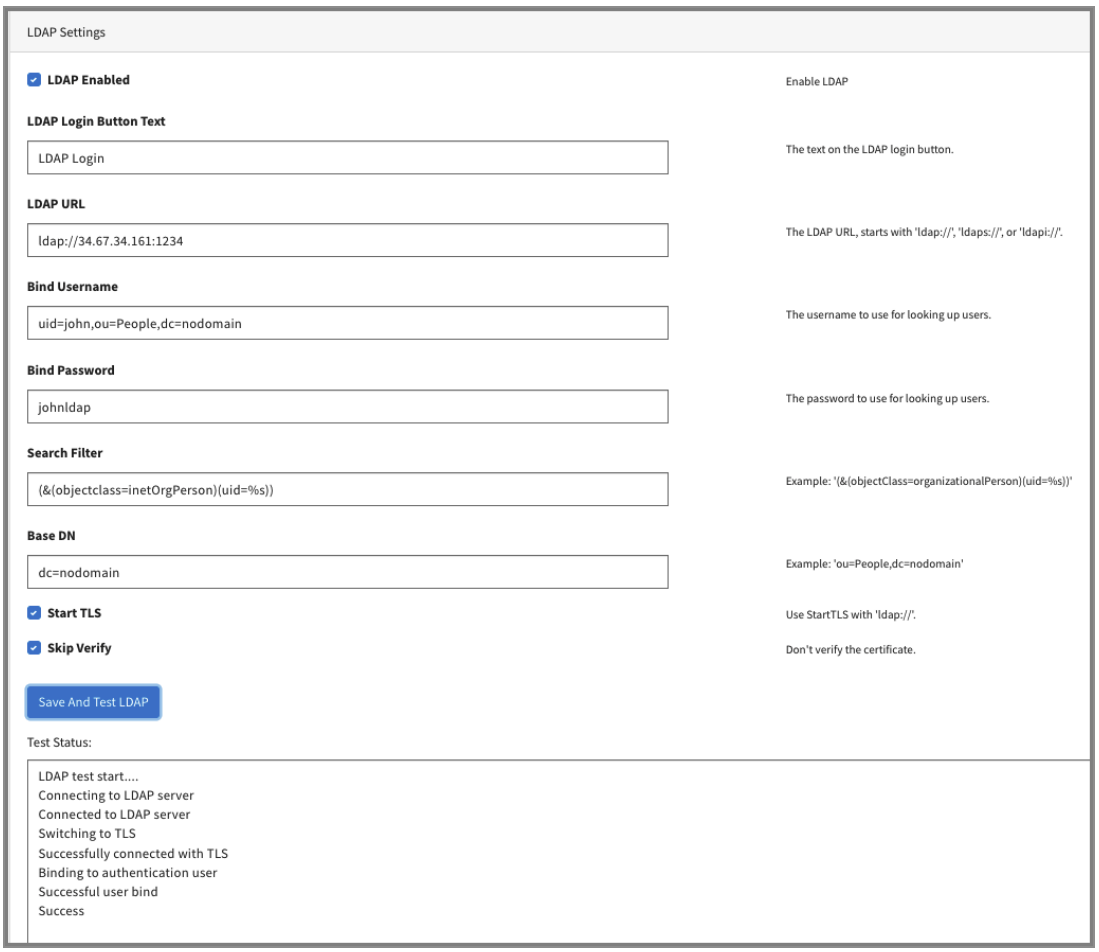
   - **Search Filter**: Filter to search for the user during authentication. Standard LDAP Filter syntax is supported. For example,

     ```
     Search Filter: (&(objectclass=inetOrgPerson)(uid=%s))
     ```

     uid can be replaced with any unique identifier. %s represents the username used on the login page. A minimal filter should contain **(<unique_ identifier>=%s)**
   - **Base DN**: The Base DN is the starting point a LDAP Server uses when

searching for users authentication within the directory.
Base DN: dc=example-domain,dc=com

- **Start TLS**: Checked (Enabled)
- **Skip Verify**: Checked (Enabled)



6. Click **Save and Test LDAP**.

   On successful configuration the test status must be :

```
LDAP test start....
Connecting to LDAP server
Connected to LDAP server
Switching to TLS
Successfully connected with TLS
Binding to authentication user
```

```
Successful user bind
Success
```

Login with LDAP User

The following section provides details to login with a LDAP user.

1. Navigate to the Login page, https://<ip_of_cm_service>:8443/admin
2. Provide the LDAP Username and password and click the **Sign In** button.

On successful authentication, you should be signed-in in the Configuration Manager.

Configuring SAML with Local Edition

Procedure

1. Generate SAML certificate and key on SAML Server. Transfer this key and certificate on to local machine.

   ```
   openssl req -x509 -newkey rsa:2048 -keyout myservice.key -out myservice.cert -days 365 -nodes -subj "/CN=myservice.example.com
   ```

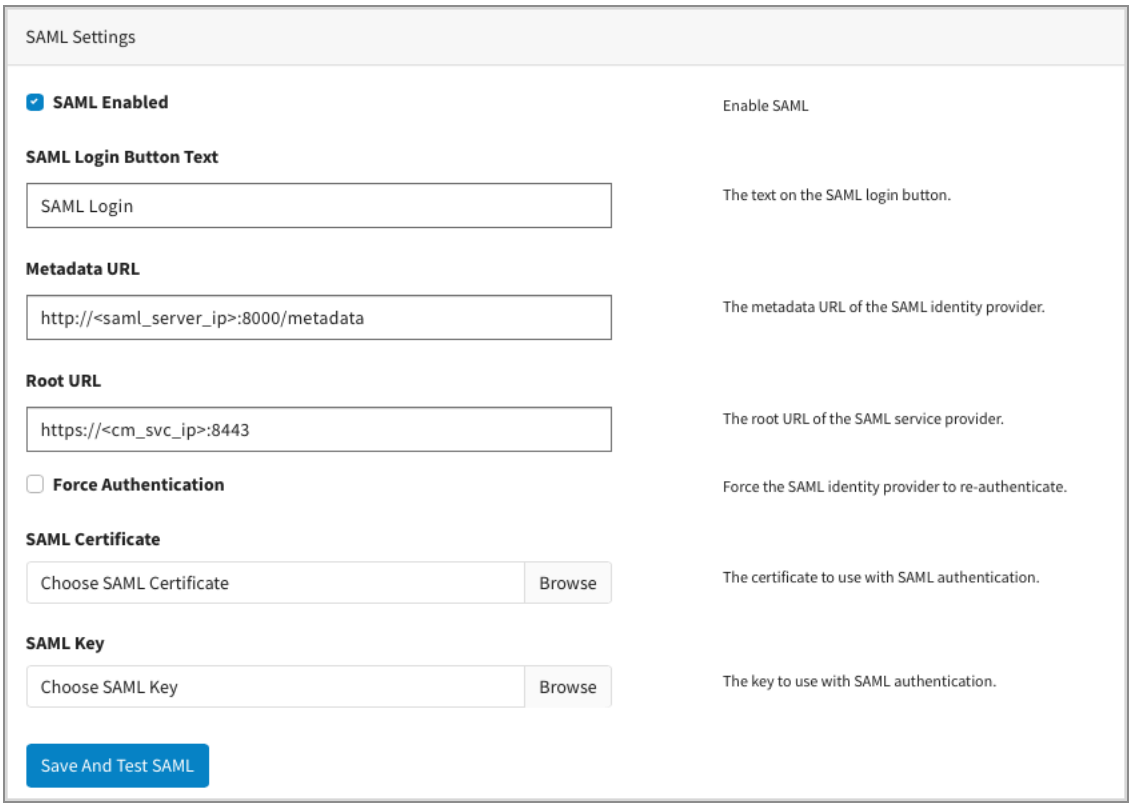
2. Launch the configuration manager and navigate to **Members**.

   ```
   https://<ip_of_cm_service>:8443/admin
   ```

3. Create a new user as SAML and in the **Portal Access Group** tab, input the Administrator role to the saml user.
4. Click **Home > Zone Settings > SAML settings** and input the following:
   - SAML Enabled: Checked
   - SAML Login Button Text: SAML Login. This can be any other text you want on the SAML Login Button.
   - Metadata URL: http://<saml_server_ip>:8000/metadata. This is the IP of VM

or instance where SAML server is running.

- Root URL: https://<cm_svc_ip>:8443. Here, cm_svc_ip is the IP of cm service that is used to launch the configuration manager.
- Force Authentication: Checked
- SAML Certificate: Browse and select the certificate generated in step 1.
- SAML Key: Browse and select the key generated in step 1.



5. Click Save and Test SAML.

   On successful configuration the test status must be :

```
SAML test start....
Loading certificate
Successfully loaded certificate
Parsing certificate
Successfully parsed certificate
Parsing metadata url
```

```
Successfully parsed metadata url
Parsing root url
Successfully parsed root url
Creating SAML service
Successfully created SAML service
```

What to do next

On successful configuration download the metadata.xml file by clicking on the metadata link below the test status box.

Load the metadata file to SAML server.

```
curl -v -T metadata.xml http://<saml_server_ip>:8000/services/test
```

```
curl -v -T /Users/<user_name>/Downloads/metadata.xml http://xx.xx.xx.xxx:8000/services/test
*   Trying xx.xx.xx.xxx...
* TCP_NODELAY set
* Connected to xx.xx.xx.xxx (xx.xx.xx.xxx) port 8000 (#0)
> PUT /services/test HTTP/1.1
> Host: xx.xx.xx.xxx:8000
> User-Agent: curl/7.54.0
> Accept: */*
> Content-Length: 3795
> Expect: 100-continue
>
< HTTP/1.1 100 Continue
* We are completely uploaded and fine
< HTTP/1.1 204 No Content
< Date: Thu, 10 Dec 2020 23:49:21 GMT
<
* Connection #0 to host xx.xx.xx.xxx left intact
```

Logging in the Configuration Manager with LDAP and SAML

Procedure

1. Launch the configuration manager.

```
https://<ip_of_cm_service>:8443/admin
```

2. Choose from the any of the three login types. The **Login** option is for default admin login or login credentials for members created using the configuration manager.

> **Note:** LDAP and SAML login credential must be different.

Boomi References

Refer to these links to learn more about Boomi privacy policy, terms of service, and Boomi help documentation:

Privacy Policy
Terms of Service
Help Documentation