Oracle® Communications

Cloud Native Configuration Console REST Specifications Guide

Release 23.4.4
F88122-05
December 2024

Oracle Communications Cloud Native Configuration Console REST Specifications Guide, Release 23.4.4

F88122-05

Contents

Acronyms

The following table provides information about the acronyms and terminologies used in the document:

Table Acronyms

Acronym	Definition
AD	Active Directory
ASM	Aspen Service Mesh
BSF	Oracle Communications Cloud Native Core, Binding Support Function
cnDBTier	Oracle Communications Cloud Native Core, cnDBTier
CNC Console	Oracle Communications Cloud Native Configuration Console
CNE	Oracle Communications Cloud Native Core, Cloud Native Environment
CS	Common Service
CRUD Operations	CREATE, READ, UPDATE, DELETE
OCNADD	Oracle Communications Network Analytics Data Director
ECDSA	Elliptic Curve Digital Signature Algorithm
EIR	Equipment Identity Register
HTTPS	Hypertext Transfer Protocol Secure
IAM	Identity Access Management
KPI	Key Performance Indicator
M-CNCC	Manager CNC Console or M-CNCC (also known as mCncc) is a CNCC instance which manages local CNE common service(s) and remote Agent CNC Console (s) (A-CNCC). M-CNCC has two components: M-CNCC IAM and M-CNCC Core.
M-CNCC IAM	Manager CNC Console IAM or M-CNCC IAM (also known as mCncc Iam) is an IAM component of M-CNCC. M-CNCC IAM contains M-CNCC IAM Ingress Gateway and M-CNCC IAM back-end microservices.
M-CNCC Core	Manager CNC Console Core or M-CNCC Core (also known as mCncc Core) is a core component of M-CNCC that provides GUI and API access portal for accessing NF and OCCNE common services. M-CNCC Core contains M-CNCC Core Ingress Gateway and M-CNCC Core back-end microservices.
A-CNCC Core	Agent CNC Console is a CNCC Core instance which manages local NF(s) and local OCCNE common services(s). A-CNCC is managed by M-CNCC. A-CNCC contains A-CNCC Core Ingress Gateway. A-CNCC has no IAM component. A-CNCC is also known as A-CNCC Core or aCncc Core.
M-CNCC Kubernetes cluster	Kubernetes cluster hosting M-CNCC
mTLS	Mutual Transport Layer Security
OCNWDAF	Oracle Communications Networks Data Analytics Function

Table (Cont.) Acronyms

Acronym	Definition
Instance	NF or CNE common service managed by either M-CNCC Core or A-CNCC Core.
Site	Kubernetes Cluster
CS	CNE Common Services like Grafana, Kibana, Jaeger, Prometheus, Alertmanager and so on.
MC	Multi Cluster. In multi cluster, a single CNCC can manage NF instances that accessess different Kubernetes clusters.
MO	Mananged Objects
MOS	My Oracle Support
LDAP	Lightweight Directory Access Protocol
LDAPS	Lightweight Directory Access Protocol (Over SSL)
NRF	Oracle Communications Cloud Native Core, Network Repository Function
OCNF	Oracle Communications Network Function
OSDC	Oracle Software Delivery Cloud
OSO	Oracle Communications Operations Services Overlay
PROVGW	Provisioning Gateway
REST API	Representational State Transfer Application Programming Interface
SCP	Oracle Communications Cloud Native Core, Service Communication Proxy
SAML	Security Assertion Markup Language
SBA	Service Based Architecture
SEPP	Oracle Communications Cloud Native Core, Security Edge Protection Proxy
TLS	Transport Layer Security
UDR	Oracle Communications Cloud Native Core, Unified Data Repository
UE	User Equipment
URI	Subscriber Location Function
SSO	Single Sign On

What's New in This Guide

This section introduces the documentation updates for release 23.4.x in Oracle Communications Cloud Native Configuration Console REST Specification Guide.

Release 23.4.4 - F88122-05, December 2024

There are no updates in this release.

Release 23.4.3 - F88122-04, October 2024

There are no updates in this release.

Release 23.4.2 - F88122-03, July 2024

There are no updates in this release.

Release 23.4.1 - F88122-02, April 2024

There are no updates in this release.

Release 23.4.0 - F88122-01, December 2023

There are no updates in this release.

1

Introduction

This document provides information about how to configure the services and manageable objects in Oracle Communication Cloud Native Configuration Console using Representational State Transfer Application Program Interfaces (REST APIs).

1.1 Overview

The Cloud Native Configuration Console (CNC Console) is a single screen solution to configure and manage Network Functions (NFs).

The CNC Console has the following modules:

- **CNC Console Core (CNCC Core)**: CNCC Core acts as Graphical User Interface (GUI) or Application Programming Interface (API) portal for NFs and Oracle Communications Cloud Native Environment (OCCNE) common services. CNCC Core module is the part of CNC Console that integrates with other cloud native core network functions.
- **CNC Console Identity and Access Management (CNCC IAM)**: CNCC IAM acts as local identity provider and as a broker for external identity provider. CNCC IAM module includes the required authentication and authorization procedures such as creating and assigning roles to users.

1.2 Reference

Refer to the following documents for more information:

- *Oracle Communications Cloud Native Core, Service Communication Proxy User Guide*
- *Oracle Communications Cloud Native Core, Network Repository Function User Guide*
- *Oracle Communications Cloud Native Core, Policy User Guide*
- *Oracle Communications Cloud Native Core, Unified Data Repository User Guide*
- *Oracle Communications Cloud Native Core, Binding Support Function User Guide*
- *Oracle Communications Cloud Native Core, Security Edge Protection Proxy User Guide*
- *Oracle Communications Cloud Native Core, Network Slice Selection Function User Guide*
- *Oracle Communications Cloud Native Core, Network Repository Function Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Core, Service Communication Proxy Installation Guide*
- *Oracle Communications Cloud Native Core, Unified Data Repository Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Core, Binding Support Function Installation Guide*
- *Oracle Communications Cloud Native Core, Policy Installation Guide*
- *Oracle Communications Cloud Native Core, Security Edge Protection Proxy Installation Guide*

- *Oracle Communications Cloud Native Core, Network Slice Selection Function Installation Guide*
- *Oracle Communications Networks Data Analytics Installation and Fault Recovery Guide*
- *Oracle Communications Cloud Native Core, Certificate Management User Guide*
- *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide*

2
REST APIs and HTTP Response Codes

This chapter provides information about REST specifications used in Oracle Communications Cloud Native Configuration Console.

For HELM configurations, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

For the configurations using CNC Console, see *Oracle Communications Cloud Native Configuration Console User Guide*.

2.1 Generate access token

CNC Console uses `Generate access token` REST API to generate the access token.

Type: POST

URI:

```
POST /{realm}/protocol/openid-connect/token
```

Table 2-1 Request Body Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
client_id	string	M	Id that has been given to the client
username	string	M	Login Username
password	string	M	Login Password
grant_type	string	M	Type of Authorization used

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/realms/master/protocol/openid-connect/token
```

Example curl command:

```
curl --location --request POST 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/realms/master/protocol/openid-connect/
token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=admin-cli' \
--data-urlencode 'username=admin' \
--data-urlencode 'password=xxxxxx' \
--data-urlencode 'grant_type=password'
```

Example of the Request Body

The following is the example of the request body:

```
{
"client_id":"admin-cli",
"username": "admin",
"password": "xxxxxx",
"grant_type": "password",
}
```

Example of the Response Body

The following is the example of the response body:

```
{
    "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lGMfag",
    "expires_in": 60,
    "refresh_expires_in": 1800,
    "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkICIxM"
    "not-before-policy": 0,
    "session_state": "52dd8d7c-f8d9-4009-9c34-0262bb7d3722",
    "scope": "email profile"
}
```

Table 2-2 Supported Response Codes

Code	Description
200 OK	Get users. Returns a list of users.
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found

2.2 Create a new user

CNC Console uses `Create a new user` REST API to create a new user. The user name must be unique.

Type: POST

URI:

```
POST /{realm}/users
```

Table 2-3 Request Body Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
enabled	boolean	M	Set to true to enable the user , an disabled user can not login

Table 2-3 (Cont.) Request Body Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
username	string	M	Name of the new user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/realms/cncc/users
```

Example curl command:

```
{curl --location --request POST 'http://${cncc-iam-ingress-extrenal-ip}:$
{cncc-iam-ingress-service-port}/cncc/auth/admin/realms/cncc/users' \
--header 'Authorization: Bearer <token>' \
--header 'Content-Type: application/json' \
--data-raw '{   "enabled": true,
    "username": "user6"}'
```

Example of the Request Body

The following is the example of the request body:

```
{   "enabled": true,
          "username": "user6"}
```

Example of the Response Code

The following is the example of the response code:

```
201 Created
```

Table 2-4 Supported Response Codes

Code	Description
201 Created	Create a new user.Username must beunique. {Requirespayload}
401 Unauthorised	MissingAuthentication
404 Not Found	Realm not found
409 Conflict	User exists withsame username

2.3 Get users

CNC Console uses `Get users` REST API to return a list of users.

Type: GET

URI:

```
GET  /{realm}/users
```

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/realms/cncc/users
```

Example curl command:

```
curl --location --request GET 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/admin/realms/cncc/users' \
--header 'Authorization: Bearer <token>'
```

Example of the Response Body

The following is the example of the response body:

```
[
    {
        "id": "48cf183c-d3e4-4917-b3e5-5e01109f534c",
        "createdTimestamp": 1659952887114,
        "username": "user",
        "enabled": true,
        "emailVerified": false,
        "access": {
            "manageGroupMembership": true,
            "view": true,
            "mapRoles": true,
            "impersonate": true,
            "manage": true
        }

     }
    ,

   .....

]
```

Table 2-5   Supported Response Codes

Code	Description
200 OK	Get users. Returns alist of users, filteredaccording to queryparameters.
401 Unauthorised	MissingAuthentication
404 Not Found	Realm not found

2.4 Get single user

CNC Console uses `Get single user` REST API to get individual user details.

Type: GET

URI:

```
GET /{realm}/users/{id}
```

Table 2-6 Request Path Parameter

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	Id of the user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/users/2a3113c0-48de-46d9-a563-6ce95eabbae4
```

Example curl command:

```
curl --location --request GET 'http://10.75.241.198:30085/cncc/auth/admin/
realms/cncc/users/754d6f6b-4ccb-44f1-abf1-00d717885dbe' \
--header 'Authorization: Bearer <token>'
```

Example of the Response Body

The following is the example of the response body:

```
{
    "id": "754d6f6b-4ccb-44f1-abf1-00d717885dbe",
    "createdTimestamp": 1661232491550,
    "username": "u1",
    "enabled": true,
    "totp": false,
    "emailVerified": false,
    "disableableCredentialTypes": [],
    "requiredActions": [],
    "notBefore": 0,
    "access": {
        "manageGroupMembership": true,
        "view": true,
        "mapRoles": true,
        "impersonate": true,
        "manage": true
            }
}
```

Table 2-7 Supported Response Codes

Code	Description
200 OK	Get users. Returns a list of users, filtered according to queryparameters.
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found
404 Not Found	User not found

2.5 Delete the User

CNC Console uses `Delete the User` REST API to delete the user.

Type: DELETE

URI:

```
DELETE /{realm}/users/{id}
```

Table 2-8 Request Path Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	Id of the user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/users/2a3113c0-48de-46d9-a563-6ce95eabbae4
```

Example curl command:

```
curl --location --request DELETE 'http://${cncc-iam-ingress-extrenal-ip}:$
{cncc-iam-ingress-service-port}/cncc/auth/admin/realms/cncc/users/
7a2d3608-95b2-4a88-8efb-dad48e7778e2' \--header 'Authorization: Bearer
<token>'
```

Example of the Response Code

The following is the example of the response code:

```
204 No Content
```

Table 2-9 Supported Response Codes

Code	Description
204 No Content	Delete the user.
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found
404 Not Found	User not found

2.6 Set up a New Password for the User

CNC Console uses `Setup a New Password for the User` REST API to set up a new password for the user.

Type: PUT

URI:

```
PUT /{realm}/users/{id}/reset-password
```

Table 2-10 Request Body Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
type	string	M	Type
value	Integer	M	Value of new password
temporary	Boolean	O	To validate temporary or not

Table 2-11 Request Path Parameter

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	Id of the user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/users/2a3113c0-48de-46d9-a563-6ce95eabbae4/reset-
password
```

Example curl command:

```
curl --location --request PUT 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/admin/realms/cncc/users/
```

```
754d6f6b-4ccb-44f1-abf1-00d717885dbe/reset-password' \
--header 'Authorization: Bearer <token>' \
--header 'Content-Type: application/json' \
--data-raw '{ "type":"password","value":"<Password
Values>","temporary":false}'
```

Example of the Request Body

The following is the example of the request body:

```
{

"type":"password",

"value":"<Password Value>",

"temporary":false

}
```

Example of the Response Code

The following is the example of the response code:

```
204 No Content
```

Table 2-12 Supported Response Codes

Code	Description
204 No Content	Set up a newpassword for the CNCC user {Requires payload}.
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found
404 Not Found	User not found
400 Bad Request	Invalid password

2.7 Get Realm Level Roles

CNC Console uses `Get Realm-level Roles` REST API to get realm-level roles.

Type: GET

URI:

```
GET /{realm}/roles
```

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/roles
```

Example curl command:

```
curl --location --request GET 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/admin/realms/cncc/roles' \
--header 'Authorization: Bearer <token>'
```

Example of the Response Body

The following is the example of the response body:

```
[
    {
        "id": "fc5006e0-3927-4034-a01f-af70d779f1f8",
        "name": "ADMIN",
        "description": "Has access to all NF resources and can perform CRUD
operations",
        "composite": true,
        "clientRole": false,
        "containerId": "cncc"
    },
    {
        "id": "1acd6c4a-115a-44ae-bf5f-139577f9df0a",
        "name": "POLICY_WRITE",
        "description": "Has access to only POLICY resources and can perform
CRUD operation on Managed Objects of POLICY.",
        "composite": true,
        "clientRole": false,
        "containerId": "cncc"
    },

...]
```

Table 2-13 Supported Response Codes

Code	Description
200 OK	Generate Get Realm Level Roles

2.8 Get Realm-level Role Mappings

CNC Console uses `Get Realm-level Role Mappings` REST API to get realm-level role mappings for a specific user id .

Type: GET

URI:

```
GET /{realm}/users/{id}/role-mappings/realm
```

Table 2-14 Request Path Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	ID assigned to the user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/users
```

Example curl command:

```
curl --location --request GET 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/admin/realms/cncc/roles' \
--header 'Authorization: Bearer <token>'
```

Example of the Response Body

The following example shows the contents of the response body in JSON format:

```
[
    {
        "id": "fc5006e0-3927-4034-a01f-af70d779f1f8",
        "name": "ADMIN",
        "description": "Has access to all NF resources and can perform CRUD
operations",
        "composite": true,
        "clientRole": false,
        "containerId": "cncc"
    },
    {
        "id": "1acd6c4a-115a-44ae-bf5f-139577f9df0a",
        "name": "POLICY_WRITE",
        "description": "Has access to only POLICY resources and can perform
CRUD operation on Managed Objects of POLICY.",
        "composite": true,
        "clientRole": false,
        "containerId": "cncc"
    },

...]
```

Table 2-15 Supported Response Codes

Code	Description
200 OK	Get realm-level role mappings for a specific user id.
401 Unauthorised	Missing Authentication

Table 2-15 (Cont.) Supported Response Codes

Code	Description
404 Not Found	Realm not found
404 Not Found	User not found

2.9 Add Realm-level Role Mappings to the User

CNC Console uses `Add Realm-level Role Mappings to the User` REST API to add realm-level role mappings to the user.

Type:POST

URI:

```
POST/{realm}/users/{id}/rolemappings/realm
```

Table 2-16 Request Body Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
description	string	M	Description of the role
name	string	M	Name of the role
composite	boolean	M	To check if the role has another realm role mapped to it
clientRole	boolean	M	To check if the role has another client role mapped to it
containerId	string	M	ID of the container where the role is present
id	string	M	ID assigned to the role

Table 2-17 Request Path Parameter

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	Id of the user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/users/754d6f6b-4ccb-44f1-abf1-00d717885dbe/role-
mappings/realm
```

Example curl command:

```
curl --location --request POST 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/admin/realms/cncc/users/
754d6f6b-4ccb-44f1-abf1-00d717885dbe/role-mappings/realm' \
--header 'Authorization: Bearer <token>' \
--header 'Content-Type: application/json' \
--data-raw '[
 {
"id": "c47ba6c5-4cc8-4b59-96bd-2ef7e57121bd",
"name": "BSF_READ",
"description": "Has access to only BSF resources and can only perform READ
Managed Objects of BSF.",
 "composite": true,
 "clientRole": false,
 "containerId": "cncc"
 }
]'
```

Example of the Request Body

The following is the example of the request body:

```
[
 {
"id": "c47ba6c5-4cc8-4b59-96bd-2ef7e57121bd",
"name": "BSF_READ",
"description": "Has access to only BSF resources and can only perform READ
Managed Objects of BSF.",
 "composite": true,
 "clientRole": false,
 "containerId": "cncc"
 }
]
```

Example of the Response Code

The following is the example of the response code:

```
  204 No Content
```

Table 2-18 Supported Response Codes

Code	Description
204 No Content	Set up a newpassword for the CNCC user {Requires payload}.
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found
404 Not Found	User not found
404 Not Found	Role not found

2.10 Remove All User Sessions Associated with the User

CNC Console uses `Remove All User Sessions Associated with the User` REST API to remove all user sessions associated with the user.

Type:POST

URI:

```
POST /{realm}/users/{id}/logout
```

Table 2-19 Request Path Parameter

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	Id of the user

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/users/48cf183c-d3e4-4917-b3e5-5e01109f534c/logout
```

Example curl command:

```
curl --location --request POST '${cncc-iam-ingress-extrenal-ip}:${cncc-iam-
ingress-service-port}/cncc/auth/admin/realms/cncc/users/48cf183c-d3e4-4917-
b3e5-5e01109f534c/logout' \
--header 'Authorization: Bearer  <token>'
```

Example of the Response Code

The following is the example of the response code:

```
  204 No Content
```

Table 2-20 Supported Response Codes

Code	Description
204 No Content	Remove all user sessions associated with the user
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found
404 Not Found	User not found

2.11 Get the Client details

CNC Console uses `Get the Client details`REST API to get clients belonging to the realm.

Type: GET

URI:

```
GET/{realm}/clients
```

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/clients
```

Example curl command:

```
curl --location --request GET '${cncc-iam-ingress-extrenal-ip}:${cncc-iam-
ingress-service-port}/cncc/auth/admin/realms/cncc/clients' \--header
'Authorization: Bearer <token>'
```

Example of the Response Body

The following example shows the contents of the response body in JSON format:

```
[
    {
        "id": "b7fa17bd-135f-441b-a5f2-1ea4897e04fc",
        "clientId": "account",
        "name": "${client_account}",
        "rootUrl": "${authBaseUrl}",
        "baseUrl": "/realms/cncc/account/",
        "surrogateAuthRequired": false,
        "enabled": true,
        "alwaysDisplayInConsole": false,
        "clientAuthenticatorType": "client-secret",
        "redirectUris": [
            "/realms/cncc/account/*"
        ],
        "webOrigins": [],
        "notBefore": 0,
        "bearerOnly": false,
        "consentRequired": false,
        "standardFlowEnabled": true,
        "implicitFlowEnabled": false,
        "directAccessGrantsEnabled": false,
        "serviceAccountsEnabled": false,
        "publicClient": false,
        "frontchannelLogout": false,
        "protocol": "openid-connect",
        "attributes": {},
        "authenticationFlowBindingOverrides": {},
        "fullScopeAllowed": false,
        "nodeReRegistrationTimeout": 0,
        "defaultClientScopes": [
            "web-origins",
            "roles",
            "profile",
            "email"
        ],
        "optionalClientScopes": [
            "address",
            "phone",
            "offline_access",
            "microprofile-jwt"
        ],
        "access": {
            "view": true,
            "configure": true,
            "manage": true
        }
    },
```

2.12 Update the Client

CNC Console uses `Update the Client` REST API to update the client. This API can be used to update the redirect URI.

Type: PUT

URI:

`PUT/{realm}/clients/{id}`

Table 2-21 Request Body Parameters

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
clientId	string	M	ID given to the client which is referenced in URIs and tokens
rootUri	string	M	Url for redirection

Table 2-22 Request Path Parameter

Field Name	Data Type	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
id	string	M	ID of the client

Sample URI:

```
http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/
auth/admin/realms/cncc/clients/9faaa454-bbaf-4af0-91dd-2d01aa82776d
```

Example curl command:

```
curl --location --request PUT 'http://${cncc-iam-ingress-extrenal-ip}:${cncc-
iam-ingress-service-port}/cncc/auth/admin/realms/cncc/clients/9faaa454-
bbaf-4af0-91dd-2d01aa82776d' \
--header 'Authorization: Bearer <token>' \
--header 'Content-Type: application/json' \
--data-raw '{
     "clientId": "cncc","rootUrl": "http://10.75.241.74:8080/"
}'
```

Example of the Request Body

The following is the example of the request body:

```
{
"clientId": "cncc",
"rootUrl": "http://10.75.241.74:8080/"
}
```

Example of the Response Code

The following is the example of the response code:

```
  204 No Content
```

Table 2-23 Supported Response Codes

Code	Description
204 No Content	Update the client.This API can beused to update Redirect URI {Requires payload}.
401 Unauthorised	Missing Authentication
404 Not Found	Realm not found
404 Not Found	Could not find the client

Glossary

Index