Hybrid Cloud Automation using Cisco CloudCenter API

Ray Doerr, Advanced Services Engineer
Cisco Spark

Questions?
Use Cisco Spark to communicate with the speaker after the session

How
1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install Spark or go directly to the space
4. Enter messages/questions in the space

cs.co/ciscolivebot#DEVNET-1651
Agenda

• CloudCenter Overview
• Introduction to REST API for CloudCenter
• CloudCenter automation - User and Tenant creation
• Private/Public Cloud Virtual Machine Deployment and Demo
  • VMware
  • OpenStack
  • AWS (Amazon Web Services)
  • Microsoft AzureRM
• Conclusion
One REST-API script, Multiple Cloud Endpoints
CloudCenter Overview
Hybrid Cloud Management with CloudCenter

Single Integrated Management Platform
Full Application Lifecycle Management
Portable
Scalable
Secure

Datacenters
Private Clouds
Public Clouds
Application Profile

Graphical Modeler

“One Click” Deploy
Multi-Cloud Application Deployment

CloudCenter Manager

Application Profile

Multi-Tier Application

Public Cloud

Private Cloud

Multi-Cloud Deployment

Public Cloud

Public Cloud

Private Cloud

MySql

WordPress

HAProxy
Introduction to REST API for CloudCenter
API Overview

CloudCenter uses the REST API conventions below

- CloudCenter API Version 1.0 (v1) - supports CloudCenter 3.x & 4.x
- CloudCenter API Version 2.0 (v2) - supports CloudCenter 4.6.0 and greater
  - Structured responses with minimum details, provides resource links
  - Improved search, sort and pagination filters

- Governance is referenced using tagIds, deployments are referenced using depEnvId, and so forth. Similarly, all other resources are referenced using the corresponding ID

- Documentation located at [http://docs.cloudcenter.cisco.com/](http://docs.cloudcenter.cisco.com/) under API
REST-API

Required Information

- REST-API method (GET, POST, PUT and DELETE)

- REST-API URL consisting of:
  - IP Address or Hostname of CloudCenter CCM or Load Balancer
  - TCP Port that CloudCenter is listening on, defaults to 443
  - REST-API resource, example: v1/users

- API Credentials (details on next slide)
  - Username
  - API Key

- Request Body (not required for all REST-API calls)
  - JSON Data
REST-API

Example using curl


Or use your favorite application like Postman
API Credentials

From CloudCenter GUI:

• Select Admin-&gt;Users from the menu

• Click on “Manage API Key” for the User you want to use for API access

• From this window you should see the User Name and Key that will be required for API Access. Note: The User Name is generated by the system and not your Email address.

• You can also generate a new API Key from this window if required.

<table>
<thead>
<tr>
<th>Name</th>
<th>Email</th>
<th>Status</th>
<th>Payment Profile Status</th>
<th>User Type</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin CloudCenter</td>
<td>admin@cloudcente...</td>
<td>Enabled</td>
<td>N/A</td>
<td>Admin</td>
<td>Add Clouds, Manage API Key</td>
</tr>
<tr>
<td>Cliqr Admin</td>
<td><a href="mailto:admin@cliqrtech.com">admin@cliqrtech.com</a></td>
<td>Enabled</td>
<td>N/A</td>
<td>Owner</td>
<td>Add Clouds, Manage API Key</td>
</tr>
</tbody>
</table>
API Credentials
(continued)

• This is the Username and Key you will use for REST API calls

Manage API Key

Manage API Access Key for user "admin@cloudcenter.com"

<table>
<thead>
<tr>
<th>User Name</th>
<th>admin_b</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>5F5DEDD42E279DEE</td>
</tr>
</tbody>
</table>

[Buttons: Generate new key, Cancel]
Functional Groups of REST API’s

- User Management. Create/View/Update and Delete.
- Group Management. Create/View/Update and Delete.
- Tenant Management. Create/View/Update and Delete.
- Job Management, both v1 and v2. List/Submit/Stop/Resume/Suspend, etc.
- Application Management. List, Get, Delete, Export, Update(name, description and version only).
- Service Management. Create/View/Update and Delete.
- Cloud Management. Cloud Accounts, Cloud Regions, Image Mapping, Cloud Storage, etc.
- Inventory Management. List VM’s, Power On/Off or Reboot/Terminate VMs.
- Policy Management. Create/Enable/View/Update and Delete.
- Report Management, both v1 and v2
API Access using Python’s Request Module

```python
#!/usr/bin/env python
import urllib3
import requests
import base64
import json
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

# CloudCenter Authentication Info:
# Note: This is not the first part of the Email address on the account!
ccm_username = 'admin\'  
ccm_api_key = '5F5DEDD4E2790DEE'
baseURL = 'https://10.11.11.443/'  
credentials = base64.b64encode(ccm_username + ':' + ccm_api_key)

url = baseURL + 'v1/users'
headers = {
    'Authorization': 'Basic %s' % credentials,
    'Accept': 'application/json',
    'Content-Type': 'application/json'
}

response = requests.get(
    url,
    headers=headers,
    verify=False
)

# Raise exception if 4xx or 5xx HTTP Error
response.raise_for_status()

print("HTTP Code: %s" % response.status_code)
print("%s" % json.dumps(response.json(), sort_keys=True, indent=4, separators=(',', ': ')))
```
JSON Output of Python Script

```
HTTP Code: 200
{
    "pageNumber": 0,
    "resource": "https://10.11.11/v1/users",
    "size": 1,
    "totalElements": 1,
    "totalPages": 1,
    "users": [
        {
            "accessKeys": "https://10.11.11/v1/users/4/keys",
            "accountSource": "AdminCreated",
            "activationData": null,
            "activationProfileId": null,
            "coAdmin": false,
            "companyName": "CloudCenter Team",
            "created": 1507651172798,
            "detail": null,
            "disableReason": null,
            "emailAddr": "janedoe@cisco.com",
            "emailVerified": true,
            "enabled": true,
            "externalId": "",
            "firstName": "Jane",
            "hasSubscriptionPlanType": false,
            "id": "4",
            "lastName": "Doe",
            "lastUpdated": 1507651172773,
            "password": "--- redacted ---",
            "phoneNumber": "",
            "resource": "https://10.11.11/v1/users/4",
            "status": "ENABLED",
            "tenantAdmin": true,
            "tenantId": "3",
            "type": "TENANT",
            "username": "jdoe_4"
        }
    ]
}
```
CloudCenter Automation
- User and Tenant Creation
User Creation

URL: /v1/users

• Create User without Activation requires:
  • Authentication Credentials of Tenant Admin
  • tenantId of Tenant Admin where this user will be created

```json
{
  "firstName": "John",
  "lastName": "Doe",
  "password": "Cisco2018",
  "emailAddr": "john.doe@cisco.com",
  "companyName": "Cisco Systems Inc."
  "phoneNumber": "999-888-7777",
  "externalId": "",
  "tenantId": {{tenantId}}
}
```
User Creation and Activation

URL: /v1/users

- Create User with Activations requires:
  - planId of Usage Plan to assign to this User
  - contractId of Contact to assign to this User
  - activateRegions, Array of regionId’s to assign to this User

```json
{
  "firstName": "John",
  "lastName": "Doe",
  "password": "Cisco2018",
  "emailAddr": "john.doe@cisco.com",
  "companyName": "Cisco Systems Inc.",
  "phoneNumber": "999-888-7777",
  "externalId": "",
  "tenantId": "[tenantId]",
  "activationData": {
    "planId": "1",
    "contractId": "1",
    "activateRegions": [
      {
        "regionId": "2"
      },
      {
        "regionId": "4"
      },
      {
        "regionId": "42"
      }
    ],
    "agreeToContract": false,
    "sendActivationEmail": false,
    "importApps": false
  }
}
```
Tenant Creation

User must already be created

- userId is for the User that will be granted Tenant Admin for this new Tenant
- shortName is the Tenant ID you use for CloudCenter GUI login
Private/Public Cloud Virtual Machine Deployments
Creating a Deployment
Get Applications

URL: /v1/apps

- Returns a list of Applications with Resource URL.

```json
{
  "apps": [
    {
      "id": "31",
      "resource": "https://172.18.31.11/v1/apps/31?version=1.0",
      "perms": [
        "read",
        "execute"
      ],
      "name": "Base_CentOS6.9",
      "version": "null",
      "description": "Image mapped to Snapshot",
      "serviceTierId": "Base_CentOS6.9-31",
      "versions": [
        "1.0"
      ],
      "executor": "NTierExecutor",
      "category": "NTier"
    },
    {
      "id": "29",
      "resource": "https://172.18.31.11/v1/apps/29?version=1.0",
      "perms": [
        "read",
        "execute"
      ],
      "name": "Base_CentOS7.4",
      "version": "null",
      "description": "Image mapped to Snapshot",
      "serviceTierId": "Base_CentOS7-29",
      "versions": ["1.0"
      ],
      "executor": "NTierExecutor",
      "category": "NTier"
    }
  ]
}
```
Get Application

URL: /v1/apps/<app-id>?version=x.y

- Gather the “id”, “metadatas” and “version”
- Collect the “id” from serviceTiers list
- For each ServiceTier gather the Supported Clouds
Get Environments

URL: /v1/environments

- Gather the Environment “id”
- Gather "regionId", cloudAccountId" and “basicSettings” from the “associatedClouds” section
Get Supported Clouds Information
URL: /v1/apps/<app-id>/cloudConfigs/<cloud-id>

- InstanceType
- numOfCPUs
- memorySize
- localStorageSize
Get Supported Clouds Information

- URL: /v1/apps/<app-id>/cloudConfigs/<cloud-id>
- Gather the Instance details (VMware, OpenStack and AWS shown below)
VM Deployment
Common section of REST-API body for all Cloud Endpoints.

- Values that change per Cloud Endpoint are:
  - environmentId
  - cloudRegionId
  - accountId
  - instanceType

```
"appId": "29",
"appVersion": "1.0",
"name": "RD-CentOS7-{{$randomInt}}",
"metadatas": [],
"environmentId": "1",
"tagIds": [],
"securityProfileIds": [],
"policyIds": null,
"preventTermination": false,
"parameters": {
  "appParams": [],
  "cloudParams": {
    "cloudRegionId": "2",
    "accountId": "1"
  }
},
"links": [ ]
```
VM Deployment

Jobs section of REST-API body, list to support MultiTier Apps

```
"jobs": [
    {
        "tierId": "30",
        "policyIds": null,
        "tagIds": [],
        "securityProfileIds": [],
        "parameters": {
            "appParams": [
                {
                    "name": "referredJob",
                    "value": ""
                },
                {
                    "name": "SSHPreference",
                    "value": "NO_PREFERENCE"
                }
            ]
        }
    },
    {
        "cloudParams": {
            "cloudRegionId": "2",
            "accountId": "1",
            "volumes": [],
            "rootVolumeSize": "0",
            "cloudProperties": [
                {
                    "name": "FullClone",
                    "value": "false"
                },
                {
                    "name": "RootDiskResizeConfig",
                    "value": "false"
                },
                {
                    "name": "UserDatastoreCluster",
                    "value": ""
                }
            ],
            "nics": [],
            "instance": "Small_1xCPU_2GB"
        }
    },
    {
        "numNodesToLaunch": 1
    }
]
```
VM Deployment

Response

- Here you can see the Parent Job with a single Child Job, since this was a single Tier application deployment.
- Each Child Job will return the Resource URL that you can use to pool the status of that job.
Polling Deployment Status

Job Status – Polling both Parent and Child Jobs

Job Submitted

```
{  
"id": "2844",
"resource": "https://172.18.31.11/v2/jobs/2844",
"name": "RD-CentOS7-1517179498",
"description": ",",
"status": "JobInProgress",
"jobStatusMessage": ",",
"startTime": "1517179502924",
"endTime": "",
}
```

Job Completed

```
{  
"id": "2844",
"resource": "https://172.18.31.11/v2/jobs/2844",
"name": "RD-CentOS7-1517179498",
"description": ",",
"status": "JobRunning",
"jobStatusMessage": "",
"startTime": "1517179502924",
"endTime": "",
}
```

```
{  
"id": "2845",
"resource": "https://172.18.31.11/v2/jobs/2845",
"name": "CentOS_2",
"description": "",
"status": "JobStarting",
"jobStatusMessage": "",
"startTime": "1517179503324",
"endTime": "",
}
```

```
{  
"id": "2845",
"resource": "https://172.18.31.11/v2/jobs/2845",
"name": "CentOS_2",
"description": "",
"status": "JobRunning",
"jobStatusMessage": "",
"startTime": "1517179503324",
"endTime": "",
}
```
Demo

- CloudCenter Application Profile deployment using REST API
- Deploy to Private/Public Clouds
  - VMware
  - OpenStack
  - AWS (Amazon Web Services)
  - Microsoft AzureRM
One REST-API script, Multiple Cloud Endpoints

```python
def submit_job_example(instanceType):
    # Submit Job Example
    print("\nSubmit Job/Deployment")
    # Main Section - Jobs
    job_appParams = []
    job_appParams.append({'name': 'referedJob', 'value': ''})
    job_appParams.append({'name': 'SSHReference', 'value': 'NO'})
    nics = []
    nics.append(nics_defaults)
    job_cloudParams = {}
    job_cloudParams['cloudRegionId'] = cloudRegionId
    job_cloudParams['accountId'] = accountId
    job_cloudParams['volumes'] = []
    job_cloudParams['rootVolumesize'] = """'
    job_cloudParams['cloudProperties'] = job_cloudProperties
    job_cloudParams['nics'] = nics
```
Questions?
Use Cisco Spark to communicate with the speaker after the session

How
1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install Spark or go directly to the space
4. Enter messages/questions in the space

How
1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install Spark or go directly to the space
4. Enter messages/questions in the space

cs.co/ciscolivebot#DEVNET-1651
• Please complete your Online Session Evaluations after each session

• Complete 4 Session Evaluations & the Overall Conference Evaluation (available from Thursday) to receive your Cisco Live T-shirt

• All surveys can be completed via the Cisco Live Mobile App or the Communication Stations

Don’t forget: Cisco Live sessions will be available for viewing on-demand after the event at CiscoLive.com/Online.
Continue Your Education

- Demos in the Cisco campus
- Walk-in Self-Paced Labs
- Tech Circle
- Meet the Engineer 1:1 meetings
- Related sessions
Thank you
You’re