Cisco APO Use Case Runbook

## Technology: IoT

## Level: FLY

## Name: **IoT API Data Integration to Cisco Kinetic for Cities**

**Scenario:** Public Sector Customer wants to understand energy usage for the lighting in street and shared public spaces. They have integrated Cisco Kinetic for Cities (CKC) into their lighting systems. By using the API integration, they can leverage the Data contained in the CKC database in other applications.

**Customer Story:** Public Entity X has established a target of reducing the cost of lighting and energy consumption in the street and shared public spaces. By incorporating the Cisco Kinetic for Cities platform and using the API connectivity to pull out the lighting data, this entity can embed the lighting information on the public web portal. They can show their compliance and movement towards their mandated goal without receiving calls and emails to support the reporting of these goals. This can save 100’s of hours of employee time which can **save thousands of dollars a month, on top of the savings they get from controlling the lighting aspects via the CKC platform**.

**Partner Story:** VAR Partner X taught their deployment engineers to use Cisco Kinetic for Cities and was able to help their customers automate the lighting on the street and shared public spaces and to sell web integration services as part of their scope, thereby increasing the product and services revenue on the deal through the sale of follow on automation services after the install.

**Code Exchange Link:** https://developer.cisco.com/codeexchange/github/repo/CiscoDevNet/scc-ckc-api-examples

## **Use Case Playbook**

This use case consists of 3 learning labs and a challenge. The 3 learning labs are written in Python and are located at: <https://github.com/CiscoDevNet/scc-ckc-api-examples>.

*NOTE: need requirements.txt for dependencies in git repo. Libraries for urllib, ssl, matplotlib, xmltodict, and json are required for API calls.*

Each lab builds on the previous lab. The labs are:

* Authentication and API Requests (Lab1)
* Retrieving Information from the Cisco Kinetic for Cities Platform (Lab2)
* Requesting Real Time Data from the Cisco Kinetic for Cities Platform (Lab3)

The labs are run against the Cisco DevNet Cisco Kinetic for Cities Sandbox. For information on Cisco Kinetic for Cities see: <https://www.cisco.com/c/en/us/solutions/industries/smart-connected-communities/kinetic-for-cities.html>

To use the Sandbox, you will first need to reserve a copy. Go to: <https://devnetsandbox.cisco.com/RM/Diagram/Index/454ca1f8-5db1-46e3-9516-969611e316b7?diagramType=Topology> (better link?) and look for the “Reserve” button on the upper right menu under your username.

All the platform APIs (except authentication) require an access token. Before any of these APIs can be used, you must authenticate with the Cisco Kinetic for Cities Platform to get the token needed to make successful calls.

### Access Details:

When you reserve the “Cisco Kinetic for Cities” Sandbox you will receive API access credentials via email at the start of the reservation. There is **no requirement** for VPN setup for the Cisco Kinetic for Cities Sandbox. It is recommended to reserve the lab for at least 2 hours to complete this lesson.

In the email, the following details will be shared with you as part of the access details:

* Token Generation URL
* client\_id
* client\_secret
* username

You will also receive an email on your registered email address with a link to set their password for the first time. This password needs to be used along with the client\_id, client\_secret and username attributes to generate the access token.

The lab will use the CKC API. To download a copy of the API for further exploration, see: [Cisco Kinetic for Cities API Documentation](https://developer.cisco.com/docs/cisco-kinetic-for-cities/).

You will want to make sure you pull down the Python code for the Labs to your local machine. The code is at: <https://github.com/CiscoDevNet/scc-ckc-api-examples>. You will need to ensure that you have Python (tested on Python V3.5+) and the urllib, json, and ssl libraries installed on your machine.

To use the [Postman](https://www.getpostman.com/) tool to manually explore the API, you can import the Postman Collection and Environment JSON files into your Postman client. Update your Postman Environment with the API credentials and start manually trying out the APIs.

Postman Collection: [https://devnetsandbox.cisco.com/sandbox-instructions/ckc/CKC%204.0%20API%20List.postman\_collection.jso](https://devnetsandbox.cisco.com/sandbox-instructions/ckc/CKC%204.0%20API%20List.postman_collection.json)n

Postman Environment: <https://devnetsandbox.cisco.com/sandbox-instructions/ckc/CKC%204.0%20DevNet%20Sandbox.postman_environment.json>

You can then follow the 3 Learning Labs in order to learn how to log in to the CKC Lab, read user information, and retrieve real-time lighting data using python code to make the API calls:

[Learning Lab 1: Authentication](https://learninglabs.cisco.com/lab/ckc101/step/1) - This lab should take approximately 20 minutes

You will prove connectivity and see your user information from the CKC platform.

Try running this lab with both Postman and then with Python.

You can add your username and password to the Python program so you don’t have to retype it each time. NOTE: you will have to adjust it in each of the 3 labs.

[Learning Lab 2: Retrieving User Information](https://learninglabs.cisco.com/lab/ckc102/step/1) - This lab should take approximately 20 minutes

You will be learning to retrieve CKC capabilities and location information from the CKC platform.

Try running this lab with both Postman and then with Python.

[Learning Lab 3: Accessing Real-time Sensor Data](https://learninglabs.cisco.com/lab/ckc103/step/1) - This lab should take approximately 45 minutes

You will learn to pull lighting information from the CKC platform.

Try running this lab with both Postman and then with Python.

To graph the output of the Lighting, you will need to make sure you have installed the additional libraries that are required. Both “pip install matplotlib” and “pip install xmltodict” are required.

Once you have a Lighting graph, try to pull information from the Parking section and graph it. You will need to adjust the program to look in the CKC platform for: parkingspot = r['ParkingSpot'] that contains ParkingSpot ID: 'sid’ and the status: 'occupied['value']'

*if parkingspot['sid']:*

*idofSpot = parkingspot['sid']*

*if parkingspot['state']:*

*state = parkingspot['state']*

*if state['occupied']:*

*isOccupied = state['occupied']*

*if isOccupied['value']:*

*spaceFree = isOccupied['value']*

*print(idofSpot, ' is free ', spaceFree)*

## 

## Completion Challenge

Once you complete the Learning labs you will be ready for the challenge.

In order to provide a Proof of Performance, we are going to set the following goal. Take what you have learned in the 3 labs and combine it with learning from the IoT lab on Graphing Real Time IoT Data. (see <https://developer.cisco.com/learning/lab/sd-wan-firewall-statistics-apis/step/1>) If you have not completed this lab, please do so as it will be required to complete the challenge.

The challenge is to make a Grafana dashboard which displays the real time information from the CKC “Parking” module to graphically display in real time the number of free versus used parking spots by city.

Once you have completed the challenge, please host a 2 minute demo of your application running and send the link to {{ [email\_addr@cisco.com](mailto:email_addr@cisco.com) }} where your video will be reviewed for credit. We suggest doing a Webex demo and recording it. Share the recording link via the email for review.