Cisco PX Cloud Data Miner User Guide

The PX Cloud Data Miner script is a sample python script that will extract all the Partner Experience (PX) Cloud data for all customers for a single partner. It will gather the JSON data and convert it to files in CSV, JSON or both for injecting into any platform that can digest these two file formats. This sample script is meant as a tool to start you on your development journey or as a guide to develop your platform to access these API's, it is not meant to be used 'as-is' for all use cases.

This script was built from the Reference SDK:

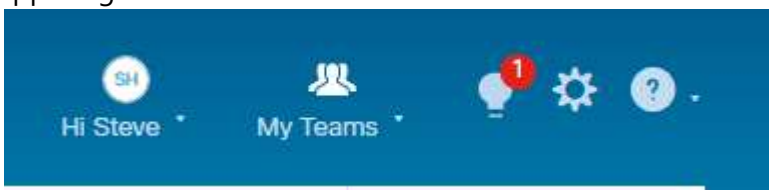
https://developer.cisco.com/docs/px-cloud/#!partner-and-customer-data-api

Table of Contents

Requirements

This script will require the following:

1. Python 3.9 or higher
2. The following Python Libraries:
    a. Required libraries
        i. json for parsing json data
        ii. csv for parsing csv data
        iii. os for file/folder management
        iv. shutil for removing entire folders
        v. zipfile for testing/extracting zip files
        vi. time for setting a sleep/wait timer
        vii. sys for changing the output to log file
        viii. math for rounding up to whole numbers
        ix. datetime for posting time stamps in the logging
    b. Optional Libraries
        i. boto3 for accessing the AWS S3 storage
3. A PX Cloud account and at least 1 customer that has authorized your access to their data.
    a. To request authorization from the customer, log into PX Cloud and click the gear icon in the upper right of the UI:



On the Manage Users screen, click the Customers icon on the left:

Click the request access button next to the name of the customer you want access to:

Customer will get a notification in the CX Cloud UI and access will be granted once accepted.

4. Access to the Cisco APIs via the internet
5. Local access to the local directory to read/write/delete files created by the script.

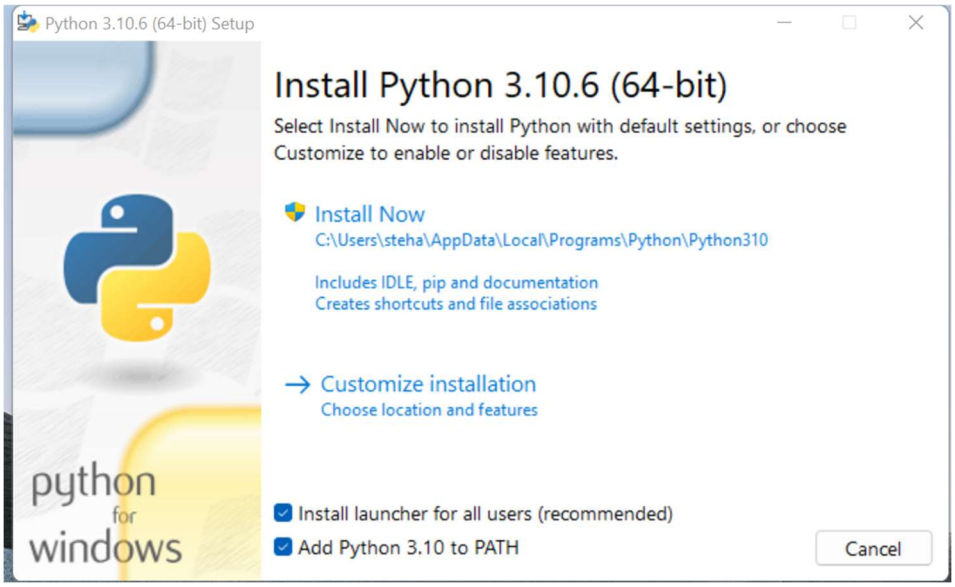
Python Setup

While there are too many environments to document them all, I will cover python install for Windows, macOS and Ubuntu Linux.

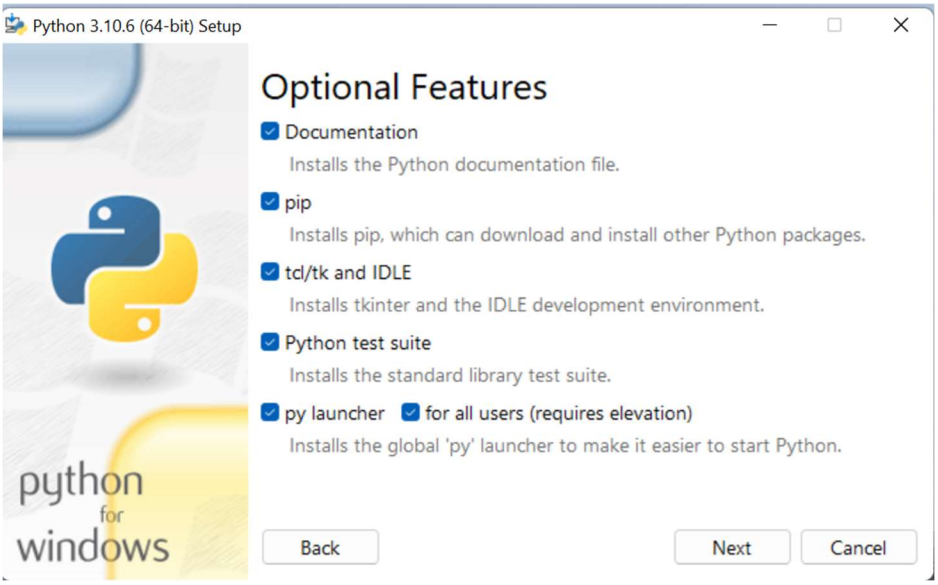
Windows setup

Open the official Python website in your web browser. https://www.python.org/downloads/windows/ and navigate to the Downloads tab for Windows. Choose the latest Python 3 release. As of this writing, the latest Python 3.10.6 version. Click on the link to download Windows x86 executable installer if you are using a 32-bit installer. In case your Windows installation is a 64-bit system, then download Windows x86-64 executable installer. Once the installer is downloaded, run the Python installer.

Check the ***Install launcher for all users*** check box. Further, you should check the ***Add Python*** to path check box to include the interpreter in the execution path.

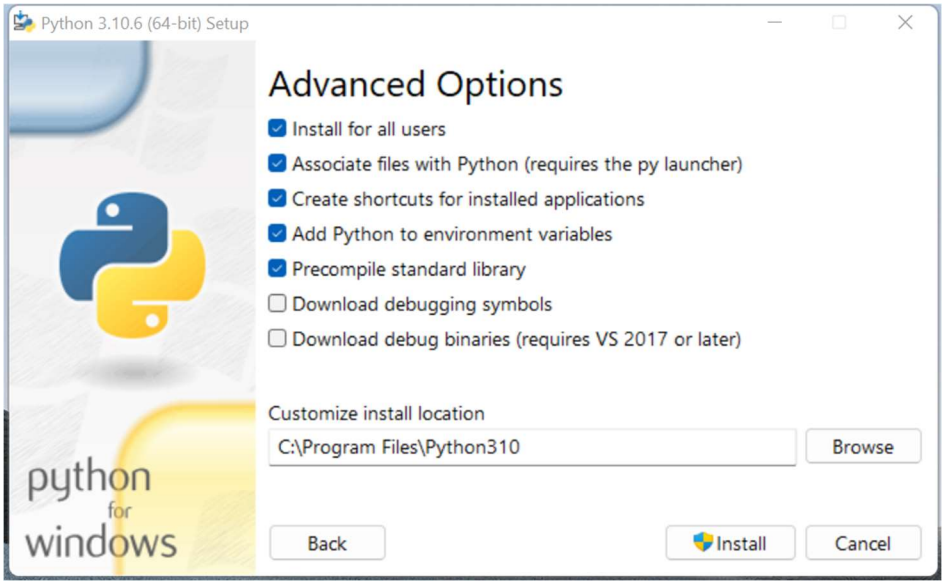


Select ***Customize installation***. Choose the optional features by checking the following check boxes:



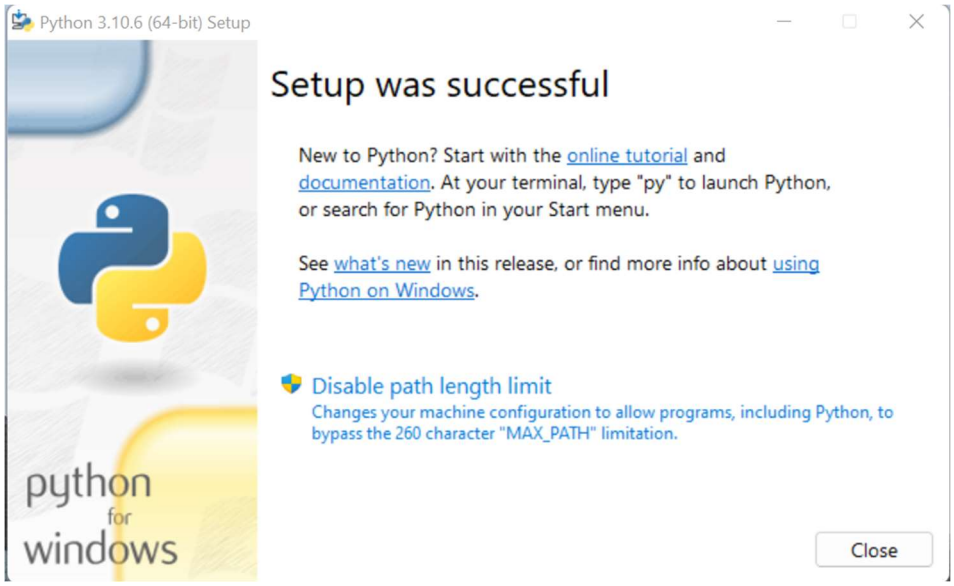
1. Documentation
2. pip
3. tcl/tk and IDLE (to install tkinter and IDLE)
4. Python test suite (to install the standard library test suite of Python)
5. Install the global launcher for `.py` files. This makes it easier to start Python
6. Install for all users.

Click Next, this takes you to **Advanced Options** available while installing Python.

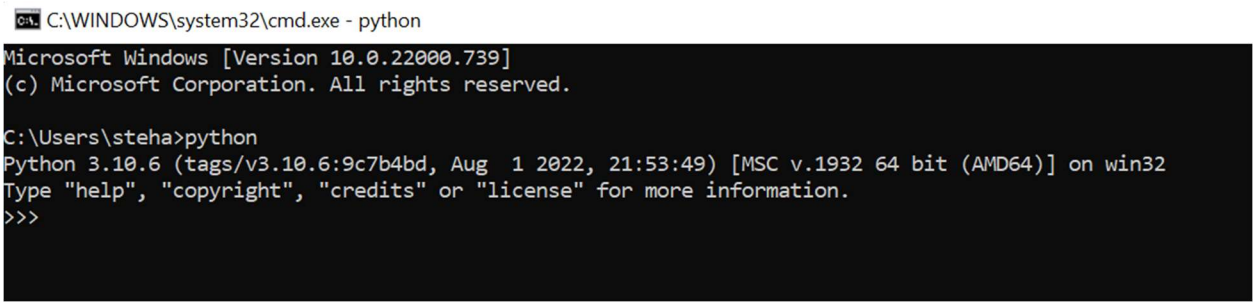


Here, select the **Install for all users** and **Add Python to environment variables** check boxes. Optionally, you can select the **Associate files with Python**, **Create shortcuts for installed applications** and other advanced options. Make note of the python installation directory displayed in this step. You would need it for the next step. After selecting the Advanced options, click Install to start installation.

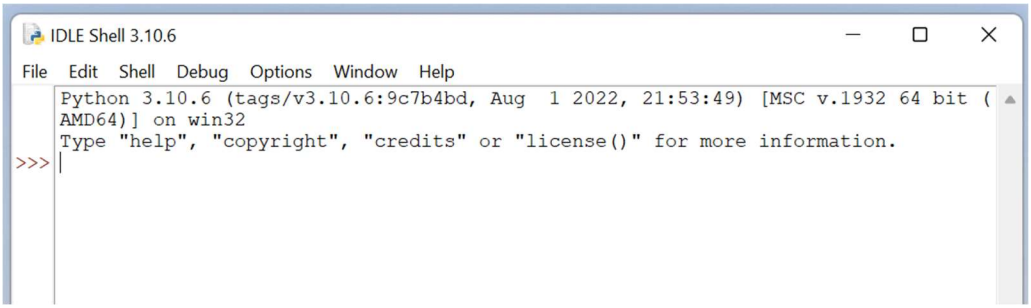
Once the installation is over, you will see a Python Setup Successful window.



You have now successfully installed Python on Windows. You can verify if the Python installation is successful either through the command line:

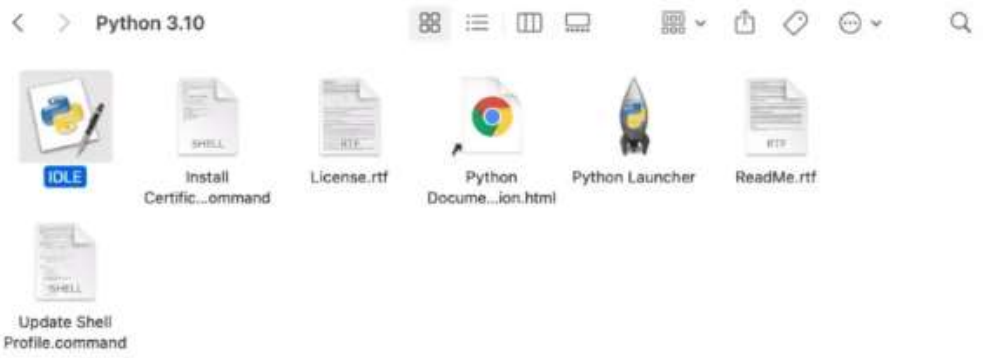


or through the IDLE app that gets installed along with the installation:

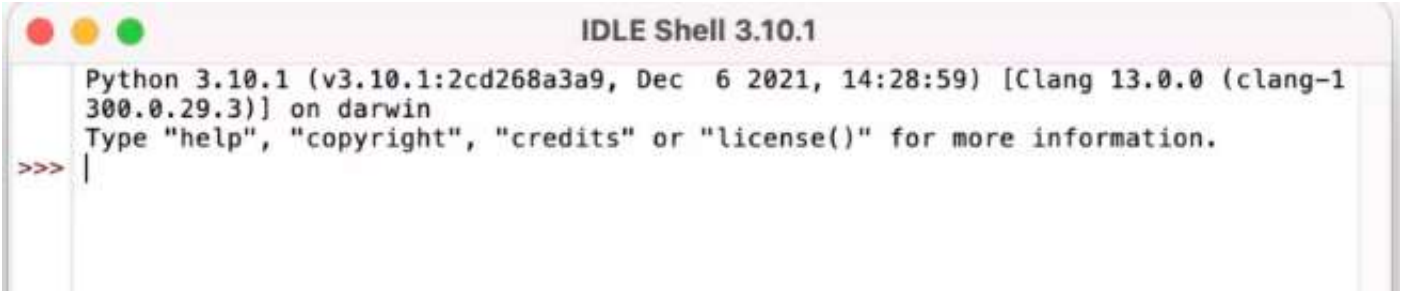


macOS setup

Open the official Python website in your web browser. https://www.python.org/downloads/macos/ and navigate to the Downloads tab for macOS. Choose the latest Python 3 release. As of this writing, the latest Python 3.10.6 version. Click on the link to download the installer file labeled *macOS 64-bit universal2 installer*. Once the download is complete, double-click the package to start installing Python. The installer will walk you through a wizard to complete the installation, and in most cases, the default settings work well, so install it like the other applications on macOS. You may also have to enter your Mac password to let it know that you agree with installing Python. When the installation completes, it will open up the Python folder.



Let's verify that the latest version of Python and IDLE installed correctly. To do that, double-click IDLE, which is the integrated development environment shipped with Python. If everything works correctly, IDLE shows the Python shell as follows:



Ubuntu Linux setup

Ubuntu comes with Linux preinstalled.

To see which version of Python 3 you have installed, open a command prompt and run

$ python3 --version

If you are using Ubuntu 16.10 or newer, then you can easily install Python 3.10 with the following commands:

$ sudo apt-get update

$ sudo apt-get install python3.

File output location

The following describes where to locate the files generated by Data Miner and their filenames and some key elements in the files.

Data Miner will create 4 empty folders each time it is run:

outputcsv/ to store CSV files

outputjson/ to store JSON files

outputlog/ to store logs if log_to_file is set to true

temp/ to store temporary file used by Data Miner

Installing required Python libraries

Going forward, I will document the process for a Windows installation, other OS's will be similar.

There are 2 libraries that are not included in the Python install which will need to be installed prior to running the script.

1. requests (this is a requirement)
2. boto3 (this is optional if not uploading data to an AWS S3 storage account)

The requests library is needed for making API calls to the API endpoints. The boto3 library is for sending data to an Amazons S3 storage account.

The boto3 library is optional if not needed but you'll also need to comment out the 3 lines with a "#" symbol in the code that loads this library and remove the values for S3 in the config.ini file.

From this in the python file:

```
import boto3
from botocore.exceptions import ClientError
from botocore.exceptions import NoCredentialsError
```

To this:

```
# import boto3
# from botocore.exceptions import ClientError
# from botocore.exceptions import NoCredentialsError
```

From this in the config.ini file:

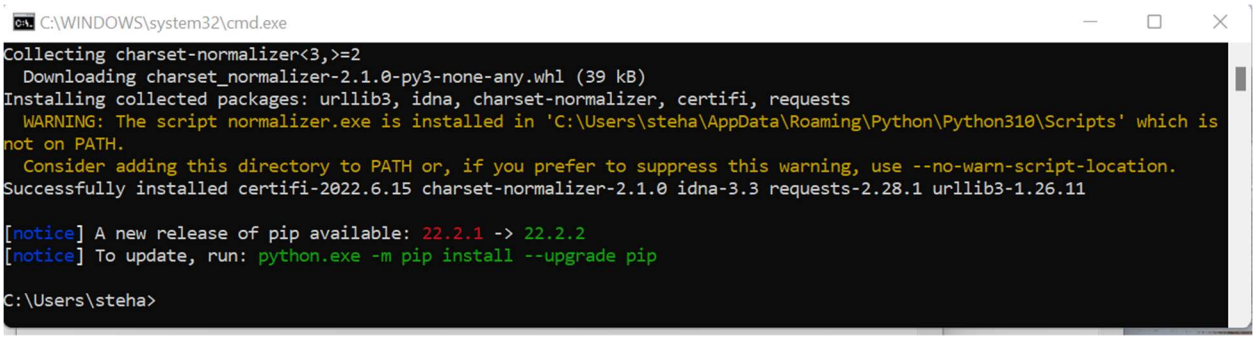
```
s3access_key = AWS S3 Access Key
s3access_secret = AWS S3 Client Secret
s3bucket_folder = AWS S3 Bucket Folder
s3bucket_name = AWS S3 Bucket Name
```

To this:

```
s3access_key =
s3access_secret =
s3bucket_folder =
s3bucket_name =
```

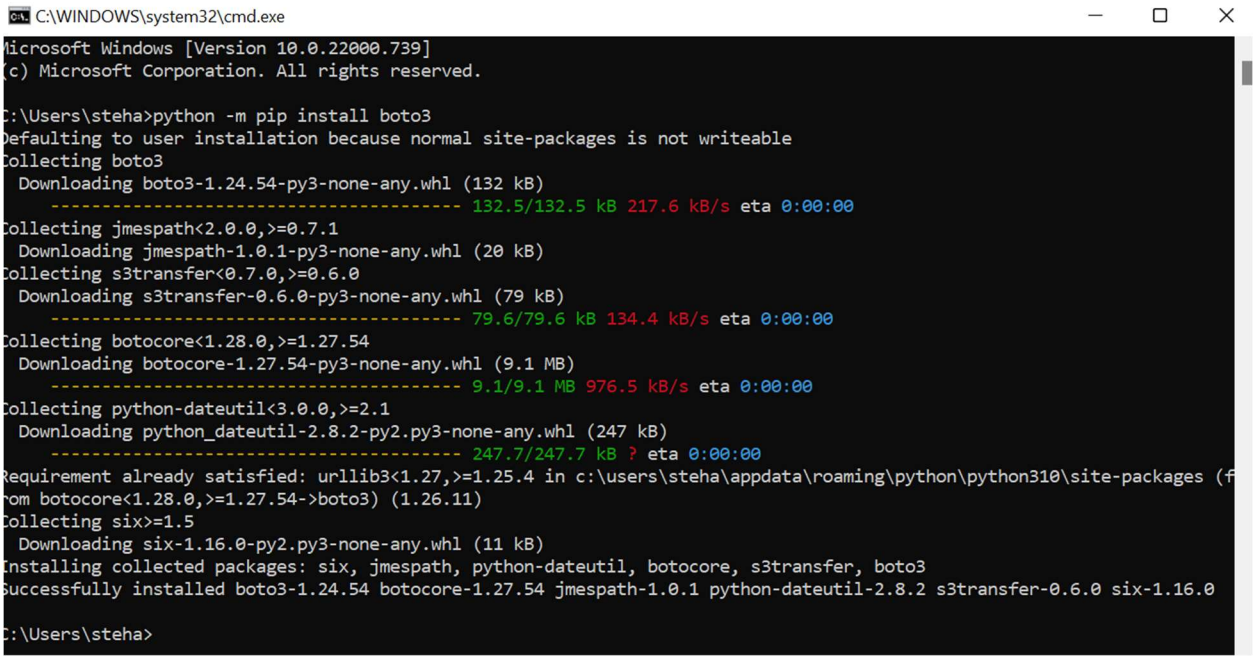
To install the requests library, open a command prompt and type in the following:

NOTE: If you get a notice to upgrade pip, follow those instructions.



If you will be using AWS S3, execute the following command below:

python -m pip install boto3



Data Miner script installation

Installation only requires that the script pxcloudv#.#.#.#.py (# in the name references the version number) and the config.ini be in the same directory and the logged in user has write/delete privileges in that directory. When running the script for the 1st time, it will auto generate a config.ini in the same directory as the Data Miner script and then exit. It will look like this:

[credentials]

Used to store the PX Cloud Client ID

pxc_client_id = PX Cloud API Client ID

Used to store the PX Cloud Client Secret

pxc_client_secret = PX Cloud Client Secret

Used to store the Amazon Web Services S3 (Simple Storage Service) Key

s3access_key = S3 Key

Used to store the Amazon Web Services S3 (Simple Storage Service) Secret

s3access_secret = S3 Secret

Used to store the Amazon Web Services S3 (Simple Storage Service) Bucket Folder

s3bucket_folder = Bucket Folder

Used to store the Amazon Web Services S3 (Simple Storage Service) Bucket Name

s3bucket_name = Bucket Name

[settings]

time to wait between errors in seconds, default = 1

the number of days to retrieve fault data for. this value can be 1, 7, 15, 30, default = 30

the maximum number of items to return per api call maximum is 50, default = 50

used for setting a debug level (0 = low, 1 = medium, 2 = high), default = 0

send all screen logging to a file (1=true, 0=false), default = 0

test the code by running through the entire sequence x times..., default = 1

generate output in the form of both, json or csv. 1=both 2=json 3=csv, default = 1

use production url (1=true 0=false) if false, use sandbox url, default = 1

Edit these values following the Configuration section and re-run Data Miner.

Configuration settings

When editing the config.ini it is important to understand the following keys and what they represent.

[credentials]

pxc_client_id = # this is the partners PX Cloud Client ID

pxc_client_secret = # this is the partners PX Cloud Client Secret

s3access_key = # this is the partners AWS S3 Access Key (omit if not used)

s3access_secret = # this is the partners AWS S3 Client Secret (omit if not used)

s3bucket_folder = # this is the partners AWS S3 folder URL (omit if not used)

s3bucket_name = # this is the partners AWS S3 bucket name (omit if not used)

NOTE: if you not using AWS S3, omit the values for the s3 keys and Data Miner will not attempt to upload the data to S3

[settings]

time to wait between errors in seconds, default = 1

wait_time = 1

the number of days to retrieve fault data for. this value can be 1, 7, 15, 30, default = 30

pxc_fault_days = 30

the maximum number of items to return per api call maximum is 50, default = 50

max_items = 50

used for setting a debug level (0 = low, 1 = medium, 2 = high), default = 0

debug_level = 0

send all screen logging to a file (1=true, 0=false), default = 0

log_to_file = 0

test the code by running through the entire sequence x times..., default = 1

testloop = 1

generate output in the form of both, json or csv. 1=both 2=json 3=csv, default = 1

outputformat = 3

use production url (1=true 0=false) if false, use sandbox url, default = 1

useproductionurl = 1

Partner Business Insight Reports

A. Customers

The Customers Report will provide the list of all the customers

   a. CSV Naming Convention:
      i. Customer.csv
   b. JSON Naming Convention:
      i. Customer.json

B. Contracts

The Contracts Report will provide a list of partner contracts transacted with Cisco.

   a. CSV Naming Convention:
      i. Contract.csv
   b. JSON Naming Convention:
      i. Contract.json

C. Contract Details

The Contracts Details Report will provide a list of partner contract line items transacted with Cisco.

   a. CSV Naming Convention:
      i. Contract_Details.csv
   b. JSON Naming Convention:
      i. {Customer Name}_Contract_Details_{ContractNumber}_Page_{page}_of_{total}.json

D. Partner Offers

This API will fetch all the offers created by the Partners.

   a. CSV Naming Convention:
      i. Partner_Offers.csv
   b. JSON Naming Convention:
      i. Partner_Offers.json

E. Partner Offer Sessions

This API will fetch all the active and inactive sessions of all the Offers created by the Partners.

   a. CSV Naming Convention:
      i. Partner_Offer_Sessions.csv
   b. JSON Naming Convention:
      i. Partner_Offer_Sessions.json

F. Success Tracks

This API will get customer success tracks which will provide the use case details.

   a. CSV Naming Convention:
      i. SuccessTracks.csv
   b. JSON Naming Convention:
      i. SuccessTracks.json

Customer Data Reports

A. Assets

The Assets report gives the list of all assets (products like network elements and other devices) owned by customer.

   a. CSV Naming Convention:
      i. Assets.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Assets_{UniqueReportID}.json

B. Hardware

The Hardware report will provide hardware inventory details for a specific device and customer.

   a. CSV Naming Convention:
      i. Hardware.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Hardware_{UniqueReportID}.json

C. Software

The Software report will provide details about the software installed on a device.

   a. CSV Naming Convention:
      i. Software.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Software_{UniqueReportID}.json

D. Purchased Licenses

Purchased Licenses report will provide details of all the licenses.

   a. CSV Naming Convention:
      i. Purchased_Licenses.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Purchased_Licenses_{UniqueReportID}.json

E. Licenses

The Licenses report will provide License details along with related asset information.

   a. CSV Naming Convention:
      i. Licenses.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Licenses_{UniqueReportID}.json

F. Customer Lifecycle

The Customer Lifecycle will provide CX solution, Use case and Pitstop info.

   a. CSV Naming Convention:
      i. Lifecycle.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Lifecycle.json

G. Security Advisories

The Security Advisories report provides security vulnerability information including CVE and CVSS for devices associated with customer ID.

   a. CSV Naming Convention:
      i. Security_Advisories.csv
   b. JSON Naming Convention:
      i. {Customer ID}_SecurityAdvisories_{UniqueReportID}.json

H. Field Notices

The Field Notices Report provides details of all notifications published and their associated details.

   a. CSV Naming Convention:
      i. Field_Notices.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Field_Notices_{UniqueReportID}.json

I. Priority Bugs

The Priority Bugs report provides many bug details including asset name and ID, serial number, IP address and other fields.

   a. CSV Naming Convention:
      i. Priority_Bugs.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Priority_Bugs_{UniqueReportID}.json

Optimal Software Version Reports

A. Software Groups
   The Software Groups report returns the Software Group information for the given customerID.
   a. CSV Naming Convention:
      i. SoftwareGroup.csv
   b. JSON Naming Convention:
      i. SoftwareGroup.json
B. Software Group suggestions
   The Software Group suggestions reports returns Software Group suggestions, including detailed information about Cisco software release recommendations and current Cisco software releases running on assets in the Software Group.
   a. CSV Naming Convention:
      i. SoftwareGroup_Suggestions_Trends.csv
      ii. SoftwareGroup_Suggestions_Summaries.csv
      iii. SoftwareGroup_Suggestions_Releases.csv
   b. JSON Naming Convention:
      i. {Customer ID}_SoftwareGroup_Suggestions_{Success Track ID}_{Suggestion ID}.json
C. Software Group Suggestions-Assets
   Software Group Suggestions-Assets returns information about assets in the Software Group based on the customerID and softwareGroupId provided.
   a. CSV Naming Convention:
      i. SoftwareGroup_Suggestions_Assets.csv
   b. JSON Naming Convention:
      i. {Customer ID}_SoftwareGroup_Suggestion_Assets_{Software Group ID}.json
D. Software Group Suggestions-Bug List
   Software Group Suggestions-Bug List returns information on bugs, including ID, description, and affected software releases.
   a. CSV Naming Convention:
      i. SoftwareGroup_Suggestions_Bug_Lists.csv
   b. # JSON Naming Convention:
      i. {Customer ID}_SoftwareGroup_Suggestions_Bug_Lists_{Machine Suggestion ID}_Page_{page}_of_{total}.json

E. Software Group Suggestions-Field Notices

Software Group Suggestions-Field Notices returns field notice information, including ID number, title, and publish date.

   a. CSV Naming Convention:
      i. SoftwareGroup_Suggestions_Field_Notices.csv
   b. JSON Naming Convention:
      i. {Customer ID}_SoftwareGroup_Suggestions_Field_Notices_{Machine Suggestion ID}_Page_{page}_of_{total}.json

F. Software Group Suggestions-Advisories

Software Group Suggestions-Advisories returns software advisory information, including ID number, version number, and severity level.

   a. CSV Naming Convention:
      i. SoftwareGroup_Suggestions_Security_Advisories.csv
   b. JSON Naming Convention:
      i. {Customer ID}_SoftwareGroup_Suggestions_Security_Advisories_{Machine Suggestion ID}_Page_{page}_of_{total}.json

Automated Fault Management Reports

A. Faults

The Faults Report returns fault information for the customerId provided.

   a. CSV Naming Convention:
      i. Automated_Fault_Management_Faults.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Automated_Fault_Management_Faults_{Success Track ID}.json

B. Fault Summary

The Fault Summary Report returns detailed information for a fault based on the fault signatureId and customerId provided.

   a. CSV Naming Convention:
      i. Automated_Fault_Management_Fault_Summary.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Automated_Fault_Management_Fault_Summary_{Fault ID}_{Success Track ID}.json

C. Affected Assets

The Affected Assets Reports returns information about the customer assets affected by the fault, based on the signatureId and customerId.

   a. CSV Naming Convention:
      i. Automated_Fault_Management_Fault_Summary.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Automated_Fault_Management_Fault_Summary_Affected_Assests_{Fault ID}_{Success Track ID}.json

Regulatory Compliance Check Reports

A. Compliance Violations
   Compliance Violations Report returns information about the rules violated for the customerId provided.
   a. CSV Naming Convention:
      i. Regulatory_Compliance_Violations.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Regulatory_Compliance_Violations_{Success Track ID}.json

B. Assets Violating Compliance Rule
   Assets Violating Compliance Rule Report returns information about the customer assets in violation of the rule which is based customer, policy, and rules.
   a. CSV Naming Convention:
      i. Regulatory_Compliance_Assets_violating_Compliance_Rule.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Assets_Violating_Compliance_Rule_{Success Track ID}_{File #}.json

C. Policy Rule Details
   Policy Rule Details Report returns information about the policy the rule belongs to.
   a. CSV Naming Convention:
      i. Regulatory_Compliance_Policy_Rule_Details.csv
   b. JSON Naming Convention:
      i. # {Customer ID}_Policy_Rule_Details_{Success Track ID}_{Page #}.json

D. Compliance Suggestions
   Compliance Suggestions Report returns information about the violated rule conditions based on the policy, and rule information provided.
   a. CSV Naming Convention:
      i. Compliance_Suggestions.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Compliance_Suggestions__{Success Track ID}_{Page #}.json

E. Assets with Violations
   Assets with Violations Report returns information about assets that have at least one rule violation based on the customerId provided.
   a. CSV Naming Convention:
      i. Regulatory_Compliance_Assets_With_Violations.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Assets_with_Violations_{successTrackId}.json

F. Asset Violations

Asset Violations Report returns information about the rules violated by an asset based on the information provided.

a. CSV Naming Convention:
    i. Regulatory_Compliance_Asset_Violations.csv
b. JSON Naming Convention:
    i. {Customer ID}_Asset_Violations_{sourceSystemId}_Page_##.json

G. Obtained

Obtained Report returns information about whether the customer has successfully configured the regulatory compliance feature and has violation data available.

a. CSV Naming Convention:
    i. Regulatory_Compliance_Obtained.csv
b. JSON Naming Convention:
    i. {Customer ID}_Obtained_{successTrackId}.json

Risk Mitigation Check Reports

A. Crash Risk Assets
   Crash Risk Assets Report returns information about assets with a crash risk value of Medium or High for the customerId provided.
   a. CSV Naming Convention:
      i. Crash_Risk_Assets.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Crash_Risk_Assets_{Success Track ID}.json

B. Crash Risk Factors
   Crash Risk Factors Report returns the risk factors that contribute to the crash risk value of the asset.
   a. CSV Naming Convention:
      i. Crash_Risk_Factors.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Crash_Risk_Factors_{Success Track ID}_{Asset ID}.json

C. Similar Assets
   Similar Assets Report returns information about similar assets with a lower crash risk rating than the specified assetId based on the similarityCriteria and customerId provided.
   a. CSV Naming Convention:
      i. Similar_Assets.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Similar_Assets_{Success Track ID}_{Asset ID}_{Features}.json

D. Assets Crashed in last 1d, 7d, 15d, 90d
   Assets Crashed Report in last 1d, 7d, 15d, 90d provides the list of devices with details (i.e. Asset, Product Id, Product Family, Software Version, Crash Count, First Occurrence and Last Occurrence) by customer ID that have crashed in the last 1d,7d,15d,90d based on the filter input. Default sort is by lastCrashDate.
   a. CSV Naming Convention:
      i. Crash_Risk_Assets_Last_Crashed.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Crash_Risk_Assets_Last_Crashed_In_{timePeriod}_Days_{Success Track ID}.json

E. Asset Crash History
   Asset Crash History Report returns information for each crash that occurred during the last 365 days for the assetId and customerId provided in the last 1 year. Default sort is by timeStamp.
   a. CSV Naming Convention:
      i. Asset_Crash_History.csv
   b. JSON Naming Convention:
      i. {Customer ID}_Asset_Crash_History_{Success Track ID}_{Base64 Asset ID}.json