pycentral

Release 1.0.0

aruba-automation@hpe.com

Apr 05, 2024

INTRODUCTION

CHAPTER

ONE

ABOUT PYCENTRAL PACKAGE

Aruba Central is an unified cloud-based network management and configuration platform for campus, branch, remote and data center networks. There are various needs for automation and programmability like automating repetitive tasks, configuring multiple devices, monitoring and more. This python package is to programmatically interact with Aruba Central via REST APIs.

CHAPTER

TWO

INSTALLATION INSTRUCTIONS

Python comes with a package management system, pip. Pip can install, update, or remove packages from the Python environment. It can also do this for virtual environments. It is a good idea to create a separate virtual environment for this project. A guide to virtual environments can be found here.

To use pip to install the pyaoscx package from the official Python Package Index, PyPI, excecute the following command:

```
pip3 install pycentral
```

To install package with extras *colorLog* which displays logs in color on stdout.

```
pip3 install pycentral[colorLog]
```

Important: This package is compatible with Python 3. Python 2 is not supported.

CHAPTER

THREE

GETTING STARTED

3.1 Package Structure

```
pycentral
│   README.md
│   Contributing.md
│   ...
│
└──pycentral
│   │   base.py
│   │   configuration.py
│   │   ...
│   │
│   └──workflows
│       │   workflows_utils.py
│       │   ...
│
└──docs
    │   ...
```

The `pycentral` package is a directory containing files and subdirectories. Contained directly within the top level `pycentral` directory are informative files such as the readme, licensing information, contribution guidelines, and release notes. Also contained within the `pycentral` package are the subdirectories relevant to the developer, `pycentral` and `workflows`.

3.1.1 pycentral

The `pycentral` subfolder contains the Aruba Central Python modules. Each module contains multiple Python classes. Each Class is a representation of some of the Aruba Central's API Reference category. Each class has its own function definitions that are used to make a single REST API call. The REST API calls are performed using the Python `requests` library, which provides functions to make HTTP GET, POST, PUT, PATCH and DELETE requests as supported by Aruba Central API Gateway.

In addition to some function definitions for API endpoints listed in API Reference page, there is also a file named `base.py` containing function *ArubaCentralBase.command()*. This function can make any REST API call using the API endpoint URL, HTTP Method, HTTP query params and HTTP payload as required by an API endpoint.

3.1.2 workflows

`workflows` folder contains scripts that combine multiple REST API calls based on function definitions in the `pycentral` modules to achieve a network automation use-case involving multiple steps or repetitive actions that has to be done in scale. Each script contains comments that describe step-by-step the operations being performed.

Check out the central-python-workflows repository for workflows that utilize the Pycentral library. Some of the workflows are -

1. Device Provisioning
2. Device Onboarding
3. MSP Customer Onboarding
4. Inventory to Excel Workflows
5. WLAN Workflows

3.1.3 Executing scripts

Refer Aruba's Developer Hub for the sample scripts written using `pycentral` modules and `pycentral.workflows` workflows.

Important: For more information about Aruba Central and REST APIs visit Aruba Central's page in Aruba Developer Hub.

CHAPTER

FOUR

PYCENTRAL

4.1 Indices and tables

- genindex
- modindex
- search

4.2 pycentral package

4.2.1 pycentral.audit_logs module

class pycentral.audit_logs.**Audit**

Bases: `object`

Get the audit logs and event logs with the functions in this class.

get_eventlogs(*conn*, *limit=100*, *offset=0*, *group_name=None*, *device_id=None*, *classification=None*, *start_time=None*, *end_time=None*)

Get audit events for all groups, sort by time in in reverse chronological order.This API returns the first 10,000 results only. Please use filter in the API for more relevant results.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **limit** (*int, optional*) – Maximum number of audit events to be returned, defaults to 100
- **offset** (*int, optional*) – Number of items to be skipped before returning the data, useful for pagination, defaults to 0
- **group_name** (*str, optional*) – Filter audit events by Group Name, defaults to None
- **device_id** (*str, optional*) – Filter audit events by Target / Device ID. Device ID for AP is VC Name and Serial Number for Switches, defaults to None
- **classification** (*str, optional*) – Filter audit events by classification, defaults to None
- **start_time** (*int, optional*) – Filter audit logs by Time Range. Start time of the audit logs should be provided in epoch seconds, defaults to None

- **end_time** (*int, optional*) – Filter audit logs by Time Range. End time of the audit logs should be provided in epoch seconds, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_eventlogs_detail(*conn*, *id*)

Get details of an audit event/log

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **id** (*str*) – ID of audit event

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_traillogs(*conn*, *limit=100*, *offset=0*, *username=None*, *start_time=None*, *end_time=None*, *description=None*, *target=None*, *classification=None*, *customer_name=None*, *ip_address=None*, *app_id=None*)

Get audit logs, sort by time in in reverse chronological order. This API returns the first 10,000 results only. Please use filter in the API for more relevant results. MSP Customer Would see logs of MSP's and tenants as well.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **limit** (*int, optional*) – Maximum number of audit events to be returned, defaults to 100
- **offset** (*int, optional*) – Number of items to be skipped before returning the data, useful for pagination, defaults to 0
- **username** (*str, optional*) – Filter audit logs by User Name, defaults to None
- **start_time** (*int, optional*) – Filter audit logs by Time Range. Start time of the audit logs should be provided in epoch seconds, defaults to None
- **end_time** (*int, optional*) – Filter audit logs by Time Range. End time of the audit logs should be provided in epoch seconds, defaults to None
- **description** (*str, optional*) – Filter audit logs by Description, defaults to None
- **target** (*str, optional*) – Filter audit logs by target, defaults to None
- **classification** (*str, optional*) – Filter audit logs by Classification, defaults to None
- **customer_name** (*str, optional*) – Filter audit logs by Customer NameFilter audit logs by Customer Name, defaults to None
- **ip_address** (*str, optional*) – Filter audit logs by IP Address, defaults to None
- **app_id** (*str, optional*) – Filter audit logs by app_id, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_traillogs_detail`(*conn*, *id*)

Get details of an audit log

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **id** (*str*) – ID of audit event

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.2 pycentral.base module

`class pycentral.base.ArubaCentralBase`(*central_info*, *token_store=None*, *logger=None*, *ssl_verify=True*, *user_retries=10*)

Bases: `object`

This is the Base class for Aruba Central which handles -

1. token management (OAUTH2.0, cache/storage for reuse and refresh token). Default token caching is in unencrypted file. Override with your implementation for secure handling of access tokens.
2. command function makes API requests, handles token expiry and auto-refresh expired tokens. Auto-refresh feature is functional only when OAUTH2.0 is provided. Otherwise, refresh expired tokens at your convenience.

Parameters

- **central_info** (*dict*) – Containing information related Aruba Central and API Gateway for HTTPS connection.
  - keyword username: (Optional) Aruba Central username string. Provide for OAUTH2.0 if access token is not provided.
  - keyword password: (Optional) Aruba Central password string. Provide for OAUTH2.0 if access token is not provided.
  - keyword client_id: (Optional) API Gateway client_id string, Provide for OAUTH2.0 and refresh token API.
  - keyword client_secret: (Optional) API Gateway client_secret string, Provide for OAUTH2.0 and refresh token API.
  - keyword customer_id: (Optional) API Gateway client_secret string, Provide for OAUTH2.0 and refresh token API.
  - keyword base_url: Provide the API Gateway string base FQDN in format *https://<Aruba-Central-API-Gateway-Domain-Name>* You need to provide either a base_url or cluster_name.

  - keyword cluster_name: Provide the name of the cluster where the Aruba Central account is provisioned. You need to provide either a base_url or cluster_name. You can find the list of supported clusters in the constants.py file.
  - keyword token: (Optional) The token dict is mutually excluive with OAUTH2.0 parameters. token dict should consist of the following key-value pairs -

    {"access_token": "xxxx", "refresh_token": "XXYY"}

- **token_store** (*dict, optional*) – Placeholder for future development which provides options to secrely cache and reuse access tokens. defaults to None
- **user_retries** (*int, optional*) – Number of times API call should be retried after a rate-limit error HTTP 429 occurs.
- **logger** (class:*logging.logger*, optional) – Provide an instance of class:*logging.logger*, defaults to logger class with name "ARUBA_BASE".
- **ssl_verify** (*bool, optional*) – When set to True, validates SSL certs of Aruba Central API Gateway, defaults to True

command(*apiMethod*, *apiPath*, *apiData={}*, *apiParams={}*, *headers={}*, *files={}*)

This function calls requestURL to make an API call to Aruba Central after gathering parameters required for API call. When an API call fails with HTTP 401 error code, the same API call is retried once after an attempt to refresh access token or create new access token is made.

Parameters

- **apiMethod** (*str*) – HTTP Method for API call. Should be one of the supported methods for the respective Aruba Central API endpoint.
- **apiPath** (*str*) – Path to the API endpoint as required by API endpoint. Refer Aruba Central API reference swagger documentation.
- **apiData** (*dict, optional*) – HTTP payload for the API call as required by API endpoint. Refer Aruba Central API reference swagger documentation, defaults to {}
- **apiParams** (*dict, optional*) – HTTP url query parameters as required by API endpoint. Refer Aruba Central API reference swagger, defaults to {}
- **headers** (*dict, optional*) – HTTP headers as required by API endpoint. Refer Aruba Central API reference swagger, defaults to {}
- **files** (*dict, optional*) – Some API endpoints require a file upload instead of apiData. Provide file data in the format accepted by API endpoint and Python requests library, defaults to {}

Returns

HTTP response with HTTP status_code and HTTP response payload.

- keyword code: HTTP status code
- keyword msg: HTTP response payload

Return type
dict

createToken()

This function generates a new access token for Aruba Central via OAUTH2.0 APIs.

Returns
The token dict conisting of access_token and refresh_token.

Return type
dict

`getToken()`

This function attempts to obtain token from storage/cache otherwise creates new access token. Stores the token if new token is generated.

Returns
API Gateway token dict consisting of access_token and refresh_token

Return type
dict

`handleTokenExpiry()`

This function handles 401 error as a result of HTTP request. An attempt to refresh token is made. If refreshing token fails, this function tries to create new access token. Stores token for reuse. If all the attemps fail, program is terminated.

`loadToken()`

This function handles loading API Gateway token from cache/storage. Default token load is from local JSON file. Override this function with storeToken function to implement secure access token caching mechanism.

Raises
UserWarning – Warning to capture empty JSON

Returns
token dict loaded from default implementation of locally stored JSON file consisting of access_token and refresh_token.

Return type
dict

`oauthAccessToken(auth_code)`

This function is Step3 of the OAUTH2.0 mechanism to generate API access token.

Parameters
auth_code (*str*) – Auth code from Step2 of OAUTH2.0.

Returns
token information received from API call consisting of access_token and refresh_token.

Return type
dict

`oauthCode(csrf_token, session_token)`

This function is Step2 of the OAUTH2.0 mechanism to get auth code using CSRF token and session key.

Parameters

- **csrf_token** (*str*) – CSRF token obtained from Step1 of OAUTH2.0
- **session_token** (*str*) – Session key obtained from Step1 of OAUTH2.0

Returns
Auth code received in the response payload of the API call.

Return type
str

`oauthLogin`()

This function is Step1 of the OAUTH2.0 mechanism to generate access token. Login to Aruba Central is performed using username and password.

Returns
Tuple with two strings, csrf token and login session key.

Return type
tuple

`refreshToken`(*old_token*)

This function refreshes the provided API Gateway token using OAUTH2.0 API. In addition to the input args, this function also depends on the class variable definitions client_id and client_secret.

Parameters
old_token (*dict*) – API Gateway token dict consisting of refresh_token.

Raises

- **UserWarning** – Raises warning when validation of availability of the required parameters fails.
- **UserWarning** – Raises warning when token input param is provided but it doesn't have refresh_token.

Returns
Token dictionary consisting of refreshed access_token and new refresh_token.

Return type
dict

`requestUrl`(*url*, *data={}*, *method='GET'*, *headers={}*, *params={}*, *files={}*)

This function makes API call to Aruba Central via python requests library.

Parameters

- **url** (*string*) – HTTP Request URL string
- **data** (*dict, optional*) – HTTP Request payload, defaults to {}
- **method** (*str, optional*) – HTTP Request Method supported by Aruba Central, defaults to "GET"
- **headers** (*dict, optional*) – HTTP Request headers, defaults to {}
- **params** (*dict, optional*) – HTTP url query parameteres, defaults to {}
- **files** (*dict, optional*) – files dictionary with file pointer depending on API endpoint as acceped by Aruba Central, defaults to {}

Returns
HTTP response of API call using requests library

Return type
class:*requests.models.Response*

`storeToken`(*token*)

This function handles storage of token for later use. Default storage is unencrypted JSON file and is not secure. Override this function and loadToken function to implement secure access token caching mechanism.

Parameters
token (*dict*) – API Gateway token dict consisting of access_token and refresh_token.

Returns
True if the access_token caching is successful.

Return type
bool

`validateOauthParams()`

This function validates if all required parameters are available to obtain access_token via OAUTH2.0 mechanism in Aruba Central.

Returns
True when validation of availability of the required parameters passed.

Return type
bool

`validateRefreshTokenParams()`

This function validates if all required parameters are available for refresh token API call.

Returns
True when the validation of the availability of required parameters passed.

Return type
bool

`class pycentral.base.BearerAuth(token)`

Bases: `AuthBase`

This class uses Bearer Auth method to generate the authorization header from Aruba Central Access Token.

Parameters
token (*str*) – Aruba Central Access Token

4.2.3 pycentral.base_utils module

`class pycentral.base_utils.clusterBaseURL`

Bases: `object`

This class helps to fetch the API Base URL when Aruba Central cluster name is provided.

`getAllBaseURLs()`

This method returns the list of base URLs of all the clusters of Aruba Central

Returns
List of all valid base URLs of Aruba Central

Return type
list

`getBaseURL(cluster_name)`

This method returns the API Base URL of the provided Aruba Central cluster.

Parameters
cluster_name (*str*) – Name of the Aruba Central cluster whose base_url needs to be returned

Returns
Base URL of provided cluster

Return type
str

pycentral.base_utils.**console_logger**(*name*, *level='DEBUG'*)

This method create an instance of python logging and sets the following format for log messages. <date> <time> - <name> - <level> - <message>

param name
String displayed after data and time. Define it to identify from which part of the code, log message is generated.

type name
str

param level
Loggin level set to display messages from a certain logging level. Refer Python logging library man page, defaults to "DEBUG"

type level
str, optional

return
An instance of class logging

rtype
class:*logging.Logger*

pycentral.base_utils.**get_url**(*base_url*, *path=''*, *params=''*, *query={}*, *fragment=''*)

This method constructs complete URL based on multiple parts of URL.

Parameters

- **base_url** (*str*) – base url for a HTTP request
- **path** (*str, optional*) – API endpoint path, defaults to ''
- **params** (*str, optional*) – API endpoint path parameters, defaults to ''
- **query** (*dict, optional*) – HTTP request url query parameters, defaults to {}
- **fragment** (*str, optional*) – URL fragment identifier, defaults to ''

Returns
Parsed URL

Return type
class:*urllib.parse.ParseResult*

pycentral.base_utils.**parseInputArgs**(*central_info*)

This method parses user input, checks for the availability of mandatory arguments. If the user opts to provide a cluster_name instead of base_url, the method will use the cluster_name to fetch the base_url. Optional missing parameters in central_info variable is initialized as defined in C_DEFAULT_ARGS.

Parameters
central_info (*dict*) – central_info dictionary as read from user's input file.

Returns
parsed central_info dict with missing optional params set to default values.

Return type
dict

pycentral.base_utils.**tokenLocalStoreUtil**(*token_store*, *customer_id='customer'*, *client_id='client'*)

Utility function for storeToken and loadToken default access token storage/cache method. This function generates unique file name for a customer and API gateway client to store and load access token in the local machine for

reuse. The format of the file name is tok_<customer_id>_<client_id>.json. If customer_id or client_id is not provided, default values mentioned in args will be used.

Parameters

- **token_store** (*dict*) – Placeholder to support different token storage mechanism.
  - keyword type: Place holder for different token storage mechanism. Defaults to local storage.
  - keyword path: path where temp folder is created to store token JSON file.
- **customer_id** (*str, optional*) – Aruba Central customer id, defaults to "customer"
- **client_id** (*str, optional*) – API Gateway client id, defaults to "client"

Returns
Filename for access token storage.

Return type
str

pycentral.base_utils.**valid_url**(*url*)

This method verifies & returns the URL in a valid format. If the URL is missing the https prefix, the function will prepend the prefix after verifiying that its a valid base URL of an Aruba Central cluster.

Parameters
base_url (*str*) – base url for a HTTP request

Returns
Valid Base URL

Return type
str

4.2.4 pycentral.configuration module

class pycentral.configuration.**ApConfiguration**

Bases: object

A python class to manage Aruba Central access points with API's from the AP configuration category.

change_wlan_status(*conn*, *group_name*, *wlan_name*, *new_wlan_status*)

This function lets you enable or disable the specified WLAN in a UI group.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of Aruba Central group which has the WLAN
- **wlan_name** (*str*) – Name of WLAN whose status has to be changed
- **new_wlan_status** (*bool*) – Status of WLAN - True => Enable WLAN, False => Disable WLAN

Returns
True when WLAN status was updated successfully. Otherwise, it will return False

Return type
bool

get_ap_config(*conn*, *group_name*)

Get whole configuration in CLI format of an UI group or AOS10 device.

Parameters
group_name (*str*) – Central group name or AOS10 AP serial number.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

replace_ap(*conn*, *group_name*, *data*)

Replace whole configuration for a Central UI group or AOS10 device. Configuration is in CLI format.

Parameters

- **group_name** (*str*) – Central group name or AOS10 AP serial number.
- **data** (*json*) – AP CLI configuration commands. Format for json value is a list of CLI command strings.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

class pycentral.configuration.**ApSettings**

Bases: object

A Python class to manage AP settings such as AP name, zonename, etc.

get_ap_settings(*conn*, *serial_number: str*)

Get existing AP settings

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **serial_number** (*str*) – Serial number of an AP. Example: CNBRHMV3HG

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

update_ap_settings(*conn*, *serial_number: str*, *ap_settings_data: dict*)

Update Existing AP Settings

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **serial_number** (*str*) – Serial number of an AP. Example: CNBRHMV3HG
- **ap_settings_data** (*dict*) – Data to update ap settings.
  - keyword hostname: Name string to set to the AP
  - keyword ip_address: IP Address string to set to AP. Should be set to "0.0.0.0" if AP get IP from DHCP.

- keyword zonename: Zonename string to set to AP
- keyword achannel: achannel string to set to AP
- keyword atxpower: atxpower string to set to AP
- keyword gchannel: gchannel string to set to AP
- keyword gtxpower: gtxpower string to set to AP
- keyword dot11a_radio_disable: dot11a_radio_disable string to set to AP
- keyword dot11g_radio_disable: dot11g_radio_disable string to set to AP
- keyword usb_port_disable: usb_port_disable string to set to AP

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`class pycentral.configuration.Devices`

Bases: `object`

A python class consisting of functions to manage Aruba Central Devices via REST API

`get_device_templates_from_hash`(*conn*, *template_hash*, *offset=0*, *limit=20*, *exclude_hash=False*, *device_type='IAP'*)

List of devices with its group name and template information is populated

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **template_hash** (*str*) – Template_hash of the template for which list of devices needs to be populated.
- **offset** (*int, optional*) – Pagination offset, defaults to 0
- **limit** (*int, optional*) – Pagination limit with Max 20, defaults to 20
- **exclude_hash** (*bool, optional*) – Fetch devices template details not matching with provided hash, defaults to False
- **device_type** (*str, optional*) – Device type (i.e. IAP/ArubaSwitch/ MobilityController/CX), defaults to "IAP"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`get_devices_config_details`(*conn*, *device_serial*, *details=True*)

Get

1. central side configuration.
2. Device running configuration.
3. Configuration error details.
4. Template error details and status of a device belonging to a template group.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of Aruba device.
- **details** (*bool, optional*) – Usually pass false to get only the summary of a device's configuration status. Pass true only if detailed response of a device's configuration status is required. Passing true might result in slower API response and performance effect comparatively., defaults to True

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`get_devices_configuration`(*conn*, *device_serial*)

Get last known device configuration for a device.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of Aruba device.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`get_devices_group`(*conn*, *device_serial*)

Get group name for a device

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of Aruba device

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_devices_group_templates`(*conn*, *device_type='IAP'*, *include_groups=[]*, *exclude_groups=[]*, *all_groups=False*, *offset=0*, *limit=20*)

Get templates for devices in a group or multiple groups

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_type** (*str, optional*) – Device type (i.e. IAP/ArubaSwitch/ MobilityController/CX), defaults to "IAP"
- **include_groups** (*list, optional*) – Fetch devices templates in list of groups, defaults to []

- **exclude_groups** (*list, optional*) – Fetch devices templates not in list of groups, defaults to []
- **all_groups** (*bool, optional*) – Set to True, to fetch devices templates details for all the groups(Only allowed for user having all_groups access or admin), defaults to False
- **offset** (*int, optional*) – Pagination offset, defaults to 0
- **limit** (*int, optional*) – Pagination limit with Max 20, defaults to 20

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

get_devices_templates(*conn*, *device_serials*)

Get existing templates for list of devices

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serials** (*list*) – List of serial number of Aruba devices

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

get_switch_variablized_templates(*conn*, *device_serial*)

Get template and variabled for Aruba device (switch)

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of Aruba device.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

move_devices(*conn*, *group_name*, *device_serials*)

Move list of devices to group and assign specified group in device management page

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of a group where devices will be moved
- **device_serials** (*list*) – A list of device serials to be moved to the mentioned group

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

set_switch_ssh_credentials(*conn*, *device_serial*, *username*, *password*)

Set SSH connection information of Aruba Device (switch)

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of Aruba device.
- **username** (*str*) – SSH username to set to the device
- **password** (*str*) – SSH password to set to the device

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

class pycentral.configuration.**Groups**

Bases: object

A python class consisting of functions to manage Aruba Central Groups via REST API

clone_create_group(*conn*, *new_group_name*, *existing_group_name*)

Clone and create new group from a given group with the given name. The configuration of the new group will be inherited from the given group.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **new_group_name** (*str*) – New group name to be created.
- **existing_group_name** (*str*) – Existing group name to be cloned.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

create_group(*conn*, *group_name*, *group_password*, *wired_template=False*, *wireless_template=False*)

Create new group given a group name, group password and configuration mode(UI or template mode of configuration) to be set per device type.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of the group to be created.
- **group_password** (*str*) – Password for the group to be created
- **wired_template** (*bool, optional*) – Set to True to make the configuration mode for switches to template mode, defaults to False
- **wireless_template** (*bool, optional*) – Set to True to make the configuration mode for IAPs and Gateways to template mode, defaults to False

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`delete_group`(*conn*, *group_name*)

Delete an existing group

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of the group to be deleted.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_config_mode_groups`(*conn*, *groups*)

For each group in the provided list, the configuration mode for Instant APs and Gateways is specified under the 'Wireless' field and for switches under the 'Wired' field. The configuration mode is specified as a boolean value indicating if the device type is managed using the template mode of configuration or not.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **groups** (*list*) – A list of group names with max of 20 groups.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_groups`(*conn*, *offset=0*, *limit=20*)

Get list of groups

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **offset** (*int, optional*) – Pagination offset, defaults to 0
- **limit** (*int, optional*) – Pagination limit with Max 20, defaults to 20

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`class` pycentral.configuration.**Templates**

Bases: `object`

A python class consisting of functions to manage Aruba Central Templates via REST API

`create_template`(*conn*, *group_name*, *template_name*, *template_filename*, *device_type='IAP'*, *version='ALL'*, *model='ALL'*)

Upload a new template file

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of the group in which the template file will be created.
- **template_name** (*str*) – Name for the template to be created
- **template_filename** (*str*) – Name of the template file in local machine to be sent to central using API
- **device_type** (*str, optional*) – Type of the Aruba device, defaults to "IAP"
- **version** (*str, optional*) – Firmware version property of template., defaults to "ALL"
- **model** (*str, optional*) – Model property of template. For 'ArubaSwitch' device_type, part number (J number) can be used, defaults to "ALL"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

delete_template(*conn*, *group_name*, *template_name*)

Delete an existing template

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of the group in which the template file exists.
- **template_name** (*str*) – Name of the template to be deleted.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

get_template(*conn*, *group_name*, *device_type=''*, *template_name=''*, *version=''*, *model=''*, *q=''*, *offset=0*, *limit=20*)

Get all templates in group. Query can be filtered by name, device_type, version, model or version number. Response is sorted by template name.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of the group for which the templates will be being queried.
- **device_type** (*str, optional*) – Filter on device_type, defaults to ""
- **template_name** (*str, optional*) – Filter on template name, defaults to ""
- **version** (*str, optional*) – Filter on version property of template, defaults to ""
- **model** (*str, optional*) – Filter on model property of template. For 'ArubaSwitch' device_type, part number(J number) can be used., defaults to ""
- **q** (*str, optional*) – Search for template OR version OR model, q will be ignored if any of filter parameters are provided, defaults to ""

- **offset** (*int, optional*) – Pagination offset, defaults to 0
- **limit** (*int, optional*) – Pagination limit with Max 20, defaults to 20

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

get_template_text(*conn*, *group_name*, *template_name*)

Get CLI template in text format.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of an existing group
- **template_name** (*str*) – Name of an existing template within mentioned group

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

update_template(*conn*, *group_name*, *template_name*, *template_filename*, *device_type=''*, *version=''*, *model=''*)

[summary]

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of the group in which the template file exists.
- **template_name** (*str*) – Name of the template to be modified
- **template_filename** (*str*) – Name of the template file in local machine to be sent to central using API
- **device_type** (*str, optional*) – Aruba device type of the template , defaults to ""
- **version** (*str, optional*) – Firmware version property of template., defaults to ""
- **model** (*str, optional*) – Device model property of template. For 'ArubaSwitch' device_type, part number (J number) can be used, defaults to ""

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

class pycentral.configuration.**Variables**

Bases: object

A python class consisting of functions to manage Aruba Central Variables via REST API

create_template_variables(*conn*, *device_serial*, *variables*)

Create template variable for a device

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial Number of a device for which the variables are to be created.
- **variables** (*dict*) – Variables defined in template file to be applied for a device. Sample Variables -

  {"_sys_serial": "AB0011111", "_sys_lan_mac": "11:12:AA:13:14:BB", "SSID_A": "Z-Employee"}

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

create_template_variables_file(*conn*, *variables_filename*, *format='JSON'*)

Create template variables for multiple devices defined in JSON format in a file

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **variables_filename** (*str*) – Name of the variable file to be sent to Central via API.
- **format** (*str, optional*) – Format of data in the file to be uploaded, defaults to "JSON"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

delete_template_variables(*conn*, *device_serial*)

Delete all existing template variables for a device

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of a device

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

get_all_template_variables(*conn*, *offset=0*, *limit=20*, *format='JSON'*)

Get template variables for all devices

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

- **offset** (*int, optional*) – Pagination offset. Number of items to be skipped before returning the data, useful for pagination, defaults to 0
- **limit** (*int, optional*) – Pagination limit. Maximum number of records to be returned, defaults to 20
- **format** (*str, optional*) – Format in which output is desired, defaults to "JSON"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

get_template_variables(*conn*, *device_serial*)

Get template variables for a device

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of the device.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

replace_template_variables(*conn*, *device_serial*, *variables*)

Delete all existing template variables and create requested template variables for a device. This API can be used for deleting some variables out of all for a device.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of a device
- **variables** (*dict*) – Delete existing template variables for a device and replace with the new variables.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

replace_template_variables_file(*conn*, *variables_filename*, *format='JSON'*)

Delete all existing template variables and create requested template variables for all devices. This API can be used for deleting some variables out of all for all devices.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **variables_filename** (*str*) – filename of template variables to be uploaded to Central via API from local machine.
- **format** (*str, optional*) – Data format of the specified file, defaults to "JSON"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`update_template_variables`(*conn*, *device_serial*, *variables*)

Update values of existing template variables and add new variables to the existing set of variables for a device.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Serial number of a device
- **variables** (*dict*) – Template variables to be updated for the mentioned device

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`update_template_variables_file`(*conn*, *variables_filename*)

Update values of existing template variables and add new variables to the existing set of variables for multiple devices defined in the specified file.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **variables_filename** (*str*) – Local filename of template variables to be uploaded to Central via API.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*.

Return type
dict

`class` pycentral.configuration.**Wlan**

Bases: `object`

A python class consisting of functions to manage Aruba Central WLANs via REST API. This class uses WLAN APIs that have to be allowlisted for the Aruba Central account

`create_full_wlan`(*conn*, *group_name*, *wlan_name*, *wlan_data*)

Create new WLAN using the full_wlan endpoint "/configuration/full_wlan/". Used for complex configurations not supported by create_wlan.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of Aruba Central group to create new WLAN inside.
- **wlan_name** (*str*) – Name string for new WLAN
- **wlan_data** (*dict(json)*) – Data to create new WLAN

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

create_wlan(*conn*, *group_name*, *wlan_name*, *wlan_data*)

Create new WLAN.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of Aruba Central group to create new WLAN inside.
- **wlan_name** (*str*) – Name string for new WLAN
- **wlan_data** (*dict(json)*) – Data to create new WLAN * keyword essid: SSID name * keyword type: type designation for new SSID * keyword hide_ssid: boolean to hide SSID * keyword vlan: vlan to add new SSID to * keyword zone: vlan zones to add SSID to * keyword opmode: SSID security opmode * keyword wpa_passphrase: Passphrase for SSID * keyword wpa_passphrase_changed: Should always be set true * keyword is_locked: Boolean * keyword captive_profile_name: name for users using captive portal * keyword bandwidth_limit_up: mb up * keyword bandwidth_limit_down: mb down * keyword bandwidth_limit_peruser_up: peruser mb up * keyword bandwidth_limit_peruser_down: peruser mb down * keyword access_rules: dict

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

delete_wlan(*conn*, *group_name*, *wlan_name*)

Delete an existing WLAN.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Group name of the group or guid of the swarm or serial number of 10x AP.
- **wlan_name** (*str*) – Name of WLAN to delete.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

disable_wlan(*conn*, *group_name*, *wlan_name*)

This function will disable the WLAN for client connections. Current connected clients will be disconnected from this WLAN.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of Aruba Central UI group which has the WLAN

- **wlan_name** (*str*) – Name of WLAN which has to be disabled

enable_wlan(*conn*, *group_name*, *wlan_name*)

This function will enable the WLAN for client connections.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Name of Aruba Central UI group which has the WLAN
- **wlan_name** (*str*) – Name of WLAN which has to be enabled

get_all_wlans(*conn*, *group_name*)

Gets a list of each wlan in a Central group.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Group name of the group or guid of the swarm.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

get_wlan(*conn*, *group_name*, *wlan_name*)

Gets configuration of a WLAN in a Central group.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Group name of the group or guid of the swarm.
- **wlan_name** (*str*) – Name string for wlan to get.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

update_full_wlan(*conn*, *group_name*, *wlan_name*, *wlan_data*)

Update an existing WLAN using the full_wlan endpoint "/configuration/full_wlan/". Used for complex configurations not supported by update_wlan.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Group name of the group or guid of the swarm or serial number of 10x AP
- **wlan_name** (*str*) – Name string for new WLAN
- **wlan_data** (*dict(json)*) – Data to update existing wlan.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

update_wlan(*conn*, *group_name*, *wlan_name*, *wlan_data*)

Update an existing WLAN.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – Group name of the group or guid of the swarm or serial number of 10x AP
- **wlan_name** (*str*) – Name string for new WLAN
- **wlan_data** (*dict(json)*) – Data to update existing wlan.

Returns
Response as provided by 'command' function in class: *pycentral.ArubaCentralBase*.

Return type
dict

4.2.5 pycentral.device_inventory module

class pycentral.device_inventory.**Inventory**

Bases: object

A python class consisting of functions to manage Aruba Central inventory from the new device inventory category via REST API.

add_devices(*conn*, *device_details*)

Add device(s) using Mac & Serial Numbers

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_details** (*list*) – List of dictionaries with Aruba devices that should be added to the Aruba Central account. Each item in the dictionary should have the following keys - *mac* & *serial*. For Central On-Premises account, an additional key is required in the dictionary - *partNumber*

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

archive_devices(*conn*, *device_serials=[]*)

Archive a list of devices using serial numbers

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serials** (*list*) – List of serial number of Aruba devices that should be archived

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_inventory`(*conn*, *sku_type='all'*, *limit=0*, *offset=0*)

Get device details from inventory.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase*.
- **sku_type** (*str*) – target device sku type to pull from inventory. Acceptable arguments: all, iap, switch, controller, gateway, vgw, cap, boc, all_ap, all_controller, others.
- **limit** (*int, optional*) – Pagination limit. Defaults to 0, which is intrepreted as get all. Maximum limit per request is 50.
- **offset** (*int, optional*) – Pagination offset, defaults to 0.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`unarchive_devices`(*conn*, *device_serials=[]*)

Unarchive a list of devices using serial numbers

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serials** (*list*) – List of serial number of Aruba devices that should be unarchived

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.6 pycentral.firmware_management module

`class pycentral.firmware_management.Firmware`

Bases: `object`

A Python Class to manage Aruba Central Device Firmware via REST APIs.

`cancel_scheduled_upgrade`(*conn*, *serial=None*, *swarm_id=None*, *device_type=None*, *group=None*)

Cancel scheduled firmware upgrade for a device or for a whole group of devices. To cancel scheduled upgrade for certain type of devices of specific group, specify device_type as one of "IAP" for swarm, "MAS" for MAS switches, "HP" for aruba switches, "CONTROLLER" for controllers respectively, and the group as the group name. To cancel scheduled upgrade a device, you can specify swarm_id of the specific swarm or serial of the specific device.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **serial** (*str, optional*) – Serial of device, defaults to None
- **swarm_id** (*str, optional*) – Swarm ID, defaults to None
- **device_type** (*str, optional*) – Specify one of "IAP/MAS/HP/CONTROLLER", defaults to None
- **group** (*str, optional*) – Specify Group Name to initiate upgrade for whole group, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

check_firmware_status(*conn*, *serial=None*, *swarm_id=None*)

Get firmware upgrade status of device. You can either specify swarm_id if device_type is "IAP" or serial for rest of device_type, but not both.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **serial** (*str, optional*) – Serial of device, defaults to None
- **swarm_id** (*str, optional*) – Swarm ID, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

check_firmware_support(*conn*, *firmware_version*, *device_type*)

Check whether specific firmware version is available or not. Specify device_type "IAP"/"MAS"/"HP"/"CONTROLLER" to get firmware versions for swarms/MAS switches/aruba switches/controllers respectively.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **firmware_version** (*str*) – Firmware Version
- **device_type** (*str*) – Specify one of "IAP/MAS/HP/CONTROLLER"

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_firmware_swarm(*conn*, *swarm_id*)

Get firmware details for specific swarm.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

- **swarm_id** (*str*) – Swarm ID for which the firmware detail to be queried

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_firmware_all_swarms(*conn*, *group=None*, *limit=20*, *offset=0*)

Get a list of swarms with their firmware details. You can optionally specify a group, to filter devices under it

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*str, optional*) – Name of the group, defaults to None
- **limit** (*int, optional*) – Pagination limit. Default is 20 and max is 1000, defaults to 20
- **offset** (*int, optional*) – Pagination offset, defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_supported_version(*conn*, *device_type=None*, *swarm_id=None*, *serial=None*)

Get list of firmware versions for device. Specify device_type "IAP"/"MAS"/"HP"/"CONTROLLER" to get firmware versions for swarms, MAS switches, aruba switches and controllers respectively or you can specify swarm_id to get list of supported version for specific swarm or you can specify serial to get list of supported version for specific device.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_type** (*str, optional*) – Specify one of "IAP/MAS/HP/CONTROLLER", defaults to None
- **swarm_id** (*str, optional*) – Swarm ID, defaults to None
- **serial** (*str, optional*) – Serial of device, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

upgrade_firmware(*conn*, *firmware_version*, *reboot=True*, *device_type=None*, *model=None*, *group=None*, *serial=None*, *swarm_id=None*, *schedule_at=None*)

Upgrade firmware version for a device or for a whole group of devices under a device type, with additional filter of model. To initiate upgrade for certain type of devices of specific group, specify device_type as one of "IAP" for swarm, "MAS" for MAS switches, "HP" for aruba switches, "CONTROLLER" for controllers respectively, and group name. To upgrade a device, you can specify swarm_id to upgrade a specific swarm or serial to upgrade a specific device. To upgrade a specific model of Aruba switches at group level, please use model in request body.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **firmware_version** (*str*) – Specify firmware version to which you want device to upgrade. If you do not specify this field then firmware upgrade initiated with recommended firmware version
- **reboot** (*bool, optional*) – Use True for auto reboot after successful firmware download. Default value is False. Applicable only on MAS, Aruba switches and controller since IAP reboots automatically after firmware download., defaults to True
- **device_type** (*str, optional*) – Specify one of "IAP/MAS/HP/CONTROLLER", defaults to None
- **model** (*str, optional*) – To initiate upgrade at group level for specific model family. Applicable only for Aruba switches, defaults to None
- **group** (*str, optional*) – Specify Group Name to initiate upgrade for whole group., defaults to None
- **serial** (*str, optional*) – Serial of device, defaults to None
- **swarm_id** (*str, optional*) – Swarm ID, defaults to None
- **schedule_at** (*int, optional*) – Firmware upgrade will be scheduled at, *firmware_scheduled_at* - current time. firmware_scheduled_at is epoch in seconds and default value is current time, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.7 pycentral.licensing module

class pycentral.licensing.**AutoLicense**

Bases: object

A python class to manage auto-licenses for Aruba Central

assign_autolicense_services(*conn*, *services*)

This function is used to assign licenses to all devices for given services and enable auto licensing.

Note: This API is not applicable for MSP customer

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

assign_msp_autolicense_services(*conn*, *services*, *include_customers=[]*, *exclude_customers=[]*)

Enable auto license settings for MSP and Tenants. Assign licenses for given services to all the devices owned by tenant customers.

Note - License assignment is not supported for the MSP owned devices. License assignment will be in paused state if the total license tokens are less than total device counts(including MSP and tenants)

Note: If include_customers and exclude_customers are not provided then license settings will be enabled for all customers i.e MSP, tenants and future tenants(Note: Newly created tenant will be inherited license settings from MSP)

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.
- **include_customers** (*list, optional*) – if provided, license settings will be enabled only for the customers present in include_customers list., defaults to []
- **exclude_customers** (*list, optional*) – if provided, license settings will be enabled for customers except the customers present in exclude_customers list, defaults to []

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

disable_autolicensing_services(*conn*, *services*)

This function is used to disable auto licensing. Note: This API is not applicable for MSP customer

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

disable_msp_autolicense_services(*conn*, *services*, *include_customers=[]*, *exclude_customers=[]*)

Disable auto license settings for MSP and Tenants for the given services. This will not change the current license device mapping

Note: If include_customers and exclude_customers are not provided then auto license setting will be disabled for all customers i.e MSP and tenants.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

- **include_customers** (*list, optional*) – if provided, licensing will be disable only for the customers present in include_customers list, defaults to []
- **exclude_customers** (*list, optional*) – if provided, licensing will be disabled for the customers except the customers present in exclude_customers list, defaults to []

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_autolicense_services(*conn*)

This function is used to get services which are auto enabled.

Parameters
conn (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_license_status(*conn*, *service_name: str*)

Get services and corresponding license token availability status. If True, license tokens are more than device count else less than device count.(Note - Autolicense is in paused state when license tokens are less than device count)

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **service_name** (*str*) – Specific service name(dm/pa/..). Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_msp_autolicense_services(*conn*, *customer_id: str*)

[summary]

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_id** (*str*) – Customer id of msp or tenant.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

class pycentral.licensing.**Subscriptions**

Bases: object

A python class to manage subscriptions for Aruba Central

assign_device_subscription(*conn*, *device_serials*, *services*)

This function is used to assign subscriptions to device by specifying its serial.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serials** (*list*) – List of serial number of device.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

assign_msp_subscription_all(*conn*, *services*, *include_customers=[]*, *exclude_customers=[]*)

Assign licenses to all devices owned by tenant customers for given services. If include_customers and exclude_customers parameters are not provided, licenses will be assigned for all customers(MSP, tenants) devices.

Note: License assignment is not supported for the MSP owned devices. Since it is a background job, please wait for few minutes for all devices to be subscribed in case of customer having large number of devices

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.
- **include_customers** (*list, optional*) – if provided, licenses will be assigned only for the customers present in include_customers list

  (Exception: License assignment will be ignored for MSP owned devices), default=[]
- **exclude_customers** (*list, optional*) – if provided, licenses will be assigned for MSP/tenant customers except the customers present in exclude_customers list, default=[]

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

assign_subscription_all(*conn*, *services*)

This function is used to assign licenses to all devices for given services.

Note: This API is not applicable for MSP customer

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_enabled_services`(*conn*)

This function is used to get the list of services which are enabled for customer.

Parameters
conn (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_services_license_config`(*conn*, *service_category=''*, *device_type=''*)

This function is used to return services configuration for licensing purpose.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **service_category** (*str, optional*) – Service category - dm/network, defaults to ""
- **device_type** (*str, optional*) – Device Type - iap/switch, defaults to ""

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_user_subscription_keys`(*conn*, *license_type=''*, *offset=0*, *limit=100*)

This function is used to get license subscription keys

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **license_type** (*str, optional*) – Accepts basic/special, defaults to ""
- **offset** (*int, optional*) – Pagination offset, defaults to 0
- **limit** (*int, optional*) – Number of subscriptions to get, defaults to 100

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_user_subscription_status`(*conn*, *license_type='all'*, *service=''*)

This function is used to return subscription stats.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **license_type** (*str, optional*) – basic/special/all. special - will fetch the statistics of special central services like presence analytics(pa), ucc, clarity etc basic - will fetch the

statistics of device management service licenses, all - will fetch both of these license types, defaults to "all"

- **service** (*str, optional*) – Service type: pa/pa,clarity etc, defaults to ""

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

unassign_device_subscription(*conn*, *device_serials*, *services*)

This function is used to unassign subscriptions to device by specifying its serial.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serials** (*list*) – List of serial number of device.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

unassign_msp_subscription_all(*conn*, *services*, *include_customers=[]*, *exclude_customers=[]*)

Remove service licenses to all devices the devices owned by tenant and MSP. However license assignment is not supported for the MSP owned devices but un-assignment is supported for the customers who are transitioning from Non-MSP to MSP mode to release license quantity for better utilization.

Note: If include_customers and exclude_customers parameters are not provided, licenses will be unassigned for all customers(MSP, tenants) devices.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.
- **include_customers** (*list, optional*) – if provided, licenses will be unassigned only for the customers present in include_customers list.

  (Exception: License assignment will be ignored for MSP owned devices), default=[]
- **exclude_customers** (*list, optional*) – if provided, licenses will be unassigned for MSP/tenant customers except the customers present in exclude_customers list, default=[]

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

unassign_subscription_all(*conn*, *services*)

This function is used to unassign licenses to all devices for given services.

Note: This API is not applicable for MSP customer

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **services** (*list*) – List of service names. Call services/config API to get the list of valid service names.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.8 pycentral.monitoring module

class pycentral.monitoring.**Sites**

Bases: object

A python class consisting of functions to manage Aruba Central Sites via REST API

associate_devices(*conn*, *site_id*, *device_type*, *device_ids*)

Associate multiple devices to a site

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_id** (*int*) – ID assigned by Aruba Central when the site is created. Can be obtained from find_site_id function.
- **device_type** (*str*) – Type of the device. One of the "IAP", "ArubaSwitch", "CX", "MobilityController".
- **device_ids** (*list*) – List of Aruba devices' serial number

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

create_site(*conn*, *site_name*, *site_address={}*, *geolocation={}*)

Creates a new site

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_name** (*str*) – Name of the site be created.
- **site_address** (*dict, optional*) – Address of the site, defaults to {}
  - keyword address: Site address string
  - keyword city: City name string
  - keyword state: State name string
  - keyword country: Country name string
  - keyword zipcode: Zipcode string

- **geolocation** (*dict, optional*) – Mutually exclusive with site address. Provide either one option, defaults to {}
  - keyword latitude: Site location latitude in the world map
  - keyword longitude: Site location longitude in the world map

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

delete_site(*conn*, *site_id*)

Delete an existing site

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_id** (*int*) – ID assigned by Aruba Central when the site is created. Can be obtained from find_site_id function.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

find_site_id(*conn*, *site_name*)

Find site id from site name

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_name** (*str*) – Name of the site be created.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_sites(*conn*, *calculate_total=False*, *offset=0*, *limit=100*, *sort='+site_name'*)

Get list of sites

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **calculate_total** (*bool, optional*) – Whether to calculate total number of sites, defaults to False
- **offset** (*int, optional*) – Pagination offset, defaults to 0
- **limit** (*int, optional*) – Pagination limit with Max 1000, defaults to 100
- **sort** (*str, optional*) – Sort list of sites based on one of '+site_name', '-site_name', defaults to "+site_name"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

unassociate_devices(*conn*, *site_id*, *device_type*, *device_ids*)

Unassociate a device from a site

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_id** (*int*) – ID assigned by Aruba Central when the site is created. Can be obtained from find_site_id function.
- **device_type** (*str*) – Type of the device. One of the "IAP", "ArubaSwitch", "CX", "MobilityController".
- **device_id** (*str*) – Aruba device serial number

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

update_site(*conn*, *site_id*, *site_name*, *site_address={}*, *geolocation={}*)

Update/Modify an existing site

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_id** (*int*) – ID assigned by Aruba Central when the site is created. Can be obtained from find_site_id function.
- **site_name** (*str*) – Name of the site be created.
- **site_address** (*dict, optional*) – Address of the site, defaults to {}
  - keyword address: Site address string
  - keyword city: City name string
  - keyword state: State name string
  - keyword country: Country name string
  - keyword zipcode: Zipcode string
- **geolocation** (*dict, optional*) – Mutually exclusive with site address. Provide either one option, defaults to {}
  - keyword latitude: Site location latitude in the world map
  - keyword longitude: Site location longitude in the world map

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.9 pycentral.msp module

class pycentral.msp.**MSP**

Bases: object

A python class consisting of functions to manage Aruba Central's MSP mode via REST API

assign_devices_to_customers(*conn*, *devices*, *group_name=None*, *customer_id=None*, *customer_name=None*)

This function assign devices to customer

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **devices** (*list*) – List of dictionaries of devices that will be assigned to the customer account. Each dictionary corresponds to a device & will have the following keys - serial, mac
- **group_name** (*str, optional*) – Name of the group to which the devices will be moved to within the customer.
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

create_customer(*conn*, *customer_details*)

This function creates a customer in the MSP account based on the provided customer details

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_details** (*dict*) – Details of customer that has to be created. The customer details should have the following the keys - customer_name, country_name, street_address, city, state, country_name, zip_postal_code. These keys are optional - lock_msp_ssids, description, group_name

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

delete_customer(*conn*, *customer_id=None*, *customer_name=None*)

This function deletes the customer in the MSP account

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.

- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

edit_msp_resources(*conn*, *resources_dict*)

This function edits the branding resources under an MSP account

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **resources_dict** (*dict*) – Details of new branding resources of the MSP. This parameter's structure should match the structure of the sample API body in the Swagger.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_all_customers(*conn*)

This function returns a list of all the customers in the MSP account

Parameters
conn (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

Returns
Returns list of dictionaries. Each dictionary has the following keys associated with a customer - account_status, account_type, ap_config_diff, application_id, application_instance_id, created_at, customer_id, customer_name, description, device_quota, hppc_config_diff, lock_msp_ssids, msp_conversion_status, msp_id, platform_customer_details, platform_customer_id, provision_status, region, switch_config_diff, updated_at, username

Return type
list

get_country_code(*conn*, *country_name*)

This function fetches the country code of a country. This country code is needed for the Create Customer API

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **country_name** (*str*) – Name of country

Returns
Country Code of country. It will return None if no country is found

Return type
string

get_country_codes_list(*conn*)

This function fetches the dictionary of the country codes of countries. The keys of the dictionary are the country names and the values are the country codes

Parameters
conn (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_customer_details(*conn*, *customer_id=None*, *customer_name=None*)

This function fetches the details of the customer in the MSP account

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_customer_devices_and_subscriptions(*conn*, *customer_id=None*, *customer_name=None*, *offset=0*, *limit=10*, *device_type=None*)

This function gets the devices & subscriptions under the customer account based on the provided parameters

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed
- **offset** (*int, optional*) – Pagination start index, defaults to 0
- **limit** (*int, optional*) – Pagination end index, defaults to 10
- **device_type** (*str, optional*) – Filter on device_type. Accepted values - iap, switch, all_controller. Defaults to None.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_customer_id(*conn*, *customer_name=None*)

This function fetches the customer id of the customer based on the customer name.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Customer ID of the customer. It will return None if no customer is found

Return type
string

get_customer_users(*conn*, *offset=0*, *limit=10*, *customer_id=None*, *customer_name=None*)

This function returns the list of users under a customer in the MSP account based on the provided parameters

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **offset** (*int, optional*) – Pagination start index, defaults to 0
- **limit** (*int, optional*) – Pagination end index, defaults to 10
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_customers(*conn*, *offset=0*, *limit=100*, *customer_name=None*)

This function returns the list of customers based on the provided parameters

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **offset** (*int, optional*) – Pagination start index, defaults to 0
- **limit** (*int, optional*) – Pagination end index, defaults to 100
- **customer_name** (*str, optional*) – Filter on customer name, defaults to None.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_customers_per_group(*conn*, *group_name*, *offset=0*, *limit=10*)

This function fetches the list of customers to MSP group based on the provided parameters.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group_name** (*str*) – MSP group name
- **offset** (*int, optional*) – Pagination start index, defaults to 0
- **limit** (*int, optional*) – Pagination end index, defaults to 10

Returns
List of device & licenses in the MSP or customer account

Return type
list

`get_msp_all_devices_and_subscriptions`(*conn*, *customer_name=None*)

This function fetches all the devices & subscriptions from a MSP account. If the customer_name parameter is passed, then it will return all the devices & licenses in the customer account.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
List of device & licenses in the MSP or customer account

Return type
list

`get_msp_devices_and_subscriptions`(*conn*, *offset=0*, *limit=10*, *device_allocation_status=0*, *device_type=None*, *customer_name=None*)

This function fetches the list of devices & licenses under the MSP account based on the provided parameters.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **offset** (*int, optional*) – Pagination start index, defaults to 0
- **limit** (*int, optional*) – Pagination end index, defaults to 100
- **device_allocation_status** (*str, optional*) – Filter on device_allocation_status. This parameter accepts the following values - 0(All), 1(Allocated),2(Available). Defaults to 0.
- **device_type** (*str, optional*) – Filter on device_type. Accepted values - iap, switch, all_controller. Defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_msp_id`(*conn*)

This function fetches the MSP ID of the MSP.

Parameters
conn (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

Returns
MSP ID of the MSP account. It will return None if no ID is found

Return type
string

`get_msp_resources`(*conn*)

This function returns the branding resources under an MSP account

Parameters
conn (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_msp_users`(*conn*, *offset=0*, *limit=10*)

This function returns the list of users under the MSP account based on the provided parameters

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **offset** (*int, optional*) – Pagination start index, defaults to 0
- **limit** (*int, optional*) – Pagination end index, defaults to 10

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`unassign_all_customer_device`(*conn*, *customer_id=None*, *customer_name=None*)

This function unassigns all devices & subscriptions from a customer's Central Instance. It will move these devices & subsciptions to the MSP's device & subscription inventory.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`unassign_devices_from_customers`(*conn*, *devices*, *msp_id=None*)

This function unassign devices from the customer to the MSP's device inventory

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **devices** (*list*) – List of dictionaries of devices that will be assigned to the customer account. Each dictionary corresponds to a device & will have the following keys - serial, mac

- **msp_id** (*str, optional*) – ID of the MSP account. If no ID is provided, then the the msp_id will be fetched with the get_msp_id function

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

update_customer(*conn*, *customer_details*, *customer_id=None*, *customer_name=None*)

This function updates the details of an existing customer in the MSP account

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **customer_details** (*dict*) – Details of customer that has to be updated. The customer details should have the following the keys - customer_name, country_name, street_address, city, state, country_name, zip_postal_code. These keys are optional - lock_msp_ssids, description, group_name
- **customer_id** (*str, optional*) – Customer ID of the customer, defaults to None.
- **customer_name** (*str, optional*) – Name of customer, defaults to None. This parameter will be ignored if customer_id parameter is passed

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.10 pycentral.rapids module

class pycentral.rapids.**Rogues**

Bases: object

A Python class to obtain Aruba Central's Rougue details via REST APIs.

list_interfering_aps(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *offset=0*)

Get interfering APs over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None
- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None

- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None
- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_neighbor_aps(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *offset=0*)

Get neighbor APs over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None
- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None
- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None
- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_rogue_aps(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *offset=0*)

Get rogue APs over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None
- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None
- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None
- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_suspect_aps(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *offset=0*)

Get suspect APs over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None
- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None
- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None

- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

class pycentral.rapids.**WIDS**

Bases: object

A Python Class to obtain Aruba Central's Wireless Intrusion Detection details based on REST APIs.

list_client_attacks(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *calculate_total=True*, *sort='-ts'*, *offset=0*)

Get client attacks over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None
- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None
- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None
- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **calculate_total** (*bool, optional*) – Whether to calculate total client attacks, defaults to True
- **sort** (*str, optional*) – Sort parameter -ts(sort based on the timestamps in descending), +ts(sort based on timestamps ascending), -macaddr (sort based on station mac descending) and +macaddr(sort based station mac ascending), defaults to "-ts"
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_infrastructure_attacks(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *calculate_total=True*, *sort='-ts'*, *offset=0*)

Get infrastructure attacks over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None
- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None
- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None
- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **calculate_total** (*bool, optional*) – Whether to calculate total client attacks, defaults to True
- **sort** (*str, optional*) – Sort parameter -ts(sort based on the timestamps in descending), +ts(sort based on timestamps ascending), -macaddr (sort based on station mac descending) and +macaddr(sort based station mac ascending), defaults to "-ts"
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_wids_attacks(*conn*, *group=None*, *label=None*, *site=None*, *swarm_id=None*, *start=None*, *end=None*, *from_timestamp=None*, *to_timestamp=None*, *limit=100*, *sort='-ts'*, *offset=0*)

Get WIDS events over a time period

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **group** (*list, optional*) – List of group names, defaults to None
- **label** (*list, optional*) – List of label names, defaults to None
- **site** (*list, optional*) – List of site names, defaults to None
- **swarm_id** (*str, optional*) – Filter by Swarm ID, defaults to None

- **start** (*int, optional*) – Need information from this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp minus 3 hours, defaults to None
- **end** (*int, optional*) – Need information to this timestamp. Timestamp is epoch in milliseconds. Default is current timestamp, defaults to None
- **from_timestamp** (*int, optional*) – This parameter supercedes start parameter. Need information from this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp minus 3 hours, defaults to None
- **to_timestamp** (*int, optional*) – This parameter supercedes end parameter. Need information to this timestamp. Timestamp is epoch in seconds. Default is current UTC timestamp, defaults to None
- **limit** (*int, optional*) – pagination size (default = 100), defaults to 100
- **sort** (*str, optional*) – Sort parameter -ts(sort based on the timestamps in descending), +ts(sort based on timestamps ascending), -macaddr (sort based on station mac descending) and +macaddr(sort based station mac ascending), defaults to "-ts"
- **offset** (*int, optional*) – Pagination offset (default = 0), defaults to 0

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.11 pycentral.refresh_api_token module

class pycentral.refresh_api_token.**RefreshApiToken**

Bases: object

Refresh the API access token in API Gateway using OAUTH API

refresh_token(*conn*, *apigw_client_id*, *apigw_client_secret*, *old_refresh_token*)

This function refreshes the existing access token and replaces old token with new token. The returned token dict will contain both access and refresh token. Use refresh token provided in the return dict for next refresh.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **apigw_client_id** (*str*) – Client ID from API Gateway page
- **apigw_client_secret** (*str*) – Client Secret from API Gateway page
- **old_refresh_token** (*str*) – Refresh token value from the current/expired API token.

Returns
Refrehed token dict consisting of access_token and refresh_token.

Return type
dict

4.2.12 pycentral.topology module

class pycentral.topology.**Topology**

Bases: object

A python class to obtain Aruba Central Site's topology details via REST APIs.

ap_lldp_neighbors(*conn*, *device_serial*)

Get neighbor details reported by AP via LLDP.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Device serial number.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_device_details(*conn*, *device_serial*)

Provides details of a device when serial number is passed as input.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **device_serial** (*str*) – Device Serial Number

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_edge_details(*conn*, *source_serial*, *dest_serial*)

Get details of an edge grouped by lagname. The serials of nodes/devices on both sides of the edge should passed as input.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **source_serial** (*str*) – Device serial number.
- **dest_serial** (*str*) – Device serial number.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_topology(*conn*, *site_id*)

Get topology details of a site. The input is the id corresponding to a label or a site.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_id** (*int*) – Site ID

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_uplink_details(*conn*, *source_serial*, *uplink_id*)

Get details of an uplink. The serials of node/device on one side of the uplink and the uplink id of the uplink should passed as input.Desired uplink id can be found in get topology details api.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **source_serial** (*str*) – Device serial number.xx
- **uplink_id** (*str*) – Uplink id.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

tunnel_details(*conn*, *site_id*, *tunnel_map_names*)

Get tunnel details.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **site_id** (*int*) – Site ID
- **tunnel_map_names** (*list*) – Comma separated list of tunnel map names.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.13 pycentral.user_management module

class pycentral.user_management.**Roles**

Bases: object

A Python class to manage Aruba Central User Roles via REST APIs.

create_user_role(*conn*, *app_name*, *rolename*, *applications*, *permission='modify'*)

Create an user role in an app

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.

- **app_name** (*str*) – app name where role needs to be created
- **rolename** (*str*) – name of the role
- **applications** (*dict*) – List of dict. Each element containing the following structure.
  - keyword appname: Name of the application. Example: 'nms', 'account_setting', etc.
  - **keyword modules: A list of dictionaries. Each element containing 'module_name'** and 'permission' for the modules.
  - keyword permission: permission for the app
- **permission** (*str, optional*) – permission of the role, defaults to "modify"

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

delete_user_role(*conn*, *app_name*, *rolename*)

Delete a role specified by role name

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **app_name** (*str*) – app name
- **rolename** (*str*) – User role name

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_user_role(*conn*, *app_name*, *rolename*)

Get Role details

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **app_name** (*str*) – app name
- **rolename** (*str*) – User role name

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_user_roles(*conn*, *app_name=None*, *limit=20*, *offset=0*, *order_by='+rolename'*)

Get list of all roles

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **app_name** (*str, optional*) – Filter users based on app_name, defaults to None

- **limit** (*int, optional*) – Maximum number of items to return, defaults to 20
- **offset** (*int, optional*) – Zero based offset to start from, defaults to 0
- **order_by** (*str, optional*) – Sort ordering. +rolename means ascending order of rolename, defaults to "+rolename"

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

update_user_role(*conn*, *app_name*, *rolename*, *applications*, *permission='modify'*)

Update a role specified by role name

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **app_name** (*str*) – app name
- **rolename** (*str*) – User role name
- **applications** (*dict*) – List of dict. Each element containing the following structure.
  - keyword appname: Name of the application. Example: 'nms', 'account_setting', etc.
  - keyword modules: A list of dictionaries. Each element containing 'module_name' and 'permission' for the modules.
  - keyword permission: permission for the app
- **permission** (*str, optional*) – [description], defaults to "modify"

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

class pycentral.user_management.**Users**

Bases: object

A Python class to manage Aruba Central Users via REST APIs.

create_user(*conn*, *username: str*, *password: str*, *description: str*, *name: dict*, *phone: str*, *address: dict*, *applications: dict*)

Create a user account. For public cloud environment, user has to re-register via invitation email. Email will be sent during processing of this request. Providing role on account setting app is mandatory in this API along with other subscribed apps. Scope must be given only for NMS app. For non-nms apps such as account setting refer the parameters in the example json payload.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **username** (*str*) – User's email id is specified as the user id
- **password** (*str*) – password of user account, defaults to None
- **description** (*str*) – description of user

- **name** (*dict*) – 'firstname' and 'lastname' of the user.
- **phone** (*str*) – Phone number. Format: +country_code-local_number
- **address** (*dict*) – Address of the user. Dict containing 'street', 'city', 'state', 'country' and 'zipcode'.
- **applications** (*dict*) – Define applications that needs access.
  - keyword name: Name of the application. Example: 'nms', 'account_setting', etc.
  - keyword info: A list of dictionaries. Each element containing 'role', 'tenant_role', and 'scope'. Where 'scope' contains 'groups' key with list of groups.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

delete_user(*conn*, *username*, *system_user=True*)

Delete user account details specified by user id

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **username** (*str*) – User's email id is specified as the user id
- **system_user** (*bool, optional*) – false if federated user, defaults to True

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_user(*conn*, *username*, *system_user=True*)

Get user account details specified by user email

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **username** (*str*) – User's email id is specified as the user id
- **system_user** (*bool, optional*) – false if federated user, defaults to True

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_users(*conn*, *limit=20*, *offset=0*, *order_by='+username'*, *app_name=None*, *user_type=None*, *status=None*)

Returns all users from the system associated to user's account

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **limit** (*int, optional*) – Maximum number of items to return, defaults to 20

- **offset** (*int, optional*) – Zero based offset to start from, defaults to 0
- **order_by** (*str, optional*) – Sort ordering (ascending or descending). +username signifies ascending order of username., defaults to "+username"
- **app_name** (*str, optional*) – Filter users based on app_name, defaults to None
- **user_type** (*str, optional*) – Filter based on system or federated user, defaults to None
- **status** (*str, optional*) – Filter user based on status (inprogress, failed), defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

list_users(*conn*, *limit=20*, *offset=0*, *sort='+timestamp'*, *email=None*)

(This API will be deprecated in future release). Use get_users(). Returns all users from the system associated to user's account

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **limit** (*int, optional*) – Maximum number of items to return, defaults to 20
- **offset** (*int, optional*) – Zero based offset to start from, defaults to 0
- **sort** (*str, optional*) – Sort ordering. One if +timestamp/-timestamp/+username/-username, defaults to "+timestamp"
- **email** (*str, optional*) – Filter users by email, defaults to None

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

update_user(*conn*, *username: str*, *description: str*, *name: dict*, *phone: str*, *address: dict*, *applications: dict*)

Update user account details specified by user id. Providing info on account setting app is mandatory in this API along with other subscribed apps.Scope must be given only for NMS app. For non-nms apps such as account setting refer the parameters in the example json payload.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **username** (*str*) – User's email id is specified as the user id
- **description** (*str*) – description of user
- **name** (*dict*) – 'firstname' and 'lastname' of the user.
- **phone** (*str*) – Phone number. Format: +country_code-local_number
- **address** (*dict*) – Address of the user. Dict containing 'street', 'city', 'state', 'country' and 'zipcode'.

- **applications** (*dict*) – Define applications that needs access.
  - keyword name: Name of the application. Example: 'nms', 'account_setting', etc.
  - keyword info: A list of dictionaries. Each element containing 'role', 'tenant_role', and 'scope'. Where 'scope' contains 'groups' key with list of groups.

Returns
HTTP Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.14 pycentral.visualrf module

class pycentral.visualrf.**ClientLocation**

Bases: object

A python class to obtain client location based on visualRF floor map.

get_client_location(*conn*, *macaddr: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get location of a client. This function provides output only when visualRF is configured in Aruba Central.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **macaddr** (*str*) – Provide a macaddr of a client. For example "ac:bb:cc:dd:ec:10"
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_floor_clients(*conn*, *floor_id: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get location of clients within a floormap in Aruba Central visualRF.

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **floor_id** (*str*) – Provide floor_id returned by *get_building_floors()* function in class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

class pycentral.visualrf.**FloorPlan**

Bases: object

A Python class to obtain information of floorplan in Aruba Central visualRF.

get_ap_location(*conn*, *ap_id: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get location of an access point within a floorplan

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **ap_id** (*str*) – Provide ap_id returned by *get_floor_aps()* within class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_building_floors(*conn*, *building_id: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get building info and floors within the building

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **building_id** (*str*) – Provide building id. Can be obtained from *get_campus_buildings* within class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_campus_buildings(*conn*, *campus_id: str*, *offset=0*, *limit=100*)

Get campus info and buildings within the campus

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **campus_id** (*str*) – Provide campus id. Can be obtained from *get_campus_list* function in class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_campus_list`(*conn*, *offset=0*, *limit=100*)

Get list of campuses in visualRF floorplan

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_floor_aps`(*conn*, *floor_id*, *offset=0*, *limit=100*, *units='FEET'*)

Get access points within a floor

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **floor_id** (*str*) – Provide floor id. Can be obtained from *get_building_floors()* within class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

`get_floor_image`(*conn*, *floor_id*, *offset=0*, *limit=100*)

Get Floor's background image in base64 format

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **floor_id** (*str*) – Provide floor id. Can be obtained from *get_building_floors()* within class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_floor_info(*conn*, *floor_id: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get floor information

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **floor_id** (*str*) – Provide floor id. Can be obtained from *get_building_floors()* within class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

class pycentral.visualrf.**RougueLocation**

Bases: object

A python class to obtain location of rogue access points

get_floor_rogueaps(*conn*, *floor_id: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get rogue access points within a floor

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **floor_id** (*str*) – Provide floor id. Can be obtained from *get_building_floors()* within class:*FloorPlan*
- **offset** (*int, optional*) – Pagination start index., defaults to 0
- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

get_rogueap_location(*conn*, *macaddr: str*, *offset=0*, *limit=100*, *units='FEET'*)

Get location of rogue a access point based on its Mac Address

Parameters

- **conn** (class:*pycentral.ArubaCentralBase*) – Instance of class:*pycentral.ArubaCentralBase* to make an API call.
- **macaddr** (*str*) – Provide Mac Address of an Access Point
- **offset** (*int, optional*) – Pagination start index., defaults to 0

- **limit** (*int, optional*) – Pagination size. Default 100 Max 100, defaults to 100
- **units** (*str, optional*) – METERS or FEET, defaults to "FEET"

Returns
Response as provided by 'command' function in class:*pycentral.ArubaCentralBase*

Return type
dict

4.2.15 Subpackages

pycentral.workflows package

pycentral.workflows.workflows_utils module

pycentral.workflows.workflows_utils.**dict_list_to_csv**(*filename*, *csv_data_list*, *logger=None*)

Write list of dictionaries into a CSV File via csv.DictWriter()

Parameters

- **filename** (*str*) – Name of the file to be created or overwritten
- **csv_data_list** (*list*) – A list of dictionaries, where each dict is a row in CSV file
- **logger** (class:*logging.logger*, optional) – Provide an instance of class:*logging.logger*.

pycentral.workflows.workflows_utils.**get_conn_from_file**(*filename*, *account=None*, *logger=None*)

Creates an instance of class`pycentral.ArubaCentralBase` based on the information provided in the YAML/JSON file.

- keyword central_info: A dict containing arguments as accepted by class`pycentral.ArubaCentralBase`
- keyword ssl_verify: A boolean when set to True, the python client validates Aruba Central's SSL certs.
- keyword token_store: Optional. Defaults to None.

Parameters

- **filename** (*str*) – Name of a JSON/YAML file containing the keywords required for class:*pycentral.ArubaCentralBase*
- **logger** (class:*logging.logger*, optional) – Provide an instance of class:*logging.logger*, defaults to logger class with name "ARUBA_BASE".

Returns
An instance of class:*pycentral.ArubaCentralBase* to make API calls and manage access tokens.

Return type
class:*pycentral.ArubaCentralBase*

pycentral.workflows.workflows_utils.**get_file_contents**(*filename*, *logger=None*)

Function to open a JSON/YAML/CSV file and return the contents of the file in dict format. (A list of dict is returned for a CSV file.)

Parameters

- **filename** (*str*) – Name of an existing JSON/YAML/CSV file.
- **logger** (class:*logging.logger*, optional) – Provide an instance of class:*logging.logger*.

Raises
`UserWarning` – Raises warning when supported filetypes are not provided.

Returns
Data loaded from JSON/YAML/CSV file

Return type
dict (a list of dict for CSV)

Module contents

CHAPTER

FIVE

LICENSE

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

INDEX

H

I

L

M

O

P

R

S

T

U

V

W