

The Cisco Live! logo, featuring the word "CISCO" in a dark blue, sans-serif font, followed by "Live!" in a dark blue, cursive script font.

CISCO *Live!*

The text "Let's go" in a large, dark blue, sans-serif font, positioned to the left of a bright white sunburst graphic that radiates across the right side of the image.

Let's go

#CiscoLiveAPJC



The bridge to possible

Sustainability with Meraki Dashboard APIs

A Deep Dive into Energy Management and
Optimization

Yuji Terada
DEVNET-1681

CISCO *Live!*

#CiscoLiveAPJC

Cisco Webex App

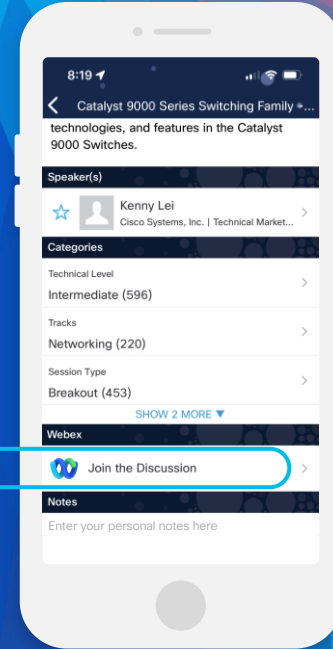
Questions?

Use Cisco Webex App to chat with the speaker after the session

How

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

Webex spaces will be moderated by the speaker until December 22, 2023.



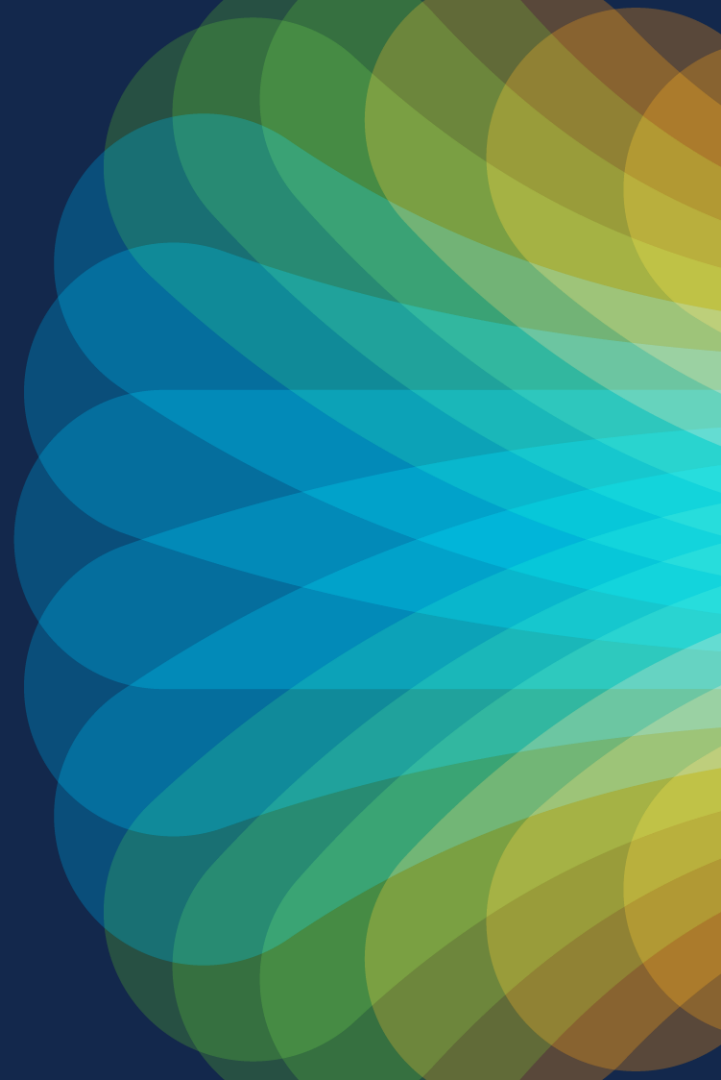
<https://ciscolive.ciscoevents.com/ciscolivebot/#DEVNET-1681>

Bitcoin

Agenda

- Overview
- Implementation
- Tips & tricks
- Call to action

Overview



Beginning

Do Wi-Fi 6E access points consume more power?



CDP/LLDP [Meraki MR56 - SFO12-2-AP01 / 0](#)(Meraki MR56 Cloud Managed Indoor AP) [raw](#)
PoE usage 8.5 W AT (Advertised 30 W AT)

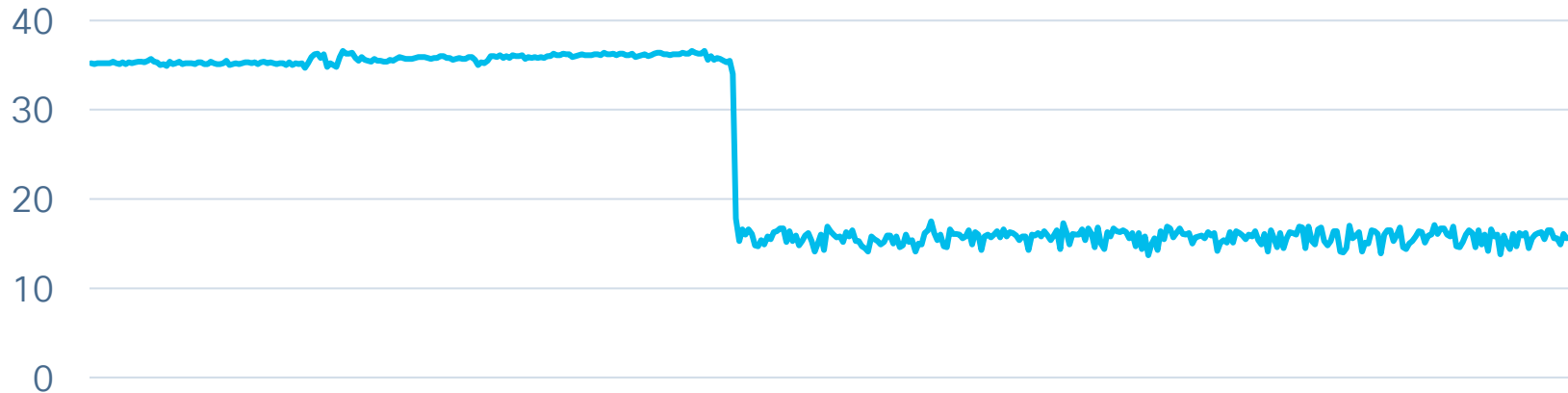


CDP/LLDP [SFO12-4-AP28.BETA_HW / 0](#)(Meraki CW9164I Cloud Managed AP) [raw](#)
PoE usage 12.3 W AT (Advertised 30 W AT)

They do! Wi-Fi 6E has an additional 6 GHz radio which the AP needs to power.

Beginning

Can I turn off certain devices to save energy?

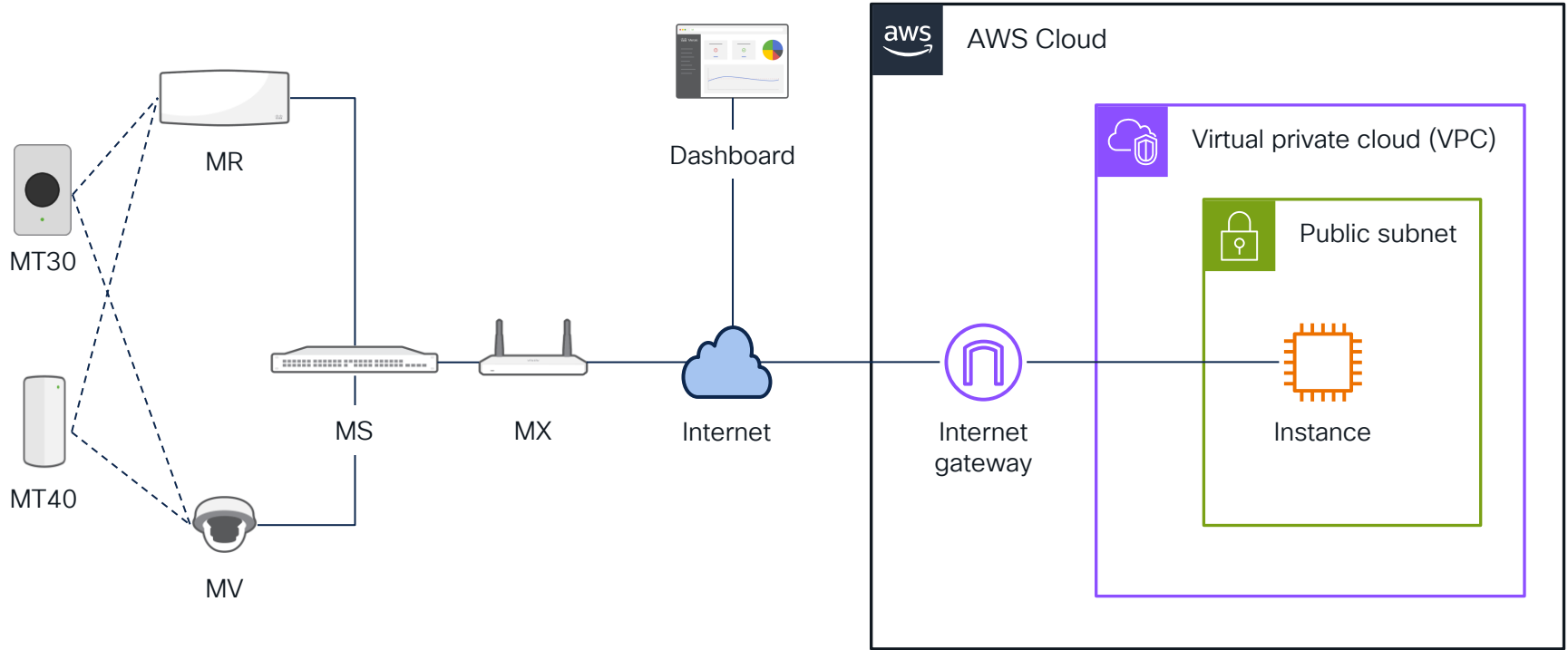


MT40 shows MS130-8X's power consumption decreases by 20 Watts when disabling a port which has CW9164I-MR connected.

Solution

1. Monitor usage
2. Turn devices off
3. Use MT30 to solve corner cases

Topology

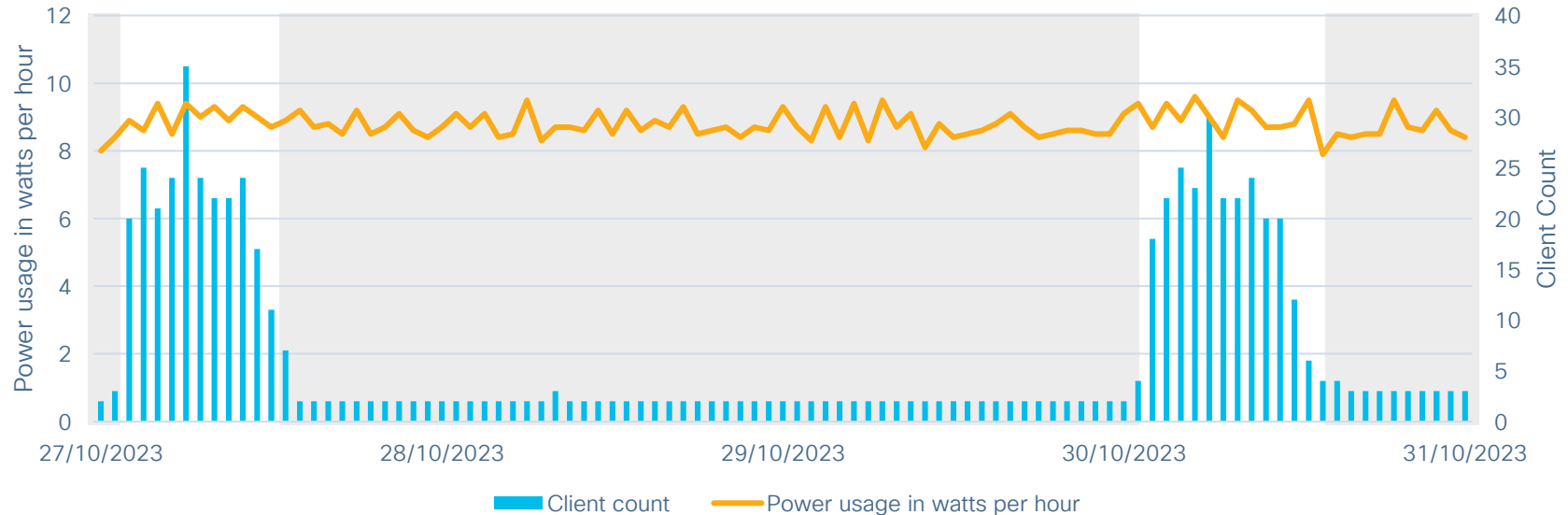


CISCO *Live!*

Data is important

No correlation between client count and power consumption for this network

Power consumption and client count

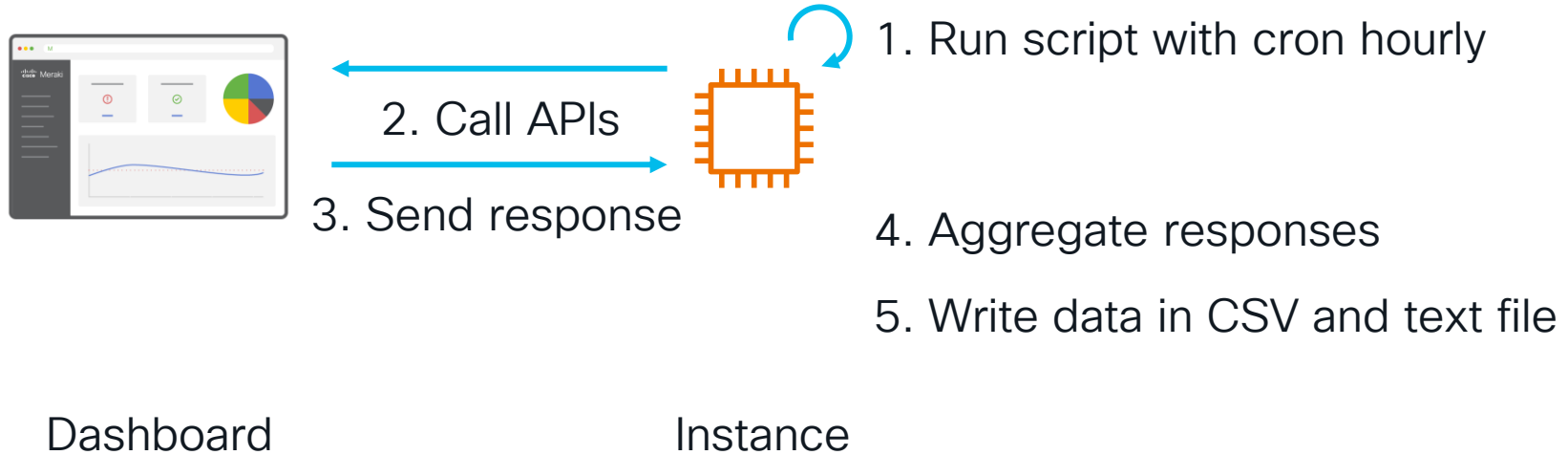


Implementation

Solution

1. Monitor usage
2. Turn devices off
3. Use MT30 to solve corner cases

Data gathering flow

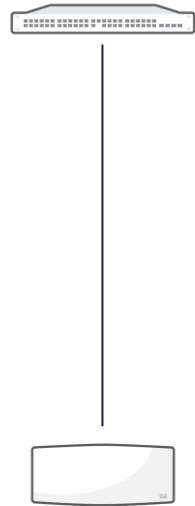


code: <https://gist.github.com/yujiterada/b636a5ffa65be067b9a1b313a079b2e4>

Row in CSV

1698354003, SB-SW-5F-01, Q4CD-U6SQ-XXXX, 1, 1, 9.2, SB-AP-5F-01,
Q5AC-D43S-XXXX, k74272e

Mapping switch and access point data



`getOrganizationDevices(productTypes=['switch'])`

```
{
  "serial": "Q4CD-U6SQ-XXXX",
  "model": "MS130-24X",
  "mac": "b8:ab:61:12:34:56",
  "name": "SB-SW-5F-01"
}
```

`getDeviceSwitchPortsStatuses(serial='Q4CD-U6SQ-XXXX')`

```
{
  "portId": "11",
  "cdp": {
    "deviceId": "cc9c3e123456",
    "address": "192.168.10.222"
  },
  "powerUsageInWh": 9.2,
  "usageInKb": { "total": 2, "sent": 1, "recv": 1 }
}
```

`getOrganizationDevices(productTypes=['wireless'])`

```
{
  "serial": "Q5AC-D43S-XXXX",
  "model": "MS1641",
  "mac": "cc:9c:3e:12:34:56",
  "name": "SB-AP-5F-01"
}
```

`getDeviceClients(serial='Q5AC-D43S-XXXX')`

```
{
  "id": "k74272e",
  "description": "Miles's phone",
  "mac": "22:33:44:55:66:77"
}
```

Expected result of combining



```
{  
  "cc9c3e123456": {  
    "switch_name": "SB-SW-5F-01",  
    "switch_serial": "Q4CD-U6SQ-XXXX",  
    "switch_model": "MS130-24X",  
    "switch_port_id": "11",  
    "power_usage_in_wh": 168.5,  
    "usage_in_kb": { "total": 2, "sent": 1, "recv": 1 },  
    "ap_name": "SB-AP-5F-01",  
    "ap_serial": "Q5AC-D43S-XXXX",  
    "ap_model": "CW9164I",  
    "ap_mac": "cc:9c:3e:12:34:56",  
    "devices": ["k74272e"],  
  }  
}
```

Works great in test environment so run it on production network

Failure :(

CDP information exists on Dashboard but API returns empty

Port 37 | [Return to port list](#)



Status

Connectivity

Usage 7.7 GB (6.4 GB sent, 1.3 GB received)

Traffic 39.5 Kbps (27.2 Kbps sent, 11.3 Kbps received)

CDP/LLDP **Meraki MR55 - SB-AP-5F-01 / 0** (Meraki MR55 Cloud Managed)

PoE usage 10.0 W (2.5 W sent, 7.5 W received)

```
683a1e123456 SB-AP-1F-02 Q2ZD-XXXX-YYYY SB-SW-1F-02 Q2CX-XXXX-YYYY 6
683a1e123456 SB-AP-2F-01 Q2ZD-XXXX-YYYY SB-SW-2F-01 Q2BY-XXXX-YYYY 22
683a1e123456 SB-AP-3F-01 Q2ZD-XXXX-YYYY SB-SW-3F-01 Q2BY-XXXX-YYYY 22
683a1e123456 SB-AP-3F-02 Q2ZD-XXXX-YYYY SB-SW-3F-01 Q2BY-XXXX-YYYY 21
683a1e123456 SB-AP-3F-03 Q2ZD-XXXX-YYYY SB-SW-3F-01 Q2BY-XXXX-YYYY 20
683a1e123456 SB-AP-3F-04 Q2ZD-XXXX-YYYY SB-SW-3F-02 Q2CX-XXXX-YYYY 6
683a1e123456 SB-AP-4F-01 Q2ZD-XXXX-YYYY SB-SW-4F-01 Q2BY-XXXX-YYYY 22
683a1e123456 SB-AP-4F-02 Q2ZD-XXXX-YYYY SB-SW-4F-01 Q2BY-XXXX-YYYY 21
683a1e123456 SB-AP-4F-03 Q2ZD-XXXX-YYYY SB-SW-4F-02 Q2CX-XXXX-YYYY 6
683a1e123456 SB-AP-5F-01 Q2ZD-XXXX-YYYY null None null
683a1e123456 SB-AP-5F-02 Q2ZD-XXXX-YYYY SB-SW-5F-02 Q2BY-XXXX-YYYY 17
683a1e123456 SB-AP-6F-01 Q2ZD-XXXX-YYYY SB-SW-6F-01 Q2BY-XXXX-YYYY 22
683a1e123456 SB-AP-6F-02 Q2ZD-XXXX-YYYY SB-SW-6F-02 Q2CX-XXXX-YYYY 6
683a1e123456 SB-AP-7F-01 Q2ZD-XXXX-YYYY SB-SW-7F-01 Q2BY-XXXX-YYYY 22
683a1e123456 SB-AP-7F-02 Q2ZD-XXXX-YYYY SB-SW-7F-01 Q2BY-XXXX-YYYY 21
683a1e123456 SB-AP-7F-03 Q2ZD-XXXX-YYYY SB-SW-7F-01 Q2BY-XXXX-YYYY 20
683a1e123456 SB-AP-7F-04 Q2ZD-XXXX-YYYY SB-SW-7F-02 Q2CX-XXXX-YYYY 6
683a1e123456 SB-AP-8F-01 Q2ZD-XXXX-YYYY SB-SW-8F-01 Q2BY-XXXX-YYYY 22
```

Workaround

Manually create static switch to access point mapping. Sometimes manual work is faster and required

Schema

```
DATA = {
  "network_1_id": {
    "switch_1_serial": {
      "switch_name": "switch_1_name",
      "ports": {
        "port_num_1": {
          "ap_name": "ap_1_name",
          "serial": "ap_1_serial"
        },
        "port_num_2": {
          "ap_name": "ap_2_name",
          "serial": "ap_2_serial"
        }
      }
    }
  }
}
```

Sample

```
DATA = {
  "L_610800699462125333": {
    "Q4CD-U6SQ-XXXX": {
      "switch_name": "SB-SW-5F-01",
      "ports": {
        7: {
          "ap_name": "SB-AP-5F-01",
          "serial": "Q5AC-D43S-XXXX"
        }
      }
    }
  }
}
```

Manual mapping

```
DATA = {  
  "L_610800699462125333": {  
    "Q4CD-U6SQ-XXXX": {  
      "switch_name": "SB-SW-5F-01",  
      "ports": {  
        7: {  
          "ap_name": "SB-AP-5F-01",  
          "serial": "Q5AC-D43S-XXXX"  
        }  
      }  
    }  
  }  
}
```

getDeviceSwitchPortsStatuses(serial='Q4CD-U6SQ-XXXX')

```
{  
  "portId": "11",  
  "cdp": {  
    "deviceId": "cc9c3e123456",  
    "address": "192.168.10.222"  
  },  
  "powerUsageInWh": 168.5,  
  "usageInKb": { "total": 2, "sent": 1, "recv": 1 }  
}
```

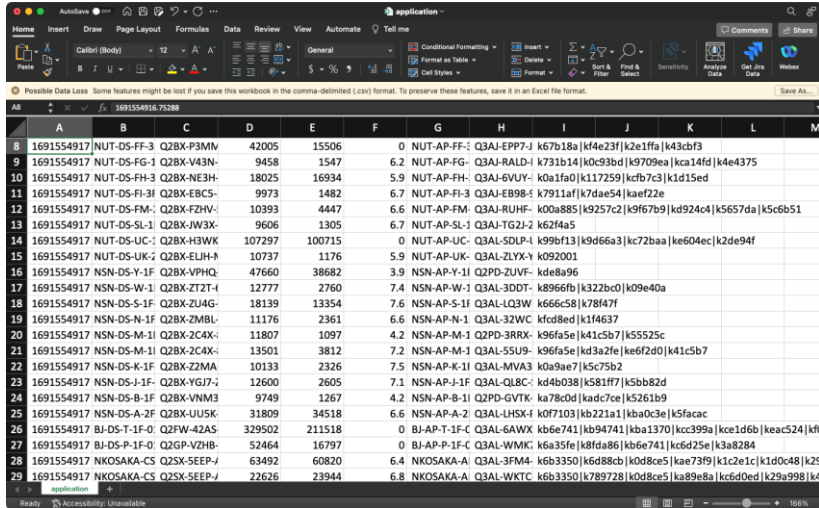
getDeviceClients(serial='Q5AC-D43S-XXXX')

```
{  
  "id": "k74272e",  
  "description": "Miles's phone",  
  "mac": "22:33:44:55:66:77"  
}
```

Run the script with cron

```
ubuntu@ip-192-168-2-109:~$ crontab -l
0 * * * * /usr/bin/python3 /home/ubuntu/poe_monitor.py
```

application.csv



	A	B	C	D	E	F	G	H	I	J	K	L	M
8	1691554917	NUT-DS-FF-3	Q2BX-P3MM	42005	15506	0	NUT-AP-FF-3	Q3AI-EPP7-J	k67b18a k4e23f k2e1fa k43cbf3				
9	1691554917	NUT-DS-FG-3	Q2BX-V43N	9458	1547	6.2	NUT-AP-FG-3	Q3AI-RALD-I	k731b14 k0c93bd k9709ea kca14fd k4e4375				
10	1691554917	NUT-DS-FH-3	Q2BX-NE3H	18025	16934	5.9	NUT-AP-FH-3	Q3AI-6VUY-I	k0a1fa0 k117259 kcfb7c3 k1d15ed				
11	1691554917	NUT-DS-FI-3H	Q2BX-EB5C	9973	1482	6.7	NUT-AP-FI-3	Q3AI-EB98-f	k7911af k7dae54 kaef22e				
12	1691554917	NUT-DS-FM-3	Q2BX-FZHV	10393	4447	6.6	NUT-AP-FM-3	Q3AI-RUHF-f	k00a885 k9257c2 k9f67b9 kd924c4 k5657da k5c6b51				
13	1691554917	NUT-DS-SL-1	Q2BX-JW3X	9606	1305	6.7	NUT-AP-SL-1	Q3AI-TG2J-2	k62fa5				
14	1691554917	NUT-DS-UC-3	Q2BX-H3WK	107297	100715	0	NUT-AP-UC-3	Q3AI-SDLP-I	k99bf13 k9d66a3 kc72baa ke04ec k2de94f				
15	1691554917	NUT-DS-UK-3	Q2BX-EUH-A	10737	1176	5.9	NUT-AP-UK-3	Q3AI-ZLYX-Y	k092001				
16	1691554917	NSN-DS-Y-1F	Q2BX-VPHQ	47660	38682	3.9	NSN-AP-Y-1F	Q2PD-ZUVF-f	kde8a96				
17	1691554917	NSN-DS-W-1	Q2BX-ZT2T-I	12777	2760	7.4	NSN-AP-W-1	Q3AL-3DDT-f	k8966fb k322bc0 k09e40a				
18	1691554917	NSN-DS-S-1F	Q2BX-ZU4G	18139	13354	7.6	NSN-AP-S-1F	Q3AL-LQ3W	k66c58 k78f47				
19	1691554917	NSN-DS-N-1F	Q2BX-ZMBL	11176	2361	6.6	NSN-AP-N-1	Q3AL-32WC	kfc08ed k1f4637				
20	1691554917	NSN-DS-M-1	Q2BX-2C4X	11807	1097	4.2	NSN-AP-M-1	Q2PD-3RRX-f	k96fa5e k41c5b7 k55525c				
21	1691554917	NSN-DS-M-1	Q2BX-2C4X	13501	3812	7.2	NSN-AP-M-1	Q3AL-55U9-f	k96fa5e kd3a2fe ke6f2d0 k41c5b7				
22	1691554917	NSN-DS-K-1F	Q2BX-Z2MA	10133	2326	7.5	NSN-AP-K-1F	Q3AL-MVA3-A	k0a9ae7 k5c75b2				
23	1691554917	NSN-DS-J-1F	Q2BX-YGJ7	12600	2605	7.1	NSN-AP-J-1F	Q3AL-QL8C-f	kd4b038 k581ff7 k5bb82d				
24	1691554917	NSN-DS-B-1F	Q2BX-VNM3	9749	1267	4.2	NSN-AP-B-1F	Q2PD-GVTK-f	ka78cd0 kadc7ce k5261b9				
25	1691554917	NSN-DS-A-2F	Q2BX-UU5K	31809	34518	6.6	NSN-AP-A-2	Q3AL-LHSX-f	k07f103 kb221a1 kba0c3e k5facac				
26	1691554917	BJ-DS-T-1F-0	Q2FW-42AS	329502	211518	0	BJ-AP-T-1F-0	Q3AL-6AWX	kb6e741 kb94741 kba1370 kcc399a kce1d6b keac524 kfo				
27	1691554917	BJ-DS-P-1F-0	Q2GP-VZHB	52464	16797	0	BJ-AP-P-1F-0	Q3AL-WMMK	k6a35fe k8fda86 kb6e741 kcf6d25e k3a8284				
28	1691554917	NKOSAKA-CS	Q2SX-SEEP-J	63492	60820	6.4	NKOSAKA-A	Q3AL-3FM4	k6b3350 k6d88cb k0d8ce5 kae73f9 k1c2e1c k1d0c48 k29				
29	1691554917	NKOSAKA-CS	Q2SX-SEEP-J	22626	23944	6.8	NKOSAKA-A	Q3AL-WKTC	k6b3350 k789728 k0d8ce5 ka89e8a kcf6d0ed k29a998 k4				

client_data.txt

```
{
  "id": "k0cfa39",
  "mac": "3e:e4:93:12:34:56",
  "description": "Yujis-iPhone",
  "mdnsName": "Yujis-iPhone.local",
  "dhcpHostname": null,
  "user": null,
  "ip": "10.12.250.57",
  "vlan": 0,
  "usage": {
    "sent": 30195.0,
    "recv": 2998.99999999999995
  }
}
```

Apply port schedule

Create port schedule

Port schedules

Local time zone: Australia - Sydney (You can set this on [General](#))

MR used by 0 ports

Templates: 8 to 5 daily 8 to 5 on weekdays only weekdays only always on always off

Time display: 24 Hour AM/PM

Day	Status	During	0:00	4:00	8:00	12:00	16:00	20:00
Monday	enabled	6:00 21:00						
Tuesday	enabled	6:00 21:00						
Wednesday	enabled	6:00 21:00						
Thursday	enabled	6:00 21:00						
Friday	enabled	6:00 21:00						
Saturday	disabled	0:00 24:00						
Sunday	disabled	0:00 24:00						

Apply port schedule

Update 1 port

Switch / Port: b8:ab:61:12:34:56 / 11

Name:

Port status: ☒ Enabled ☐ Disabled

Link negotiation: Auto negotiate

Port schedule: MR

Tags: Unscheduled

Port profile: MR

Type: ☒ Trunk ☐ Access

Access policy @: Open

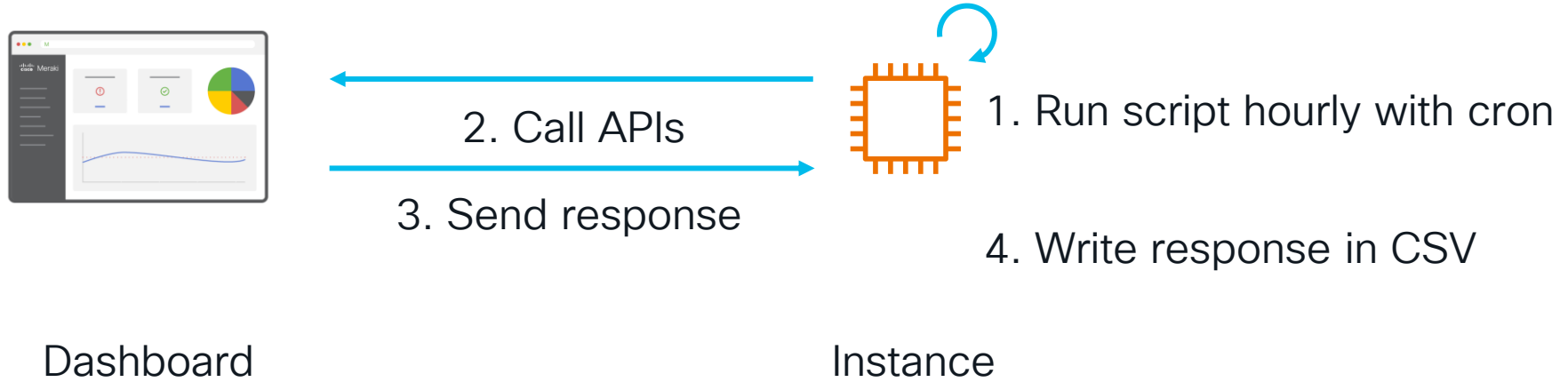
Native VLAN: 1

Allowed VLANs: all

RSTP: ☒ Enabled ☐ Disabled

Cancel Update

MT40 flow



MT30 flow

```
{
  ...,
  "sharedSecret": "Meraki1234",
  "organizationId": "734417",
  "networkId": "L_726205439913496800",
  "deviceSerial": "Q3CH-MT30-XXXX",
  "alertData": {
    ...,
    "message": "Pressed MT30 in room"
  }
}
```



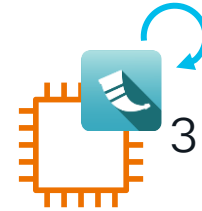
1. Triggers alert



Dashboard

2. Sends webhook

4. Return 200 OK



3. Run script

MT30

Instance

code: <https://gist.github.com/yujiterada/4bd7332de926525ee388ba2c1219779c>

Send response before processing

Filter by

All Requests

Clear Requests

5 minutes ago Duration 0ms IP 209.206.XX.XX

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

POST /receive_json

Summary

Headers

Raw

Binary

Replay

1107 bytes application/json

```
{
  "version": "0.1",
  "hostname": "Meraki1234",
  "sentAt": "2023-10-23T08:48:25.172395Z",
  "organizationId": "123456",
  "organizationName": "TEST ORGANIZATION",
  "organizationUrl": "https://dashboard.meraki.com",
  "networkId": "123456789",
  "networkName": "TEST NETWORK",
  "networkUrl": "https://dashboard.meraki.com",
  "networkTags": [],
  "deviceSerial": "ABCD-EFGH-IJKL",
  "deviceMac": "12:34:56:AB:CD:EF",
  "deviceName": "Button",
  "deviceUrl": "https://n290.meraki.com/Home-Sydney-entv
i/n/Udb55bte/manage/nodes/new_list/216184030720320",
  "deviceTags": [],
  "deviceModel": "MT30",
  "alertId": "726205439932911081",
  "alertType": "Sensor automation action",
  "alertTypeId": "sensor_automation",
  "alertLevel": "informational",
  "occurredAt": "2023-10-23T08:48:20.000000Z",
  "alertData": {
    "automationId": 726205439913492600,
    "conditionId": 726205439913492600,
    "action": {
      "id": 726205439913492600,
      "type": null
    },
    "trigger": {
```

Dashboard sends retries if there's no response. This could trigger the script to run multiple times.

mt30_receiver_2.py

Users > yujterad > Desktop > mt30_receiver_2.py

```
1 from flask import Flask, request, jsonify
2 import meraki
3
4 import json
5 import os
6 import threading
7 import time
8
9 app = Flask(__name__)
10
11 SHARED_SECRET = "Meraki1234"
12 API_KEY = os.environ.get('MERAKI_API_KEY')
13 > DATA = {
14     }
15
16 def background_task(data):
17     # This code will be executed in the background after sending the response
18     print("Background task is running")
19
20 @app.route('/receive_json', methods=['POST'])
21 def receive_json():
22     try:
23         # Get JSON data from the request
24         data = request.get_json()
25
26         if data.get('sharedSecret') != SHARED_SECRET:
27             raise ValueError("Invalid sharedSecret provided.")
28
29         # Start the background task
30         threading.Thread(target=background_task(data)).start()
31
32         # Create response to Dashboard
33         response = jsonify(success=True), 200
34
35         # Respond to Dashboard
36         return response
37
38 except Exception as e:
39     print("An error occurred:", str(e))
40     return jsonify(success=False, error=str(e)), 400
41
42 if __name__ == '__main__':
43     app.run(host='0.0.0.0', port=8080)
```

Python - Desktop

yujterad@YUJTERAD-M-HCSK Desktop % python3 mt30_receiver_2.py

```
* Serving Flask app 'mt30_receiver_2'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://192.168.10.232:8080
Press CTRL+C to quit
```

CISCO *Live!*

#CiscoLiveAPJC

DEVNET-1681

© 2023 Cisco and/or its affiliates. All rights reserved. Cisco Public

29

Background task detail

Change port schedule to unscheduled

Update 1 port

Switch / Port: b8:ab:61:12:34:56 / 11

Name: [text field]

Port status: **Enabled** Disabled

Link negotiation: Auto negotiate

Port schedule: MR

Tags: Unscheduled

Port profile: MR

Enabled Disabled

Type: **Trunk** Access

Access policy: Open

Native VLAN: 1

Allowed VLANs: all

RSTP: **Enabled** Disabled

Cancel Update

Port schedule does not disable port status. Disables port in the background.

Change port schedule to unscheduled to enable port.

test.py

mt30_receiver.py

mt30_sender.py

poe_monitor.py

...

Users > yujterad > Desktop > mt30_receiver.py

```
1 from flask import Flask, request, jsonify
2 import meraki
3
4 import json
5 import os
6 import threading
7 import time
8
9 app = Flask(__name__)
10
11 SHARED_SECRET = "Meraki1234"
12 API_KEY = os.environ.get('MERAKI_API_KEY')
13 DATA = {
14     "Q3CH-MT30-XXXX": {
15         "Q4CD-MS13-YYYY": [11]
16     }
17 }
18
19 def background_task(data):
20     # This code will be executed in the background after sending the response
21     m_client = meraki.DashboardAPI(API_KEY, output_log=False, suppress_logging=True)
22     for serial, ports in DATA[data['deviceSerial']].items():
23         for port in ports:
24             resp = m_client.switch.updateDeviceSwitchPort(
25                 serial=serial, portId=port, portScheduleId=None)
26             print(json.dumps(resp, indent=4))
27
28 @app.route('/receive_json', methods=['POST'])
29 def receive_json():
30
31     if __name__ == '__main__':
32         app.run(host='0.0.0.0', port=8080)
```

zsh - Desktop

yujterad@YUJTERAD-M-HC5K Desktop % python3 mt30_receiver.py

```
* Serving Flask app 'mt30_receiver'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://192.168.10.232:8080
Press CTRL+C to quit
{
  "portId": "11",
  "name": null,
  "tags": [],
  "enabled": true,
  "poeEnabled": true,
  "type": "trunk",
  "vlan": 1,
  "voiceVlan": null,
  "allowedVlans": "all",
  "isolationEnabled": false,
  "rstpEnabled": true,
  "stpGuard": "disabled",
  "linkNegotiation": "Auto negotiate",
  "portScheduleId": null,
  "udld": "Alert only",
  "linkNegotiationCapabilities": [
    "Auto negotiate",
    "2.5 Gigabit full duplex (forced)",
    "1 Gigabit full duplex (forced)",
    "100 Megabit (auto)"
  ],
  "accessPolicyType": "Open",
  "daiTrusted": false,
  "profile": {
    "enabled": false,
    "id": "",
    "iname": null
  }
}
```

127.0.0.1 -- [23/Oct/2023 21:00:49] "POST /receive_json HTTP/1.1" 200 -

Ln 15, Col 24

Spaces: 4

UTF-8

LF

Python

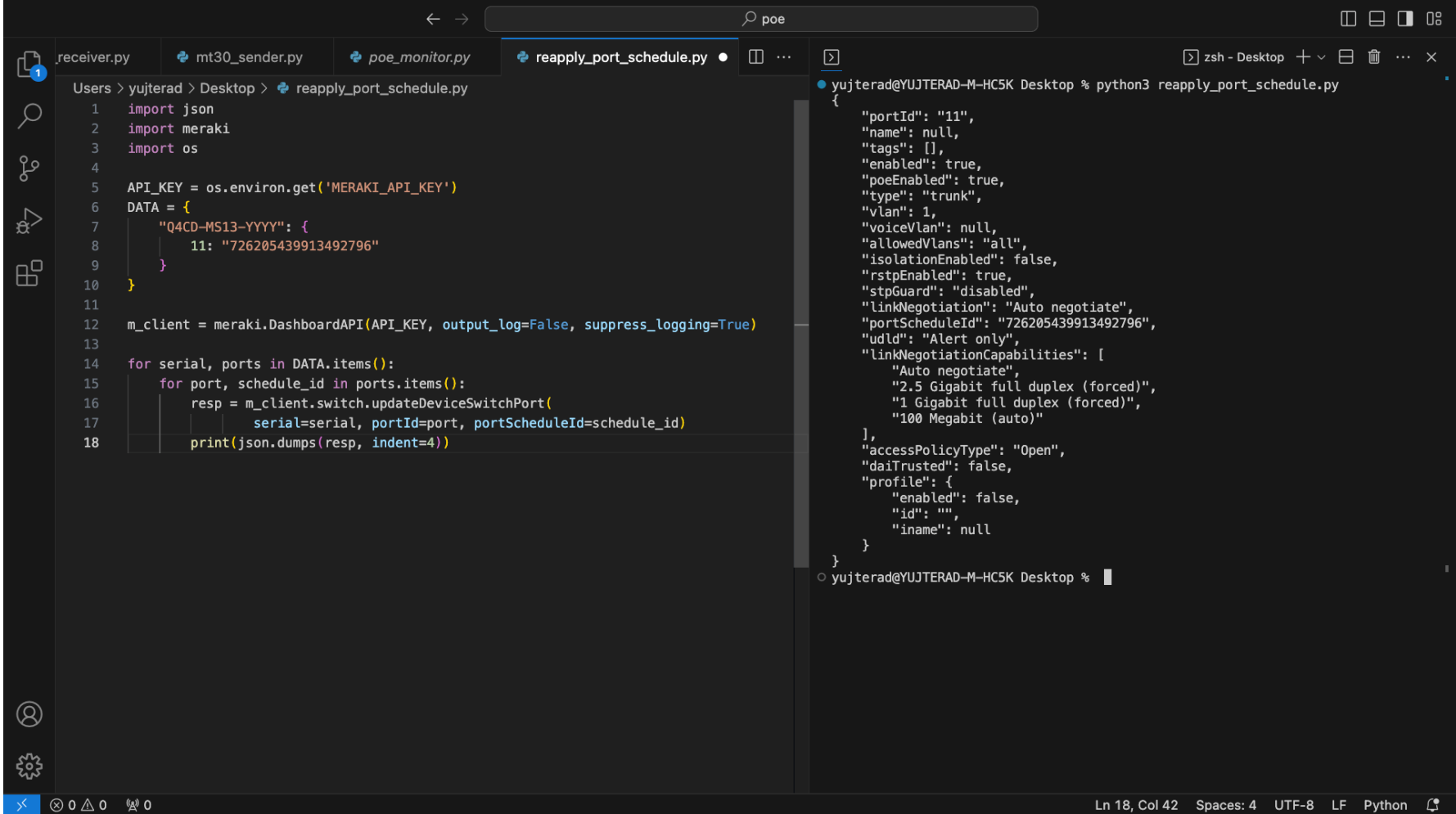
Reapply port schedule

Once MT30 button is pressed, port schedule changes

The screenshot shows the 'Update 1 port' configuration window. The 'Port schedule' dropdown is set to 'Unscheduled'. Other visible settings include: Switch / Port (b8:ab:61:12:34:56 / 11), Name (empty), Port status (Enabled), Link negotiation (Auto negotiate), Tags (+), Port profile (Enabled), Type (Trunk), Access policy (Open), Native VLAN (1), Allowed VLANs (all), and RSTP (Enabled). The 'Update' button is at the bottom right.

Need to reapply original port schedule so access point will power off the next day

code: <https://gist.github.com/yujiterada/5c198d07092bf359db785290af569fc5>



Run the script with cron

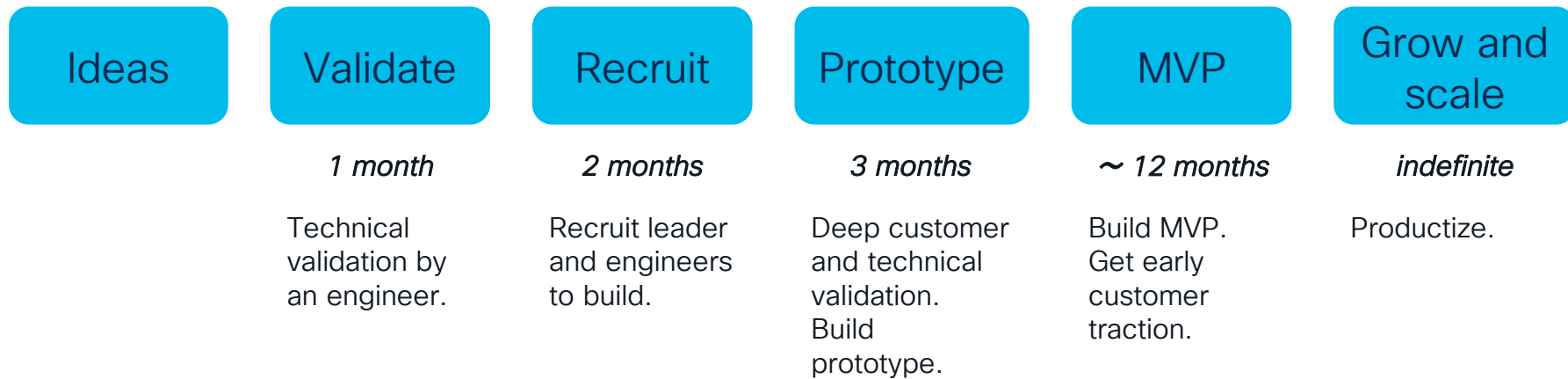
```
ubuntu@ip-192-168-2-109:~$ timedatectl
Local time: Tue 2023-10-24 03:18:52 UTC
Universal time: Tue 2023-10-24 03:18:52 UTC
RTC time: Tue 2023-10-24 03:18:52
Time zone: Etc/UTC (UTC, +0000)
System clock synchronized: yes
NTP service: active
RTC in local TZ: no

ubuntu@ip-192-168-2-109:~$ crontab -l
0 * * * * /usr/bin/python3 /home/ubuntu/poe_monitor.py
0 3 * * * /usr/bin/python3 /home/ubuntu/reapply_port_schedule.py

ubuntu@ip-192-168-2-109:~$ cat /var/log/syslog grep CRON
Oct 24 03:00:01 ip-192-168-2-109 CRON[297410]: (ubuntu) CMD (/usr/bin/python3
/home/ubuntu/poe_monitor.py)
Oct 24 03:00:01 ip-192-168-2-109 CRON[297411]: (ubuntu) CMD (/usr/bin/python3
/home/ubuntu/reapply_port_schedule.py)
```

Tips & tricks

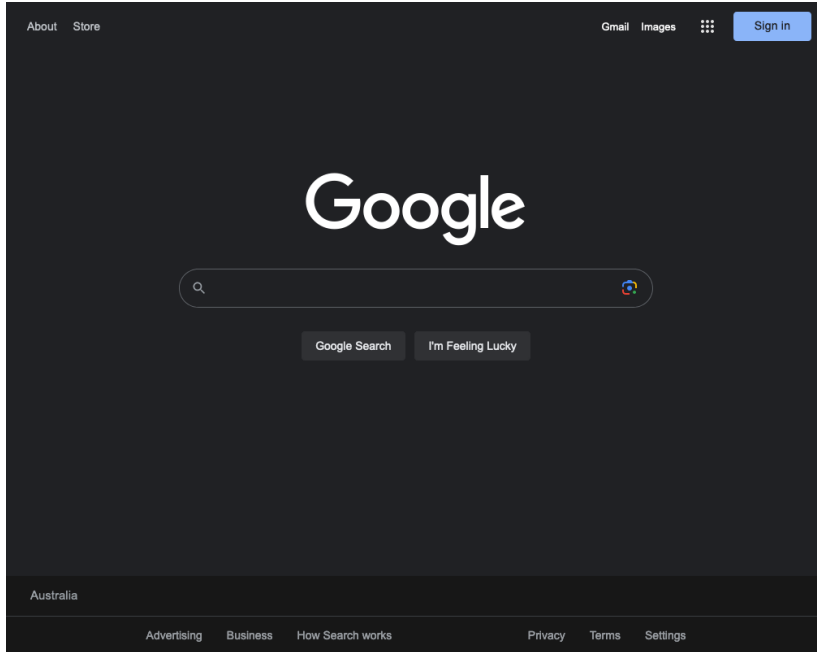
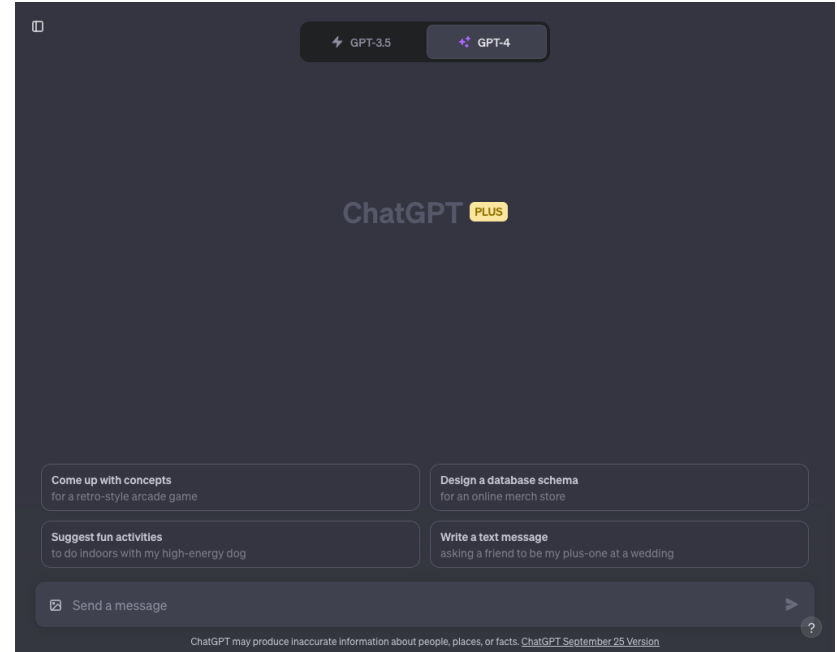
Create prototype



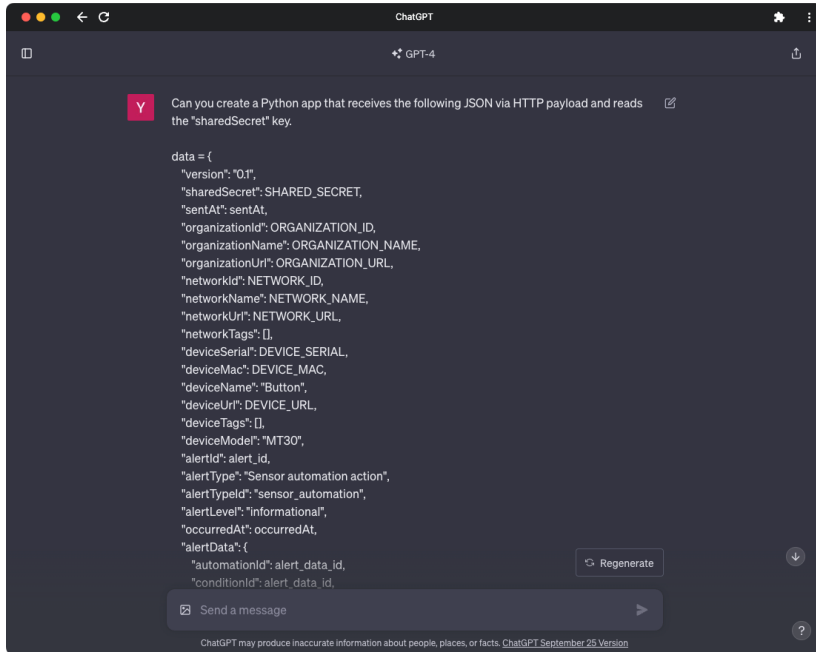
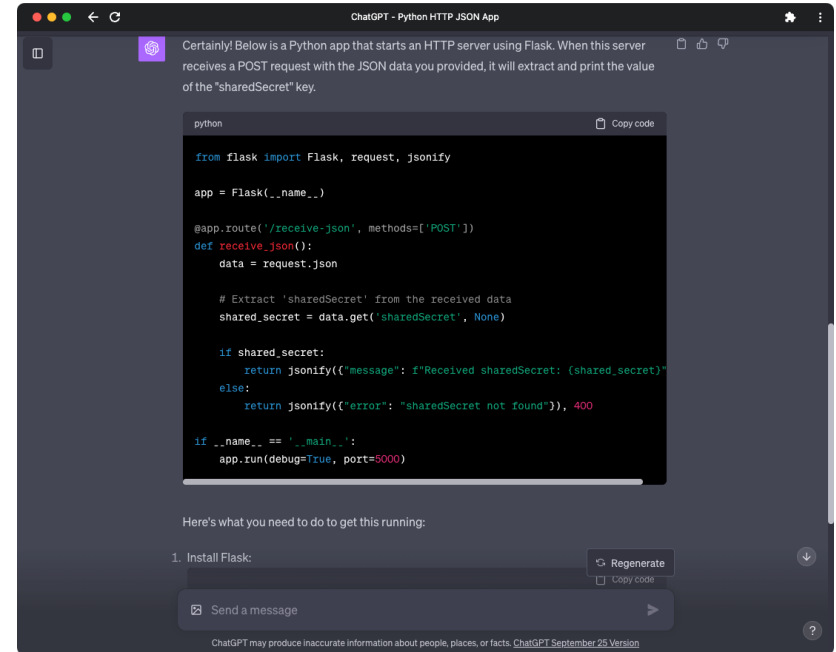
source: [Andrew Ng: Opportunities in AI - 2023](#)

Changing habits

Google Search vs ChatGPT

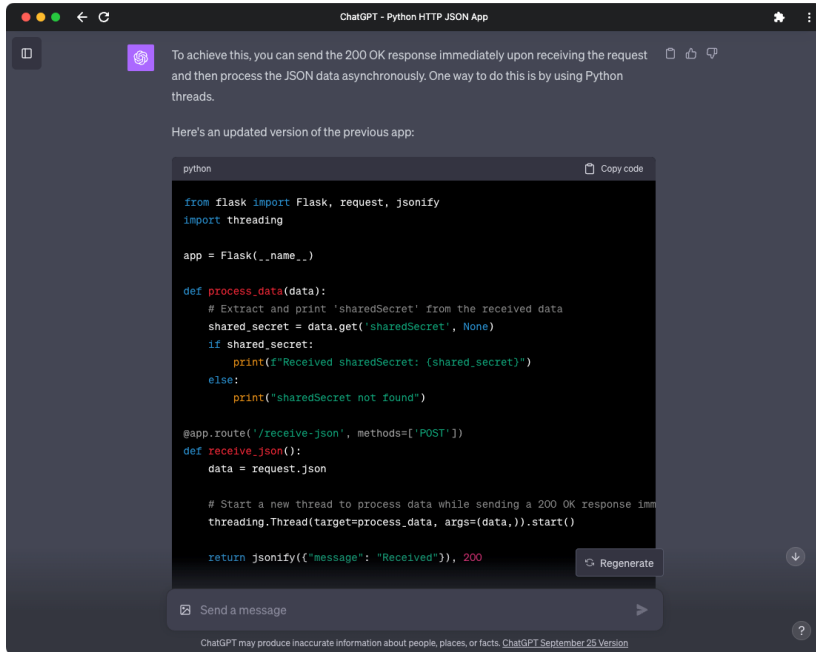


Create a Python app that receives MT30 payload



<https://chat.openai.com/share/e64959d0-4950-4481-ae35-4414be02dce0>

The app should respond 200 OK first



ChatGPT - Python HTTP JSON App

To achieve this, you can send the 200 OK response immediately upon receiving the request and then process the JSON data asynchronously. One way to do this is by using Python threads.

Here's an updated version of the previous app:

```
python
from flask import Flask, request, jsonify
import threading

app = Flask(__name__)

def process_data(data):
    # Extract and print 'sharedSecret' from the received data
    shared_secret = data.get('sharedSecret', None)
    if shared_secret:
        print(f"Received sharedSecret: {shared_secret}")
    else:
        print("sharedSecret not found")

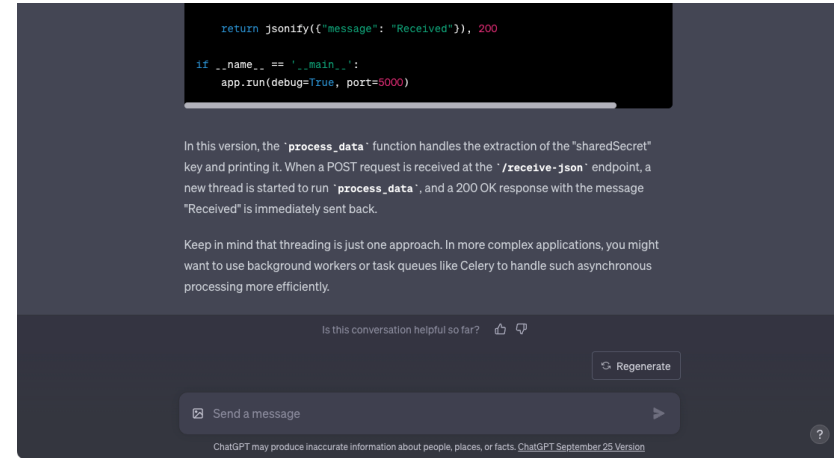
@app.route('/receive-json', methods=['POST'])
def receive_json():
    data = request.json

    # Start a new thread to process data while sending a 200 OK response immediately
    threading.Thread(target=process_data, args=(data,)).start()

    return jsonify({"message": "Received"}), 200
```

Send a message

ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT September 25 Version



```
return jsonify({"message": "Received"}), 200

if __name__ == '__main__':
    app.run(debug=True, port=5000)
```

In this version, the `process_data` function handles the extraction of the `sharedSecret` key and printing it. When a POST request is received at the `/receive-json` endpoint, a new thread is started to run `process_data`, and a 200 OK response with the message "Received" is immediately sent back.

Keep in mind that threading is just one approach. In more complex applications, you might want to use background workers or task queues like Celery to handle such asynchronous processing more efficiently.

Is this conversation helpful so far?

Regenerate

Send a message

ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT September 25 Version

<https://chat.openai.com/share/e64959d0-4950-4481-ae35-4414be02dce0>

Call to action

Get your hands dirty

Webex space



Meraki DevNet



Meraki community



Continue your education



- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Expert meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand



Did you know?

You can have a
one-on-one session with
a technical expert!

Visit Meet the Expert in The HUB
to meet, greet, whiteboard & gain
insights about your unique questions
with the best of the best.



Meet the Expert Opening Hours:

Tuesday	3:00pm – 7:00pm
Wednesday	11:15am – 7:00pm
Thursday	9:30am – 4:00pm
Friday	10:30am – 1:30pm

Engage with us: Become a Cisco Meraki Insider



Join
the community



Complete
challenges



Receive
rewards

Session surveys

We would love to know your feedback on this session!

- Complete a minimum of four session surveys and the overall event surveys to claim a Cisco Live T-Shirt





The bridge to possible

Thank you

CISCO *Live!*

#CiscoLiveAPJC

The Cisco Live! logo, featuring the word "CISCO" in a dark blue, sans-serif font, followed by "Live!" in a dark blue, cursive script font.

CISCO *Live!*

The text "Let's go" in a dark blue, sans-serif font, positioned to the left of a bright, multi-colored sunburst graphic that radiates from the right side of the image.

Let's go

#CiscoLiveAPJC

Appendix

Traffic and PoE usage

View historical data and calculate how much power can be saved

Dashboard

Port 11 [Return to port list](#)

[Learn more](#)



Historical data for the last day ▾

Port traffic



Status

Connectivity 

Usage 8.0 MB (5.8 MB sent, 2.2 MB received)

Traffic 42.6 Kbps (24.4 Kbps sent, 18.2 Kbps received)

CDP/LLDP [Office - Sydney - wireless / 0](#) (Meraki CW9164I Cloud Managed AP) [raw](#)

PoE usage 12.172 W AT (Advertised 30 W AT)

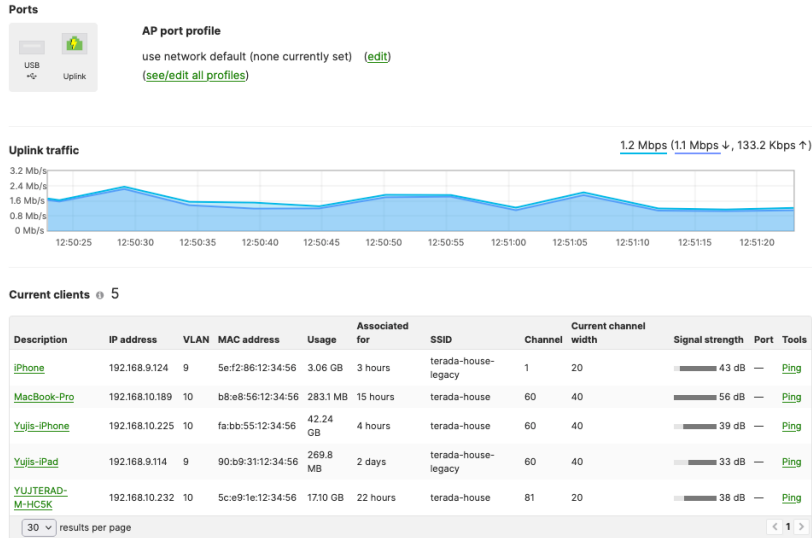
API *getDeviceSwitchPortsStatuses*

```
[
  ...,
  {
    "portId": "11",
    "enabled": true,
    "status": "Connected",
    "errors": [],
    "warnings": [],
    "powerUsageInWh": 2.5,
    "trafficInKbps": {
      "total": 14.6,
      "sent": 12.6,
      "recv": 2.0
    },
    ...
  },
  ...
]
```

Client details of a device

Find out office hour trends and if there's any clients connected 24/7

Dashboard



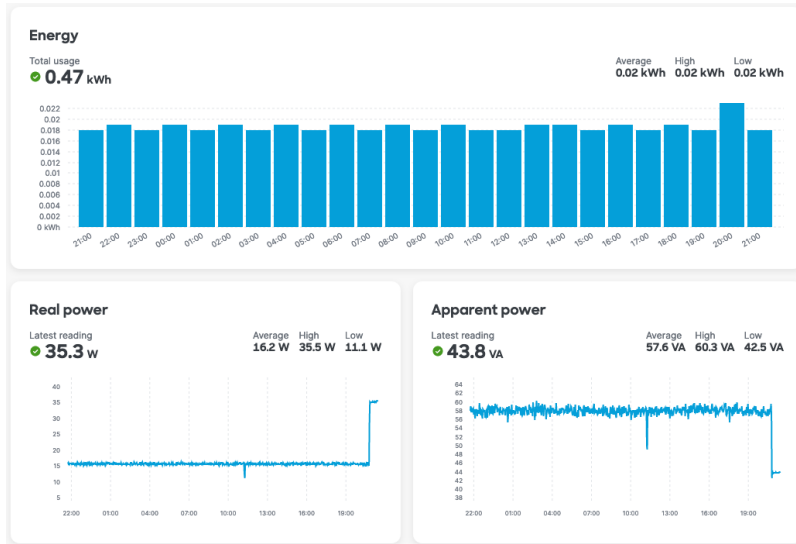
API `getDeviceClients`

```
[
  {
    "id": "kac63bb",
    "mac": "fa:bb:55:ab:cd:ef",
    "description": "Yujis-iPhone",
    "mdnsName": "Yujis-iPhone.local",
    "user": null,
    "ip": "192.168.10.225",
    "vlan": 10,
    "usage": {
      "sent": 3039116.99999999995,
      "recv": 160607.99999999994
    }
  }, ...
]
```

Real power usage

Monitor power consumption of AC powered device via MT40

Dashboard



API `getOrganizationSensorReadingsLatest`

```
[
  ...,
  {
    "serial": "Q3CJ-ABCD-EFGH",
    "readings": [
      ...,
      {
        "ts": "2023-10-
18T04:26:09Z",
        "metric": "realPower",
        "realPower": {
          "draw": 14.8
        }
      }, ...
    ]
  }, ...
]
```

Port schedule

Dashboard

Port schedules

Local time zone: Australia - Sydney (You can set this on [General](#))

MR used by 0 ports

Templates: 8 to 5 daily 8 to 5 on weekdays only weekdays only always on always off

Time display: 24 Hour AM/PM

Day	Status	During	
Monday	enabled	6:00 21:00	
Tuesday	enabled	6:00 21:00	
Wednesday	enabled	6:00 21:00	
Thursday	enabled	6:00 21:00	
Friday	enabled	6:00 21:00	
Saturday	disabled	0:00 24:00	
Sunday	disabled	0:00 24:00	

API *createNetworkSwitchPortSchedule*

```
{
  "portSchedule": {
    ...,
    "friday": {
      "active": "true",
      "from": "6:00",
      "to": "21:00"
    },
    "saturday": {
      "active": "false",
      "from": "0:00",
      "to": "24:00"
    }, ...
  }
}
```

MT30 send webhook to trigger code

Change port schedule by updating port config

Dashboard

- ☐ **Send a Notification** Stay informed with a custom notification over SMS or e-mail
- ☐ **Camera snapshot** Take and send a camera snapshot from your MV cameras and never miss a moment
- ☐ **Toggle SSID** Control internet access by turning a Meraki network SSID on or off
- ☐ **Toggle Switchport** Toggle any Meraki switchport and adjust your network to real-time feedback

☒ **Send a Webhook command** Tap into thousands of smart automations by setting up a custom Webhook integration

Message

Pressed MT30 in room

Webhook Recipients

Ngrok

Configure HTTP servers on [network-wide alerts](#)

API *updateDeviceSwitchPort*

Apply port schedule

```
{  
  "portScheduleId": "726205439913492796"  
}
```

Remove port schedule

```
{  
  "portScheduleId": None  
}
```

Alerts

Suppress alerts when access points goes down due to port schedule

 Global Overview

 Organization
Yuji Terada Personal ▾

 Network
Home - Sydney ▾

 Network-wide

 Security & SD-WAN

 Switching

 Wireless

 Cameras

Alerts

Configure organization-wide alerts at [Organization > Settings](#)

Alerts Settings

Default recipients

Network-wide

- ☐ Configuration settings are changed
- ☐ A VPN connection comes up or goes down
- ☐ A rogue access point is detected
- ☐ Network usage exceeds
- ☒ Mute wireless alerts based on switch port schedules ⓘ

Wireless

- ☐ A gateway goes offline for minutes

If toggled on, then wireless unreachable alerts will be muted when caused by a port schedule.
[Learn more](#)

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

poe_monitor.py

1 import json

2 import os

3 import sys

4 import re

5 import logging

6 import time

7

8 import meraki

9

10 > DATA = {-

11 }

12

13 # Get the API key and organization ID from the environment variables

14 API_KEY = os.environ.get('MERAKI_API_KEY')

15

16 if not API_KEY:

17 | sys.exit()

18

19 # Configure logging

20 logging.basicConfig(filename='application.csv', format='%(message)s', level=logging.INFO)

21 logger = logging.getLogger()

22

23 # Load client data

24 client_data = {}

25 if os.path.exists('client_data.txt'): # check if the file exists

26 | with open('client_data.txt', 'r') as infile:

27 | | client_data = json.load(infile)

28

29 # Obtain current time

30 timestamp = time.time()

31

32 # Initialize the Dashboard API with your API key

33 m_client = meraki.DashboardAPI(API_KEY, output_log=False, suppress_logging=True)

34

35 for network_id, network_DATA in DATA.items():

36

37 | for switch_serial, switch_DATA in network_DATA.items():

38

39 | | switch_name = switch_DATA['switch_name']

40 | | port_statuses = m_client.switch.getDeviceSwitchPortsStatuses(

41 | | | serial=switch_serial, timespan=3600)

42 |

43 | | for port_status in port_statuses:

44

45

46

47

48

49

50

51

52

53

poe_monitor.py

754

755 | ap_DATA = switch_DATA['ports'].get(int(port_status['portId']))

756 | if not ap_DATA:

757 | | continue

758 | sent_in_kb = port_status.get('usageInKb', {}).get('sent', 0)

759 | recv_in_kb = port_status.get('usageInKb', {}).get('recv', 0)

760 | power_usage_in_wh = port_status.get('powerUsageInWh', 0)

761 | ap_name = ap_DATA['ap_name']

762 | ap_serial = ap_DATA['serial']

763

764 | device_clients = m_client.devices.getDeviceClients(serial=ap_serial, timespan=3600)

765 | for client in device_clients:

766 | | if not client_data.get(client['id']):

767 | | | client_data[client['id']] = client

768 | client_ids = '|'.join([client['id'] for client in device_clients])

769

770 | logger.info('{}', {}, {}, {}, {}, {}, {}, {}, {}, {}).format(

771 | | timestamp, switch_name, switch_serial, sent_in_kb, recv_in_kb,

772 | | power_usage_in_wh, ap_name, ap_serial, client_ids)

773

774 # Write client details

775 with open('client_data.txt', 'w') as outfile:

776 | json.dump(client_data, outfile)

Ln 749, Col 50 Spaces: 4 UTF-8 LF Python