

CISCO *Live!*

GO BEYOND

#CiscoLiveAPJC



Cisco APIs for a Sustainable Future

Adrian ILIESIU, Senior Developer Advocate
@aidevnet
DEVNET-1022



#CiscoLiveAPJC

Cisco Webex App

Questions?

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

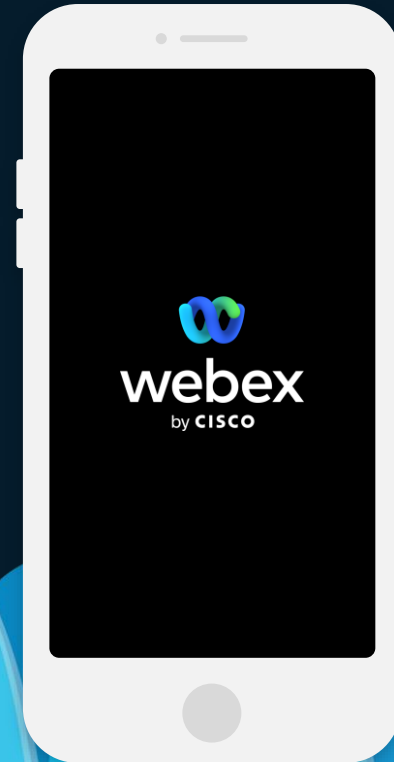
How

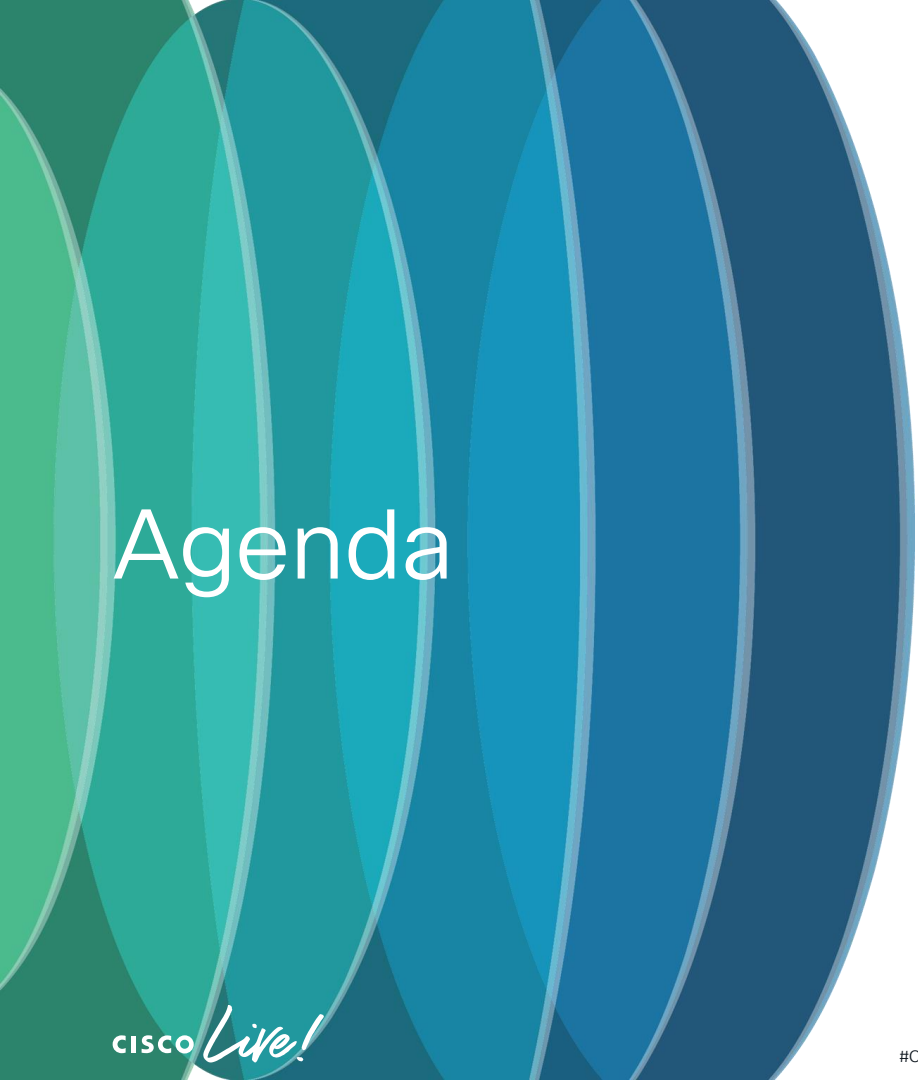
- 1 Find this session in the Cisco Live Mobile App
- 2 Click “Join the Discussion”
- 3 Install the Webex App or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until November 15, 2024.



<https://ciscolive.ciscoevents.com/ciscolivebot/#DEVNET-1022>





Agenda

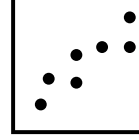
- Sustainability at Cisco
- Catalyst Center APIs for sustainability
- Meraki APIs for sustainability
- Cisco IOS XE sustainability

Sustainability at Cisco

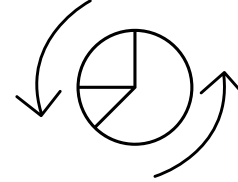
Where to start?



Sustainable Goals



Metrics



Data



<https://www.unicef.org.au/un-sustainable-development-goals>

Goals

Cisco's Goal

Net zero GHG emissions across our value chain by 2040

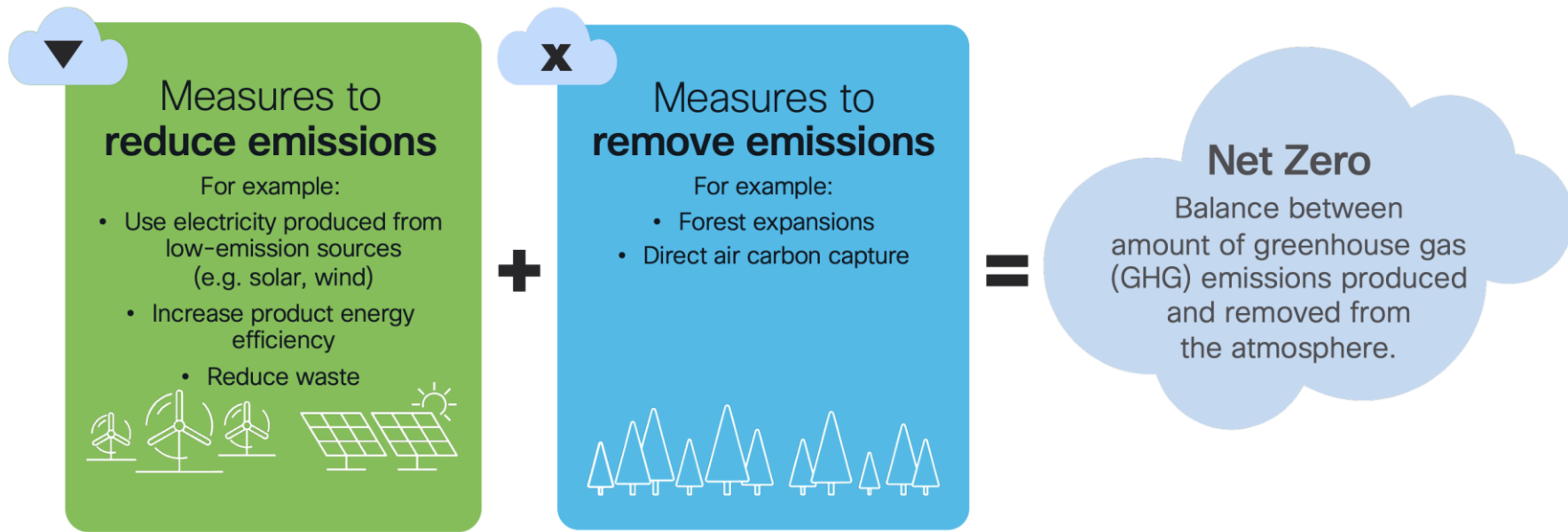
Our 2040 net-zero goals and near-term targets are approved by the Science Based Targets initiative (SBTi).



DRIVING AMBITIOUS CORPORATE CLIMATE ACTION



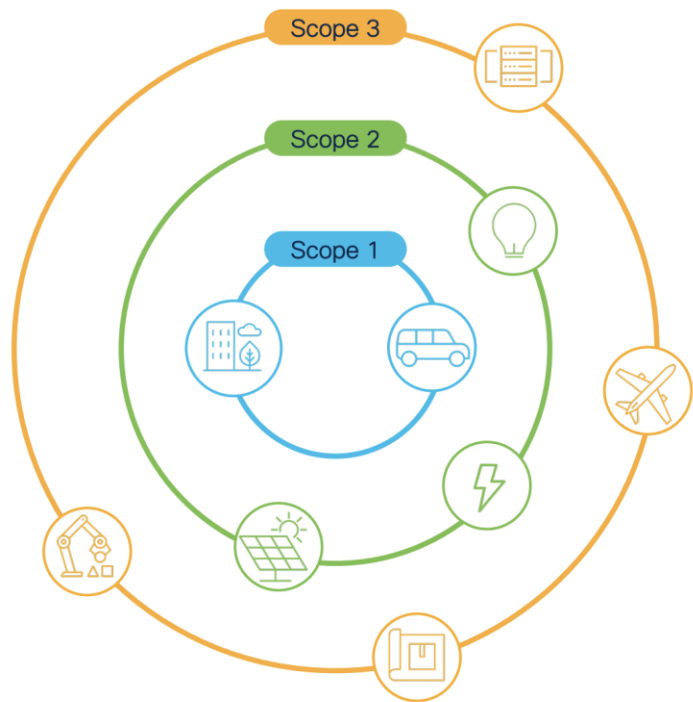
What is NetZero?



Cisco's SBTi Commitment

Cisco's 2040 Net Zero target is approved by the Science Based Targets Initiative (SBTi). This means we have pledged to reduce our emissions by at least 90% (compared to our base year of 2019). We will invest in verified carbon removal offsets to compensate for any residual emissions.

Sustainability Metrics



Greenhouse gas (GHG) emissions of organizations are classified in 3 scopes:

Scope 1

Direct emissions from operations owned by the company

Examples: Heating fuel, company vehicles

Scope 2

Indirect emissions from the generation of purchased electricity, steam, heating and cooling

Examples: Electricity used in company buildings

Scope 3

All other indirect emissions that occur in a company's value chain

Examples: Employee business travel and commuting, supplier emissions, use of sold products, transportation and distribution

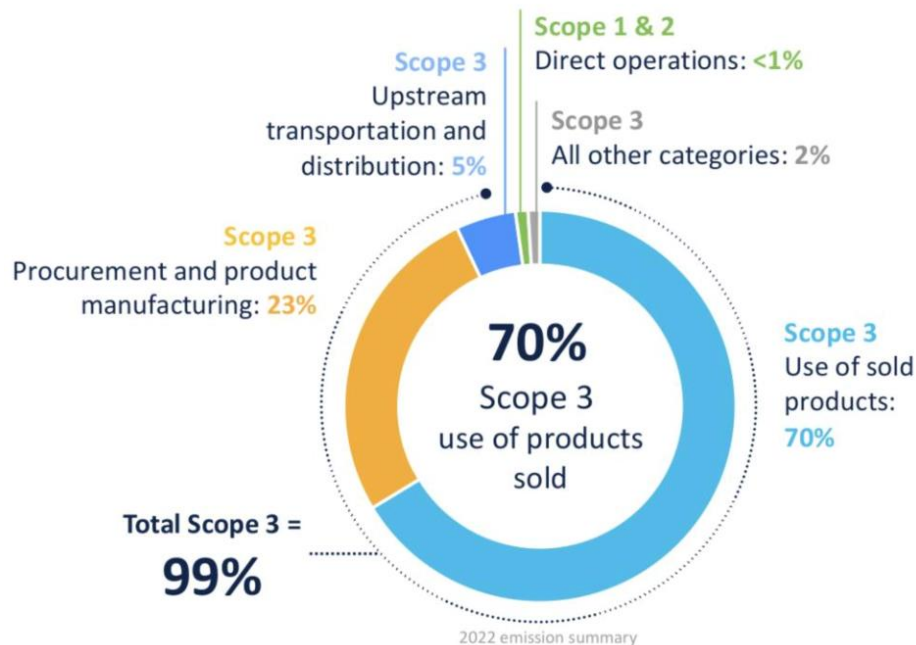
Goals

Cisco's Goal Net zero GHG emissions across our value chain by 2040

Our 2040 net-zero goals and near-term targets are approved by the Science Based Targets initiative (SBTi).

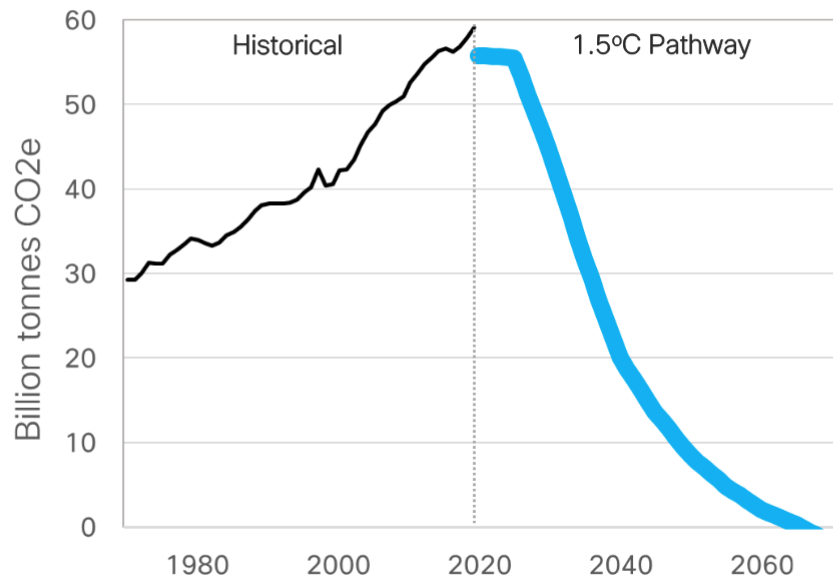


Metrics



What about the data?

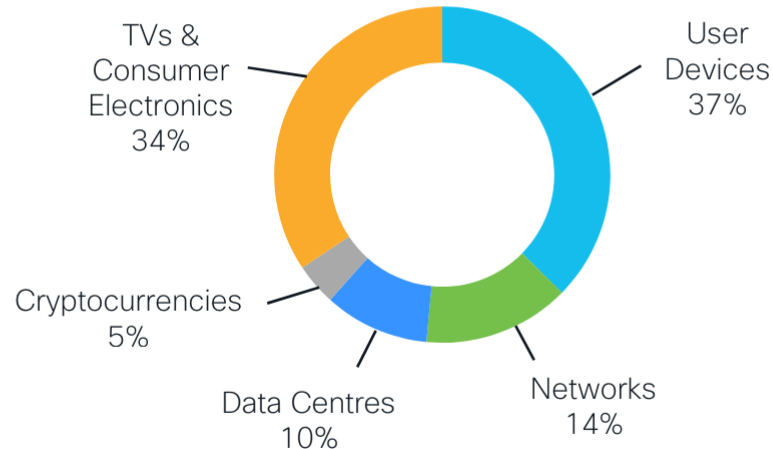
Total GHG emissions ~60 Billion tonnes CO₂e



Based on data from: EDGARv7.0, Crippa et al (2021, 2022), IIASA (AR6 Scenario database)

Digital GHG emissions

~1.4 Billion tonnes CO₂e (2-4% Total)



Digital = ICT + Entertainment & Media

Source: Freitag (2021), Malmodin (2018), Andrae (2019)

What is the role of the network and IT?

“Sustainability in IT”



- Renewable Energy
- Greener Data Centre
- Cloud Migration
- Energy Optimization
- Increased Asset Utilization
- Equipment Modernization
- Extended Life
- Remanufactured Equipment
- Supplier Management

2-4%



15%



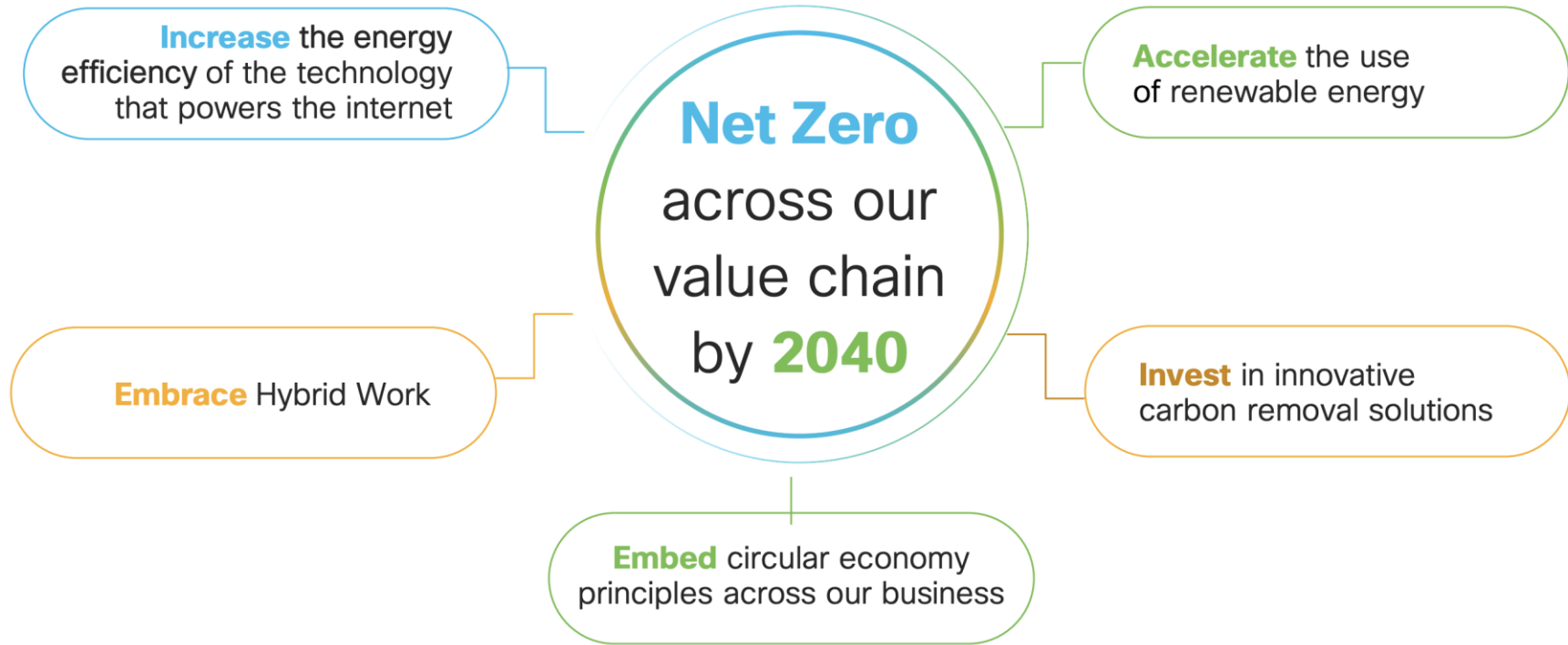
“Sustainability by IT”



- Connected Transport
- E-Work / Hybrid Work
- E-Health
- E-Learning
- E-Commerce
- Smart Buildings
- Smart Grid
- Smart Agriculture
- Smart Manufacturing

*Based on Exponential Roadmap 1.5.1 (2020),
Smarter 2030 (GeSI 2015), Malmodyn (2015)*

Cisco's strategies to get to NetZero

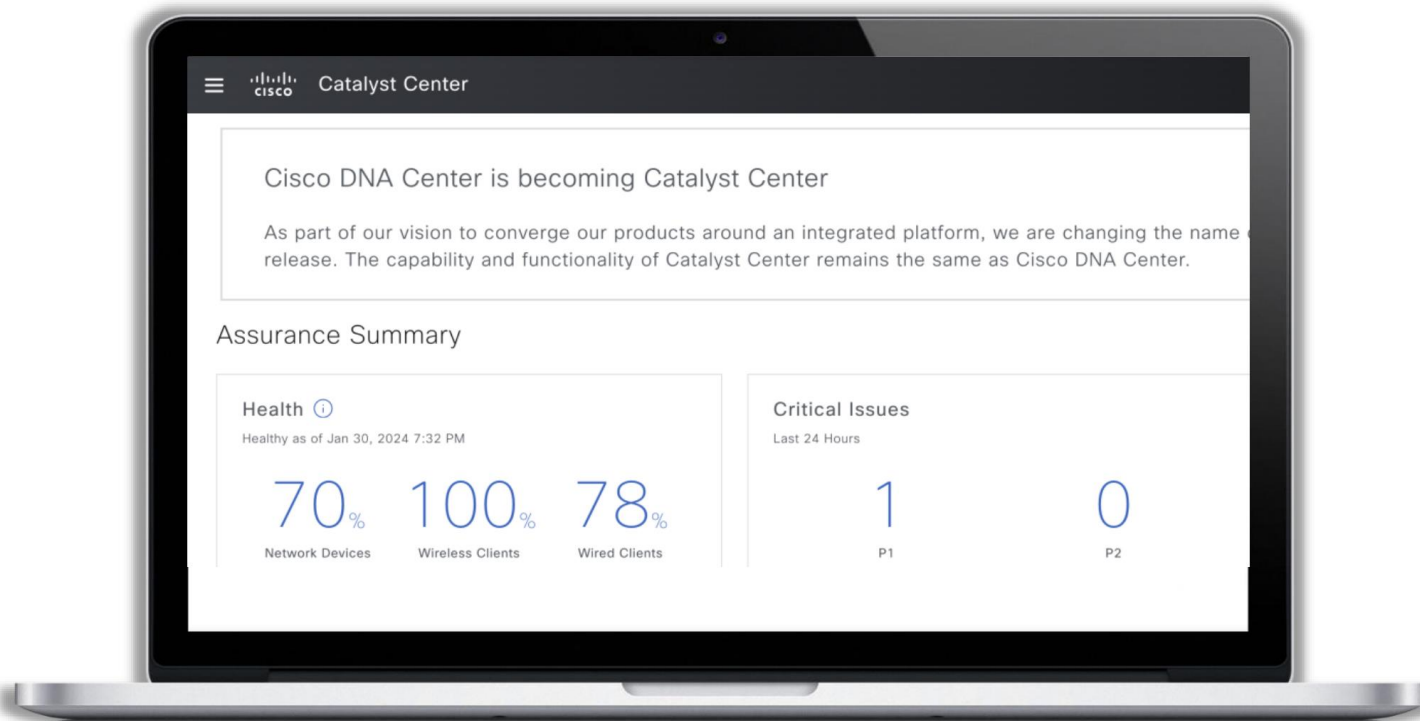


Catalyst Center



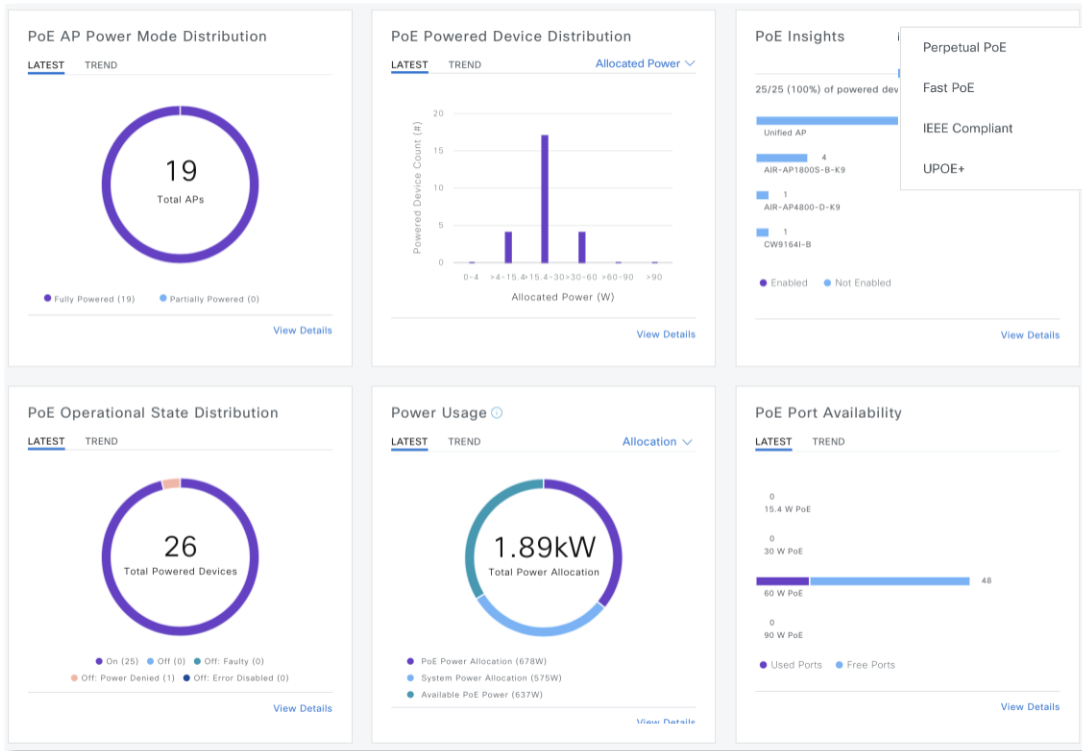
What is Cisco Catalyst Center?

- (formerly known as Cisco DNA Center)



PoE Power Management

In Catalyst Center Dashboard



- Monitor Catalyst switches and PoE enabled devices power consumption
- Provides information such as:
 - Used power, remaining power and power usage
- Monitor switch's power budget
- Relay on streaming telemetry from Catalyst switches to Catalyst Center

Catalyst Center PoE API Endpoints

<https://developer.cisco.com/docs/dna-center>

Devices

POE details

Operation Id: *pOEDetails*

Description: *Returns POE details for device.*

GET /dna/intent/api/v1/network-device/{deviceUuid}/poe

```
{
  "response": {
    "powerAllocated": "string",
    "powerConsumed": "string",
    "powerRemaining": "string"
  },
  "version": "string"
}
```

Devices

Returns POE interface details for the device.

Operation Id: *returnsPOEInterfaceDetailsForTheDevice_*

Description: *Returns POE interface details for the device, where deviceuuid is mandatory & accepts comma separated interface names which is optional and returns information for that particular interfaces where(operStatus = operationalStatus)*

GET /dna/intent/api/v1/network-device/{deviceUuid}/interface/poe-detail

```
{
  "version": "string",
  "response": [
    {
      "adminStatus": "string",
      "operStatus": "string",
      "interfaceName": "string",
      "maxPortPower": "string",
      "allocatedPower": "string",
      "portPowerDrawn": "string"
    }
  ]
}
```

Catalyst Center Python SDK

<https://dnacentersdk.readthedocs.io/en/latest/api/api.html>

```
poe_details(device_uuid, headers=None, **request_parameters) \[source\]
```

Returns POE details for device. .

- Parameters:**
- **device_uuid** (*basestring*) – deviceId path parameter. uuid of the device .
 - **headers** (*dict*) – Dictionary of HTTP Headers to send with the Request .
 - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
- Returns:** JSON response. Access the object's properties by using the dot notation or the bracket notation.
- Return type:** [MyDict](#)
- Raises:**
- **TypeError** – If the parameter types are incorrect.
 - **MalformedRequest** – If the request body created is invalid.
 - **ApiError** – If the DNA Center cloud returns an error.

Documentation Link:

<https://developer.cisco.com/docs/dna-center/#!/poe-details>

Catalyst Center Python SDK

<https://dnacentersdk.readthedocs.io/en/latest/api/api.html>

```
poe_interface_details(device_uuid, interface_name_list=None, headers=None, **request_parameters) \[source\]
```

Returns POE interface details for the device, where deviceuuid is mandatory & accepts comma separated interface names which is optional and returns information for that particular interfaces where(operStatus = operationalStatus) .

- Parameters:**
- **device_uuid** (*basestring*) – deviceId path parameter. uuid of the device .
 - **interface_name_list** (*basestring*) – interfaceNameList query parameter. comma separated interface names .
 - **headers** (*dict*) – Dictionary of HTTP Headers to send with the Request .
 - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns: JSON response. Access the object's properties by using the dot notation or the bracket notation.

Return type: [MyDict](#)

- Raises:**
- **[TypeError](#)** – If the parameter types are incorrect.
 - **[MalformedRequest](#)** – If the request body created is invalid.
 - **[ApiError](#)** – If the DNA Center cloud returns an error.

Documentation Link:

<https://developer.cisco.com/docs/dna-center/#!/poe-interface-details>

Catalyst Center Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/dnac/index.html>

- name: Get all Network Device Poe

```
cisco.dnac.network_device_poe_info:
  dnac_host: "{{dnac_host}}"
  dnac_username: "{{dnac_username}}"
  dnac_password: "{{dnac_password}}"
  dnac_verify: "{{dnac_verify}}"
  dnac_port: "{{dnac_port}}"
  dnac_version: "{{dnac_version}}"
  dnac_debug: "{{dnac_debug}}"
  headers: "{{my_headers | from_json}}"
  deviceId: string
register: result
```

- name: Get all Network Device Interface Poe

```
cisco.dnac.network_device_interface_poe_info:
  dnac_host: "{{dnac_host}}"
  dnac_username: "{{dnac_username}}"
  dnac_password: "{{dnac_password}}"
  dnac_verify: "{{dnac_verify}}"
  dnac_port: "{{dnac_port}}"
  dnac_version: "{{dnac_version}}"
  dnac_debug: "{{dnac_debug}}"
  headers: "{{my_headers | from_json}}"
  interfaceNameList: string
  deviceId: string
register: result
```

Catalyst Center Terraform Provider

<https://registry.terraform.io/providers/cisco-en-programmability/dnacenter/latest/docs>

```
data "dnacenter_network_device_poe" "example" {  
  provider      = dnacenter  
  device_uuid   = "string"  
}  
  
output "dnacenter_network_device_poe_example" {  
  value = data.dnacenter_network_device_poe.example.item  
}
```

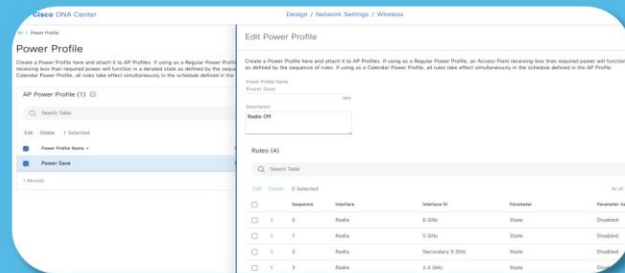
```
data "dnacenter_network_device_interface_poe" "example" {  
  provider      = dnacenter  
  device_uuid   = "string"  
  interface_name_list = "string"  
}  
  
output "dnacenter_network_device_interface_poe_example" {  
  value = data.dnacenter_network_device_interface_poe.example.items  
}
```

Save Energy with AP Power Save

We estimate that using Power Save on 2,000 APs (1.5 W) for 10 hours a day (10pm to 8am), saves up to ca.:

- 54,750 kWh energy
- 43,253 metric tones CO2 emissions over 5 years.

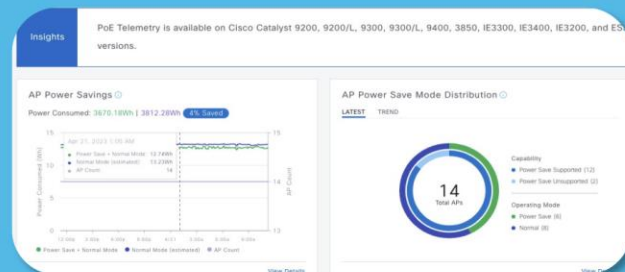
Design



Act



Report



Catalyst Center RF Profile API Endpoint

<https://developer.cisco.com/docs/dna-center>

Wireless

Create or Update RF profile

Operation Id: *createOrUpdateRFProfile*

Description: *Create or Update RF profile*

POST /dna/intent/api/v1/wireless/rf-profile

```
{
  "name": "string",
  "defaultRfProfile": true,
  "enableRadioTypeA": true,
  "enableRadioTypeB": true,
  "channelWidth": "string",
  "enableCustom": true,
  "enableBrownField": true,
  "radioTypeAProperties": {
    "parentProfile": "string",
    "radioChannels": "string",
    "dataRates": "string",
    "mandatoryDataRates": "string",
    "powerThresholdV1": 0,
    "rxSopThreshold": "string",
    "minPowerLevel": 0,
    "maxPowerLevel": 0
  },
  "radioTypeBProperties": {
    "parentProfile": "string",
    "radioChannels": "string",
    "dataRates": "string",
    "mandatoryDataRates": "string",
    "powerThresholdV1": 0,
    "rxSopThreshold": "string",
    "minPowerLevel": 0,
    "maxPowerLevel": 0
  },
  "radioTypeCProperties": {
    "parentProfile": "string",
    "radioChannels": "string",
    "dataRates": "string",
    "mandatoryDataRates": "string",
    "rxSopThreshold": "string",
    "minPowerLevel": 0,
    "maxPowerLevel": 0,
    "powerThresholdV1": 0
  },
  "enableRadioTypeC": true
}
```

Catalyst Center AP Provision API Endpoints

<https://developer.cisco.com/docs/dna-center>

Wireless

AP Provision

Operation Id: *aProvision*

Description: *Access Point Provision and ReProvision*

POST `/dna/intent/api/v1/wireless/ap-provision`

```
[
  {
    "rfProfile": "string",
    "deviceName": "string",
    "customApGroupName": "string",
    "customFlexGroupName": [ "string" ],
    "type": "string",
    "siteNameHierarchy": "string"
  }
]
```

Catalyst Center Python SDK

<https://dnacentersdk.readthedocs.io/en/latest/api/api.html>

```
create_or_update_rf_profile(channelWidth=None, defaultRfProfile=None, enableBrownField=None, enableCustom=None, enableRadioTypeA=None, enableRadioTypeB=None, enableRadioTypeC=None, name=None, radioTypeAProperties=None, radioTypeBProperties=None, radioTypeCProperties=None, headers=None, payload=None, active_validation=True, **request_parameters) \[source\]
```

Create or Update RF profile .

Parameters:

- **channelWidth** (*string*) – Wireless's Channel Width .
- **defaultRfProfile** (*boolean*) – Wireless's is Default Rf Profile .
- **enableBrownField** (*boolean*) – Wireless's Enable Brown Field .
- **enableCustom** (*boolean*) – Wireless's Enable Custom .
- **enableRadioTypeA** (*boolean*) – Wireless's Enable Radio Type A .
- **enableRadioTypeB** (*boolean*) – Wireless's Enable Radio Type B .
- **enableRadioTypeC** (*boolean*) – Wireless's Enable Radio Type C (6GHz) .
- **name** (*string*) – Wireless's RF Profile Name .
- **radioTypeAProperties** (*object*) – Wireless's radioTypeAProperties.
- **radioTypeBProperties** (*object*) – Wireless's radioTypeBProperties.
- **radioTypeCProperties** (*object*) – Wireless's radioTypeCProperties.
- **headers** (*dict*) – Dictionary of HTTP Headers to send with the Request .
- **payload** (*dict*) – A JSON serializable Python object to send in the body of the Request.
- **active_validation** (*bool*) – Enable/Disable payload validation. Defaults to True.
- ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).

Returns: JSON response. Access the object's properties by using the dot notation or the bracket notation.

Return type: `MyDict`

Raises:

- `TypeError` – If the parameter types are incorrect.
- `MalformedRequest` – If the request body created is invalid.
- `ApiError` – If the DNA Center cloud returns an error.

Documentation Link:

<https://developer.cisco.com/docs/dna-center/#!/create-or-update-rf-profile>

Catalyst Center Python SDK

<https://dnacentersdk.readthedocs.io/en/latest/api/api.html>

```
ap_provision(headers=None, payload=None, active_validation=True, **request_parameters) \[source\]
```

Access Point Provision and ReProvision .

- Parameters:**
- **headers** (*dict*) – Dictionary of HTTP Headers to send with the Request .
 - **payload** (*list*) – A JSON serializable Python object to send in the body of the Request.
 - **active_validation** (*bool*) – Enable/Disable payload validation. Defaults to True.
 - ****request_parameters** – Additional request parameters (provides support for parameters that may be added in the future).
- Returns:** JSON response. Access the object's properties by using the dot notation or the bracket notation.
- Return type:** *MyDict*
- Raises:**
- **TypeError** – If the parameter types are incorrect.
 - **MalformedRequest** – If the request body created is invalid.
 - **ApiError** – If the DNA Center cloud returns an error.

Documentation Link:

<https://developer.cisco.com/docs/dna-center/#!ap-provision>

Catalyst Center Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/dnac/index.html>

```
- name: Create
  cisco.dnac.wireless_rf_profile:
    dnac_host: "{{dnac_host}}"
    dnac_username: "{{dnac_username}}"
    dnac_password: "{{dnac_password}}"
    dnac_verify: "{{dnac_verify}}"
    dnac_port: "{{dnac_port}}"
    dnac_version: "{{dnac_version}}"
    dnac_debug: "{{dnac_debug}}"
    state: present
    channelWidth: string
    defaultRfProfile: true
    enableBrownField: true
    enableCustom: true
    enableRadioTypeA: true
    enableRadioTypeB: true
    enableRadioTypeC: true
    name: string
    radioTypeAProperties:
      dataRates: string
      mandatoryDataRates: string
      maxPowerLevel: 0
      minPowerLevel: 0
      parentProfile: string
      powerThresholdV1: 0
      radioChannels: string
      rxSopThreshold: string
```

```
    radioTypeBProperties:
      dataRates: string
      mandatoryDataRates: string
      maxPowerLevel: 0
      minPowerLevel: 0
      parentProfile: string
      powerThresholdV1: 0
      radioChannels: string
      rxSopThreshold: string
    radioTypeCProperties:
      dataRates: string
      mandatoryDataRates: string
      maxPowerLevel: 0
      minPowerLevel: 0
      parentProfile: string
      powerThresholdV1: 0
      radioChannels: string
      rxSopThreshold: string
```

Catalyst Center Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/dnac/index.html>

```
- name: Create
  cisco.dnac.wireless_provision_access_point:
    dnac_host: "{{dnac_host}}"
    dnac_username: "{{dnac_username}}"
    dnac_password: "{{dnac_password}}"
    dnac_verify: "{{dnac_verify}}"
    dnac_port: "{{dnac_port}}"
    dnac_version: "{{dnac_version}}"
    dnac_debug: "{{dnac_debug}}"
    headers: '{{my_headers | from_json}}'
    payload:
      - customApGroupName: string
        customFlexGroupName:
          - string
        deviceName: string
        rfProfile: string
        siteId: string
        siteNameHierarchy: string
        type: string
```

Catalyst Center Terraform Provider

<https://registry.terraform.io/providers/cisco-en-programmability/dnacenter/latest/docs>

```
resource "dnacenter_wireless_rf_profile" "example" {  
  provider = dnacenter  
  parameters {  
  
    channel_width = "string"  
    default_rf_profile = "false"  
    enable_brown_field = "false"  
    enable_custom = "false"  
    enable_radio_type_a = "false"  
    enable_radio_type_b = "false"  
    enable_radio_type_c = "false"  
    name = "string"  
    radio_type_a_properties {  
  
      data_rates = "string"  
      mandatory_data_rates = "string"  
      max_power_level = 1.0  
      min_power_level = 1.0  
      parent_profile = "string"  
      power_threshold_v1 = 1.0  
      radio_channels = "string"  
      rx_sop_threshold = "string"  
    }  
  }  
}
```

```
    radio_type_b_properties {  
  
      data_rates = "string"  
      mandatory_data_rates = "string"  
      max_power_level = 1.0  
      min_power_level = 1.0  
      parent_profile = "string"  
      power_threshold_v1 = 1.0  
      radio_channels = "string"  
      rx_sop_threshold = "string"  
    }  
    radio_type_c_properties {  
  
      data_rates = "string"  
      mandatory_data_rates = "string"  
      max_power_level = 1.0  
      min_power_level = 1.0  
      parent_profile = "string"  
      power_threshold_v1 = 1.0  
      radio_channels = "string"  
      rx_sop_threshold = "string"  
    }  
  }  
}  
  
output "dnacenter_wireless_rf_profile_example" {  
  value = dnacenter_wireless_rf_profile.example  
}
```

Catalyst Center Terraform Provider

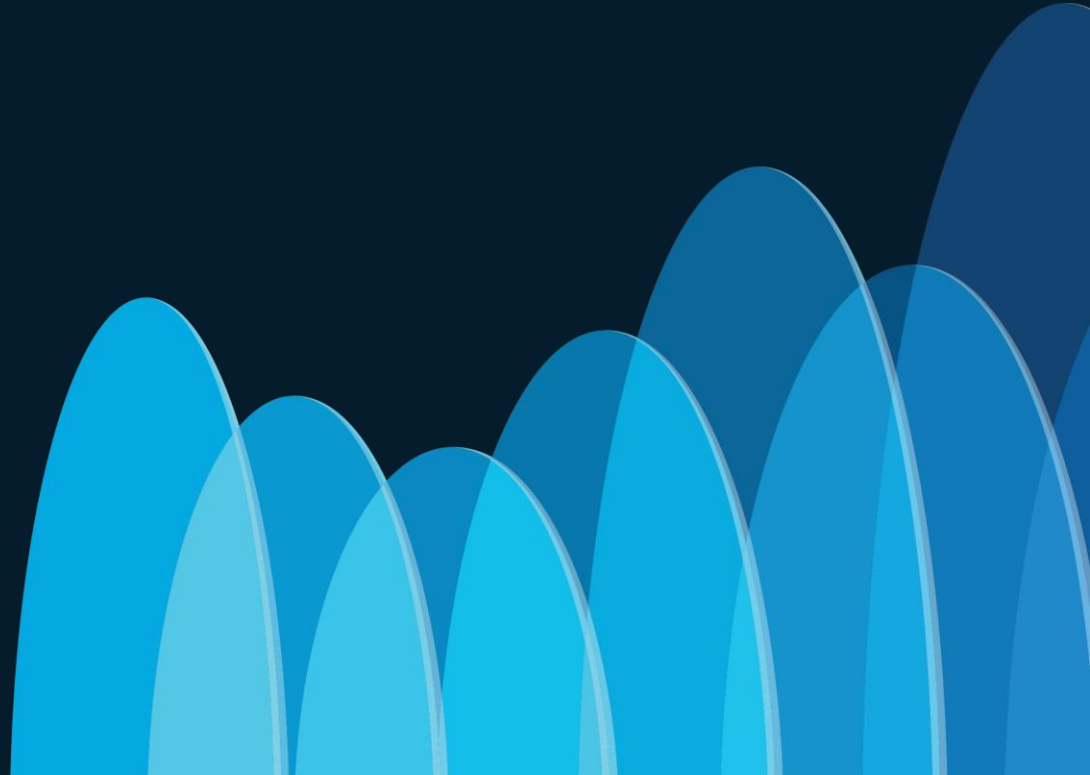
<https://registry.terraform.io/providers/cisco-en-programmability/dnacenter/latest/docs>

```
resource "dnacenter_wireless_provision_access_point" "example" {
  provider = dnacenter
  parameters {
    payload {
      custom_ap_group_name = "string"
      custom_flex_group_name = ["string"]
      device_name = "string"
      rf_profile = "string"
      site_id = "string"
      site_name_hierarchy = "string"
      type = "string"
      persistbapioutput = "false"
    }
  }
}

output "dnacenter_wireless_provision_access_point_example" {
  value = dnacenter_wireless_provision_access_point.example
}
```

Meraki

CISCO *Live!*



Powering off/on: Switch Port Schedules

Meraki MS Switches

[Show admin](#)

[Demo Networks](#)

Network

Meraki San Francisco SFO12

Secure Connect

Network-wide

Security & SD-WAN

Switching

Wireless

Systems Manager

Cameras

Sensors

Insight

Organization

Port schedules

Local time zone: America - Los Angeles (You can set this on [General](#))

used by [1 port](#)

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	<input type="text" value="enabled"/>	8:00 17:00						
Tuesday	<input type="text" value="enabled"/>	8:00 17:00						
Wednesday	<input type="text" value="enabled"/>	8:00 17:00						
Thursday	<input type="text" value="enabled"/>	8:00 17:00						
Friday	<input type="text" value="enabled"/>	8:00 17:00						
Saturday	<input type="text" value="enabled"/>	8:00 17:00						
Sunday	<input type="text" value="enabled"/>	8:00 17:00						

[Add a new port schedule](#)

Energy Saving Port Schedules

Create port schedules to turn off POE power to access points during off peak hours

Meraki Port Scheduling API Endpoint

<https://developer.cisco.com/meraki/api-v1/create-network-switch-port-schedule/>

switch

configure

portSchedules

Create Network Switch Port Schedule

Operation Id: *createNetworkSwitchPortSchedule*

Description: *Add a switch port schedule*

POST `/networks/{networkId}/switch/portSchedules`

```
{
  "name": "Weekdays schedule",
  "portSchedule": {
    "monday": {
      "active": true,
      "from": "9:00",
      "to": "17:00"
    },
    "tuesday": {
      "active": true,
      "from": "9:00",
      "to": "17:00"
    },
    ...
    "friday": {
      "active": true,
      "from": "9:00",
      "to": "17:00"
    },
    "saturday": {
      "active": false,
      "from": "0:00",
      "to": "24:00"
    },
    "sunday": {
      "active": false,
      "from": "0:00",
      "to": "24:00"
    }
  }
}
```

Meraki Dashboard API Python SDK

<https://github.com/meraki/dashboard-api-python/>

```
def createNetworkSwitchPortSchedule(self, networkId: str, name: str, **kwargs)
```

****Add a switch port schedule****

<https://developer.cisco.com/meraki/api-v1/create-network-switch-port-schedule>

- networkId (string): (required)
- name (string): The name for your port schedule. Required
- portSchedule (object): The schedule for switch port scheduling. Schedules are applied to days of the week.

When it's empty, default schedule with all days of a week are configured.

Any unspecified day in the schedule is added as a default schedule configuration of the day.

Meraki Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/meraki/index.html>

```
- name: Create
  cisco.meraki.networks_switch_port_schedules:
    meraki_api_key: "{{meraki_api_key}}"
    meraki_base_url: "{{meraki_base_url}}"
    meraki_single_request_timeout: "{{meraki_single_request_timeout}}"
    meraki_certificate_path: "{{meraki_certificate_path}}"
    meraki_requests_proxy: "{{meraki_requests_proxy}}"
    meraki_wait_on_rate_limit: "{{meraki_wait_on_rate_limit}}"
    meraki_nginx_429_retry_wait_time: "{{meraki_nginx_429_retry_wait_time}}"
    meraki_action_batch_retry_wait_time: "{{meraki_action_batch_retry_wait_time}}"
    meraki_retry_4xx_error: "{{meraki_retry_4xx_error}}"
    meraki_retry_4xx_error_wait_time: "{{meraki_retry_4xx_error_wait_time}}"
    meraki_maximum_retries: "{{meraki_maximum_retries}}"
    meraki_output_log: "{{meraki_output_log}}"
    meraki_log_file_prefix: "{{meraki_log_file_prefix}}"
    meraki_log_path: "{{meraki_log_path}}"
    meraki_print_console: "{{meraki_print_console}}"
    meraki_suppress_logging: "{{meraki_suppress_logging}}"
    meraki_simulate: "{{meraki_simulate}}"
    meraki_be_geo_id: "{{meraki_be_geo_id}}"
    meraki_use_iterator_for_get_pages: "{{meraki_use_iterator_for_get_pages}}"
    meraki_inherit_logging_config: "{{meraki_inherit_logging_config}}"
    state: present
    name: Weekdays schedule
    networkid: string
  ...
```

```
portSchedule:
  friday:
    active: true
    from: '9:00'
    to: '17:00'
  monday:
    active: true
    from: '9:00'
    to: '17:00'
  saturday:
    active: false
    from: 0:00
    to: '24:00'
  sunday:
    active: false
    from: 0:00
    to: '24:00'
  thursday:
    active: true
    from: '9:00'
    to: '17:00'
  tuesday:
    active: true
    from: '9:00'
    to: '17:00'
  wednesday:
    active: true
    from: '9:00'
    to: '17:00'
```

Meraki Terraform Provider

<https://registry.terraform.io/providers/cisco-open/meraki/latest/docs>

```
resource "meraki_networks_switch_port_schedules" "example" {
```

```
  name                = "Weekdays schedule"
  network_id          = "string"
  port_schedule       = {
```

```
    friday = {
```

```
      active      = true
      from        = "9:00"
      to          = "17:00"
    }
```

```
    monday = {
```

```
      active      = true
      from        = "9:00"
      to          = "17:00"
    }
```

```
    saturday = {
```

```
      active      = false
      from        = "0:00"
      to          = "24:00"
    }
```

```
    sunday = {
```

```
      active      = false
      from        = "0:00"
      to          = "24:00"
    }
```

```
    thursday = {
```

```
      active      = true
      from        = "9:00"
      to          = "17:00"
    }
```

```
    tuesday = {
```

```
      active      = true
      from        = "9:00"
      to          = "17:00"
    }
```

```
    wednesday = {
```

```
      active      = true
      from        = "9:00"
      to          = "17:00"
    }
```

```
  }
```

```
output "meraki_networks_switch_port_schedules_example" {
  value = meraki_networks_switch_port_schedules.example
}
```

Save Energy with Meraki Switch Port Schedules

Powering off 2,000 PoE devices (10W) for 10 hours a day is estimated to save up to ca.:

- 364,000 kWh energy
 - 76 metric tones CO2 emissions
- over 5 years.

Discover



Act



Report



Powering off/on: Wireless Networks

<https://developer.cisco.com/meraki/api-v1/update-network-wireless-rf-profile/>

wireless

configure

rfProfiles

Update Network Wireless Rf Profile

Operation Id: *updateNetworkWirelessRfProfile*

Description: *Updates specified RF profile for this network*

PUT

`/networks/{networkId}/wireless/rfProfiles/{rfProfileId}`

Powering off/on: Wireless Networks

```
{
  "name": "1234",
  "clientBalancingEnabled": true,
  "minBitrateType": "band",
  "bandSelectionType": "ap",
  "apBandSettings": {
    "bandOperationMode": "dual",
    "bands": {
      "enabled": [ "2.4", "5" ]
    },
    "bandSteeringEnabled": true
  },
  "twoFourGhzSettings": {
    "maxPower": 30,
    "minPower": 5,
    "minBitrate": 11,
    "validAutoChannels": [ 1, 6, 11 ],
    "axEnabled": true,
    "rxsop": -95
  },
  "fiveGhzSettings": {
    "maxPower": 30,
    "minPower": 8,
    "minBitrate": 12,
    "validAutoChannels": [
      36,
      40,
      ...
      161,
      165
    ],
    "channelWidth": "auto",
    "rxsop": -95
  },
}
```

```
"sixGhzSettings": {
  "maxPower": 30,
  "minPower": 8,
  "minBitrate": 12,
  "validAutoChannels": [
    1,
    5,
    9,
    ...
    229,
    233
  ],
  "channelWidth": "auto",
  "rxsop": -95
},
"transmission": { "enabled": true },
"perSsidSettings": {
  "0": {
    "minBitrate": 11,
    "bandOperationMode": "dual",
    "bands": {
      "enabled": [ "2.4", "5" ]
    },
    "bandSteeringEnabled": true
  },
  "1": {
    "minBitrate": 11,
    "bandOperationMode": "dual",
    "bands": {
      "enabled": [ "2.4", "5" ]
    },
    "bandSteeringEnabled": true
  },
  ...
}
```

```
"13": {
  "minBitrate": 11,
  "bandOperationMode": "dual",
  "bands": {
    "enabled": [ "2.4", "5" ]
  },
  "bandSteeringEnabled": true
},
"14": {
  "minBitrate": 11,
  "bandOperationMode": "dual",
  "bands": {
    "enabled": [ "2.4", "5" ]
  },
  "bandSteeringEnabled": true
},
"flexRadios": {
  "byModel": [
    {
      "model": "MR34",
      "bands": [ "5" ]
    }
  ]
}
}
```

Meraki Dashboard API Python SDK

<https://github.com/meraki/dashboard-api-python/>

```
def updateNetworkWirelessRfProfile(self, networkId: str, rfProfileId: str, **kwargs)
```

****Updates specified RF profile for this network****

<https://developer.cisco.com/meraki/api-v1/#!/update-network-wireless-rf-profile>

- networkId (string): (required)
- rfProfileId (string): (required)
- name (string): The name of the new profile. Must be unique.
- clientBalancingEnabled (boolean): Steers client to best available access point. Can be either true or false.
- minBitrateType (string): Minimum bitrate can be set to either 'band' or 'ssid'.
- bandSelectionType (string): Band selection can be set to either 'ssid' or 'ap'.
- apBandSettings (object): Settings that will be enabled if selectionType is set to 'ap'.
- twoFourGhzSettings (object): Settings related to 2.4Ghz band
- fiveGhzSettings (object): Settings related to 5Ghz band
- transmission (object): Settings related to radio transmission.
- perSsidSettings (object): Per-SSID radio settings by number.

Meraki Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/meraki/index.html>

```
- name: Create
  cisco.meraki.networks_wireless_rf_profiles:
    meraki_api_key: "{{meraki_api_key}}"
    meraki_base_url: "{{meraki_base_url}}"
    state: present
    apBandSettings:
      bandOperationMode: dual
      bandSteeringEnabled: true
    bandSelectionType: ap
    clientBalancingEnabled: true
    fiveGhzSettings:
      channelWidth: auto
      maxPower: 30
      minBitrate: 12
      minPower: 8
      rxSop: -95
      validAutoChannels:
        - 36
        - 40
        - 165
    minBitrateType: band
    name: Main Office
    networkId: string
    perSsidSettings:
      '0':
        bandOperationMode: dual
        bandSteeringEnabled: true
        minBitrate: 11
```

```
'1':
  bandOperationMode: dual
  bandSteeringEnabled: true
  minBitrate: 11
'13':
  bandOperationMode: dual
  bandSteeringEnabled: true
  minBitrate: 11
'14':
  bandOperationMode: dual
  bandSteeringEnabled: true
  minBitrate: 11
sixGhzSettings:
  channelWidth: auto
  maxPower: 30
  minBitrate: 12
  minPower: 8
  rxSop: -95
  validAutoChannels:
    - 1
    - 5
    - 233
transmission:
  enabled: true
twoFourGhzSettings:
  axEnabled: true
  maxPower: 30
  minBitrate: 11
  minPower: 5
  rxSop: -95
  validAutoChannels:
    - 1
    - 6
    - 11
```

Meraki Terraform Provider

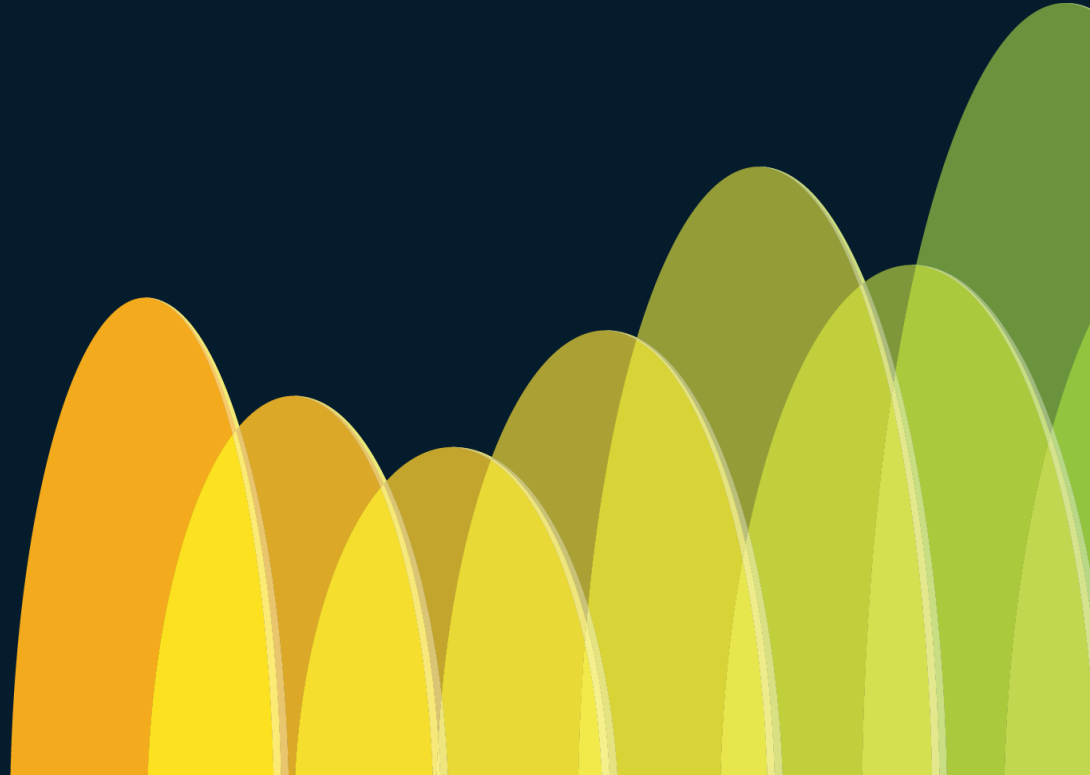
<https://registry.terraform.io/providers/cisco-open/meraki/latest/docs>

```
resource "meraki_networks_wireless_rf_profiles" "example" {  
  
  ap_band_settings = {  
  
    band_operation_mode = "dual"  
    band_steering_enabled = true  
  }  
  band_selection_type = "ap"  
  client_balancing_enabled = true  
  five_ghz_settings = {  
  
    channel_width = "auto"  
    max_power = 30  
    min_bitrate = 12  
    min_power = 8  
    rx_sop = -95  
    valid_auto_channels = [36, 40, 165]  
  }  
  min_bitrate_type = "band"  
  name = "Main Office"  
  network_id = "string"  
  per_ssid_settings = {  
  
    status_0 = {  
  
      band_operation_mode = "dual"  
      band_steering_enabled = true  
      min_bitrate = 11  
    }  
  }  
}
```

```
    status_14 = {  
  
      band_operation_mode = "dual"  
      band_steering_enabled = true  
      min_bitrate = 11  
    }  
  }  
  six_ghz_settings = {  
  
    channel_width = "auto"  
    max_power = 30  
    min_bitrate = 12  
    min_power = 8  
    rx_sop = -95  
    valid_auto_channels = [1, 5, 233]  
  }  
  transmission = {  
  
    enabled = true  
  }  
  two_four_ghz_settings = {  
  
    ax_enabled = true  
    max_power = 30  
    min_bitrate = 11  
    min_power = 5  
    rx_sop = -95  
    valid_auto_channels = [1, 6, 11]  
  }  
}
```

```
output "meraki_networks_wireless_rf_profiles_example" {  
  value = meraki_networks_wireless_rf_profiles.example  
}
```

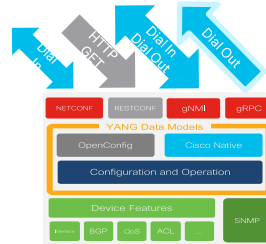
Cisco IOS XE



Catalyst Powers Telemetry for Smart Buildings



C9300-24HX - Stack

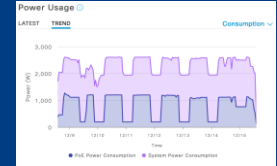


Telemetry

Telemetry

Telemetry

Cisco Catalyst Center



Grafana



25.2 kWh

Which YANG APIs are used?

#	Feature	YANG Data Model	Subscription Xpath Configuration
1	Temperature	Cisco-IOS-XE-environment-oper	filter xpath /environment-ios-xe-oper:environment-sensors
2	Fan Speed	openconfig-platform.yang	filter xpath /oc-platform:components
3	PowerSupply	Cisco-IOS-XE-platform-oper.yang	filter xpath /platform-ios-xe-oper:components/component/platform-properties/platform-property
4	POE Module	Cisco-IOS-XE-poe-oper.yang	filter xpath /poe-ios-xe-oper:poe-oper-data/poe-module
5	POE Port	Cisco-IOS-XE-poe-oper.yang	filter xpath /poe-ios-xe-oper:poe-oper-data/poe-port-detail
6	POE Stack	Cisco-IOS-XE-poe-oper.yang	filter xpath /poe-ios-xe-oper:poe-oper-data/poe-stack
7	POE Switch	Cisco-IOS-XE-poe-oper.yang	filter xpath /poe-ios-xe-oper:poe-oper-data/poe-switch
8	Memory	Cisco-IOS-XE-memory-oper.yang	filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization
9	CPU	Cisco-IOS-XE-process-cpu-oper.yang	filter xpath /process-memory-ios-xe-oper:memory-usage-processes

YANG Suite

<https://developer.cisco.com/yangsuite/>

The screenshot displays the Cisco YANG Suite interface. On the left is a navigation menu with options: Admin, Setup, Analytics, Explore, Protocols, and Help. The main content area is titled 'Explore YANG Models' and shows a tree structure of YANG models. The selected model is 'Cisco-IOS-XE-poe-oper', and the specific node being viewed is 'available-power'. The 'Node Properties' panel on the right lists various attributes for this node, including its name, datatype, description, module, revision, XPath, prefix, namespace, schema node ID, min/max values, fraction digits, access, and operations. Below the properties, a 'Reference URL' is provided, linking to the IETF RFC 6020 section 7.6, which discusses the 'leaf' statement. The interface also includes search bars for XPaths and nodes, and buttons for expanding all nodes and loading modules.

Node Properties

Name	available-power
Nodetype	leaf
Datatype	decimal64
Description	Available power of module
Module	Cisco-IOS-XE-poe-oper
Revision	2023-07-01
Xpath	/poe-oper-data/poe-module/available-power
Prefix	poe-ios-xe-oper
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper
Schema Node Id	/poe-oper-data/poe-module/available-power
Min	-92233720368547758.08
Max	92233720368547758.07
Fraction Digits	2
Access	read-only
Operations	• "get"

Reference URL:
<https://tools.ietf.org/html/rfc6020#section-7.6>

7.6. The Leaf Statement

The "leaf" statement is used to define a leaf node in the schema tree. It takes one argument, which is an identifier, followed by a block of substatements that holds detailed leaf information.

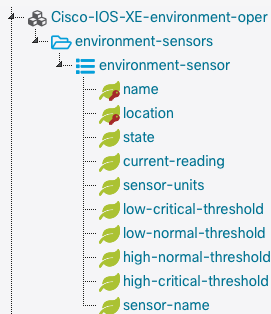
A leaf node has a value, but no child nodes in the data tree. Conceptually, the value in the data tree is always in the canonical form (see Section 9.1).

A leaf node exists in zero or one instances in the data tree.

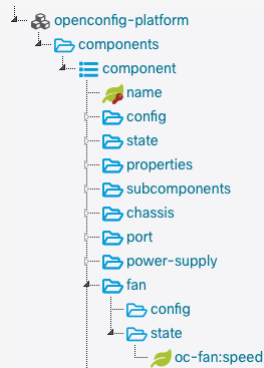
The "leaf" statement is used to define a scalar variable of a particular built-in or derived type.

Which YANG APIs are used ?

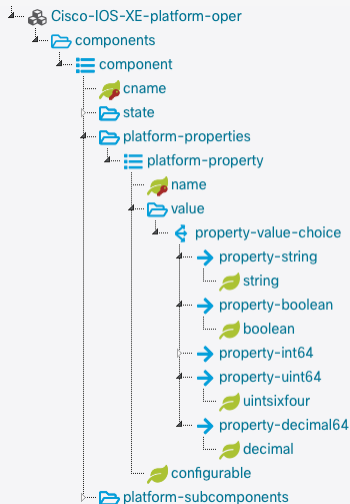
Temperature current-reading



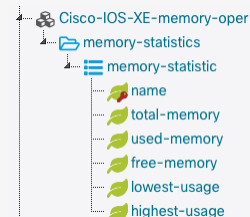
Fan state speed



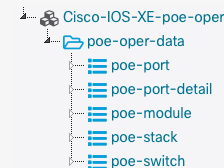
PowerSupply component states



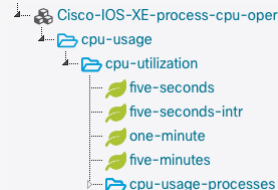
Memory usage



POE



CPU



PoE Telemetry

<https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/17101/Cisco-IOS-XE-poe-oper.yang>

Telemetry configuration

```
no telemetry ietf subscription 69001
telemetry ietf subscription 69001
filter xpath /poe-ios-xe-oper:poe-oper-data/poe-port-detail
receiver ip address 10.85.134.66 57508 protocol grpc-tcp
source-address 10.85.134.70
source-vrf Mgmt-vrf
stream yang-push
update-policy periodic 3000
encoding encode-kvgpb
```

```
no telemetry ietf subscription 69002
telemetry ietf subscription 69002
filter xpath /poe-ios-xe-oper:poe-oper-data/poe-switch
receiver ip address 10.85.134.66 57508 protocol grpc-tcp
source-address 10.85.134.70
source-vrf Mgmt-vrf
stream yang-push
update-policy periodic 3000
encoding encode-kvgpb
```

```
no telemetry ietf subscription 69003
telemetry ietf subscription 69003
filter xpath /poe-ios-xe-oper:poe-oper-data/poe-stack
receiver ip address 10.85.134.66 57508 protocol grpc-tcp
source-address 10.85.134.70
source-vrf Mgmt-vrf
stream yang-push
update-policy periodic 3000
encoding encode-kvgpb
```

YANG data model

poE-port-detail	Name	poE-port-detail
intf-name	Nodetype	list
power-used	Description	List of PoE interfaces, keyed by interface name
pd-class	Module	Cisco-IOS-XE-poe-oper
device-detected	Revision	2022-07-01
device-name	Xpath	/poe-oper-data/poe-port-detail
police	Prefix	poe-ios-xe-oper
power-admin-max	Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper
power-from-pse	Schema Node Id	/poe-oper-data/poe-port-detail
power-to-pd	Keys	• "intf-name"
	Access	read-only
	Operations	• "get"
poE-switch	Name	poE-switch
switch-num	Nodetype	list
power-budget	Description	List of PoE switches, keyed by switch number
power-allocated	Module	Cisco-IOS-XE-poe-oper
low-port-priority	Revision	2022-07-01
high-port-priority	Xpath	/poe-oper-data/poe-switch
switch-priority	Prefix	poe-ios-xe-oper
port-one-status	Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper
port-two-status	Schema Node Id	/poe-oper-data/poe-switch
	Keys	• "switch-num"
	Access	read-only
	Operations	• "get"
poE-stack	Name	poE-stack
power-stack-name	Nodetype	list
mode	Description	List of PoE stacks, keyed by stack name
topolgy	Module	Cisco-IOS-XE-poe-oper
total-power	Revision	2022-11-01
rsvd-power	Xpath	/poe-oper-data/poe-stack
alloc-power	Prefix	poe-ios-xe-oper
unused-power	Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper
num-sw	Schema Node Id	/poe-oper-data/poe-stack
num-ps	Keys	• "power-stack-name"
	Access	read-only
	Operations	• "get"



Cisco IOS XE Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/ios/index.html>

```
- name: Cisco show power example
  hosts: switches
  gather_facts: false

tasks:
- name: run show power inline on remote devices
  cisco.ios.ios_command:
    commands: show power inline
    register: output
- name: print output
  debug:
    var: output.stdout_lines
```

Cisco IOS XE Ansible Collection

<https://docs.ansible.com/ansible/latest/collections/cisco/ios/index.html>

```
TASK [print output] *****
ok: [10.10.20.181] => {
  "output.stdout_lines": [
    [
      "Module", "Available", "Used", "Remaining",
      " ", " ", " (Watts)", " (Watts)", " (Watts)",
      "-----", "-----", "-----", "-----",
      "1", " ", "370.0", "0.0", "370.0",
      "Interface Admin Oper Power Device Class Max",
      " ", " ", " ", " (Watts)", " ",
      "-----", "-----", "-----", "-----", "-----",
      "Gi1/0/1 auto off 0.0 n/a n/a 30.0",
      "Gi1/0/2 auto off 0.0 n/a n/a 30.0",
      "Gi1/0/3 auto off 0.0 n/a n/a 30.0",
      "Gi1/0/4 auto off 0.0 n/a n/a 30.0",
      "...",
      "Gi1/0/23 auto off 0.0 n/a n/a 30.0",
      "Gi1/0/24 auto off 0.0 n/a n/a 30.0",
      "-----", "-----",
      "Totals: 0 on 0.0",
      ""
    ]
  ]
}
```

```
PLAY RECAP *****
10.10.20.181 : ok=1 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
```

Cisco IOS XE Terraform Provider

<https://registry.terraform.io/providers/CiscoDevNet/iosxe/latest/docs>

```
data "iosxe_restconf" "example" {  
  path = "Cisco-IOS-XE-poe-oper:poe-oper-data"  
}
```

```
data "iosxe_restconf" "example" {  
  attributes = {  
    "poe-module" = jsonencode(  
      [  
        {  
          available-power = "370.0"  
          chassis-num = 1  
          free-ports = 24  
          if-max-power = "30000.0"  
          module = 1  
          module-type = "module-type-fixed"  
          num-ports = 24  
          power-pool-idx = 1  
          remaining-power = "370.0"  
          tot-consumed-pwr = "0.0"  
          tot-consumed-pwr-valid = [  
            null,  
          ]  
          used-ports = 0  
          used-power = "0.0"  
        }  
      ]  
    )  
  }  
  id = "Cisco-IOS-XE-poe-oper:poe-oper-data"  
  path = "Cisco-IOS-XE-poe-oper:poe-oper-data"  
}
```

Resources



Catalyst Center

- <https://developer.cisco.com/docs/dna-center>
- <https://dnacentersdk.readthedocs.io/en/latest/api/api.html>
- <https://docs.ansible.com/ansible/latest/collections/cisco/dnac/index.html>
- <https://registry.terraform.io/providers/cisco-en-programmability/dnacenter/latest/docs>

Meraki

- <https://developer.cisco.com/meraki/api-v1>
- <https://github.com/meraki/dashboard-api-python/>
- <https://docs.ansible.com/ansible/latest/collections/cisco/meraki/index.html>
- <https://registry.terraform.io/providers/cisco-open/meraki/latest/docs>

IOS XE

- <https://developer.cisco.com/yangsuite/>
- <https://docs.ansible.com/ansible/latest/collections/cisco/ios/index.html>
- <https://registry.terraform.io/providers/CiscoDevNet/iosxe/latest/docs>

Keynote Deep Dives

Wednesday
10:30am – 11:30am



Experiences Amplified:
How AI Can Fuel Better Employee and Customer Experiences

Level 1
Room 106



Smart, Secure, Seamless:
Transforming Experiences with Next-Generation Networking

Level 2
Room 204



Harness a Bold New Era:
Transform Data Centre and Service Provider Connectivity

Level 2
Room 203



Securing User to Application and Everything in Between

Level 2
Melbourne Room 2



Unlocking Digital Resilience through Unified Observability

The HUB
Centre Stage

Complete Your Session Evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to claim a **Cisco Live T-Shirt**.



Complete your surveys in the **Cisco Live mobile app**.



Continue your education

- Visit the Cisco Stand 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

Contact me at: @aidevnet



Thank you

CISCO *Live!*

#CiscoLiveAPJC

CISCO *Live!*

GO BEYOND

#CiscoLiveAPJC