

CISCO *Live!*

GO BEYOND

#CiscoLiveAPJC



Programmability on Catalyst 9000 enabling Sustainability

Telemetry Dashboards

Story DeWeese, Technical Marketing
[@StoryDeWeese](#)
BRKAPP-2245

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 November 15, 2024.



[https://ciscolive.ciscoevents.com/
ciscolivebot/#BRKAPP-2245](https://ciscolive.ciscoevents.com/ciscolivebot/#BRKAPP-2245)



Agenda

- **Current Situation**
 - Sustainability in Today's complex networks
- **Challenges & Impacts**
- **Demo**
- **Solutions**
 - Programmability with IOS XE
 - Model Driven Telemetry (MDT)
 - TIG Stack
 - Demo + YANG Suite
- **Benefits**
- **Next Steps**
- **Tooling**
- **Resources**

Current Situation

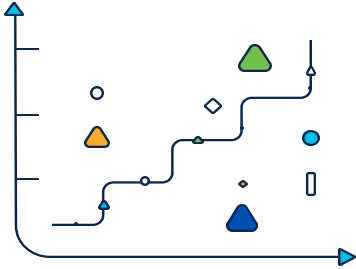
Sustainability in today's complex networks

Current Situation

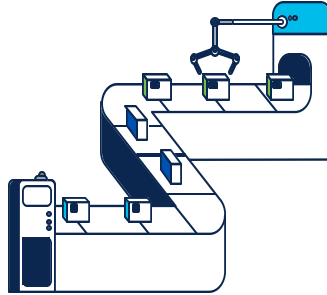
- Complex Networks
- Limited Network Visibility
- Limited Device Tracking over time

Challenges & Impacts

Challenges & Impacts



↑ Energy Consumption

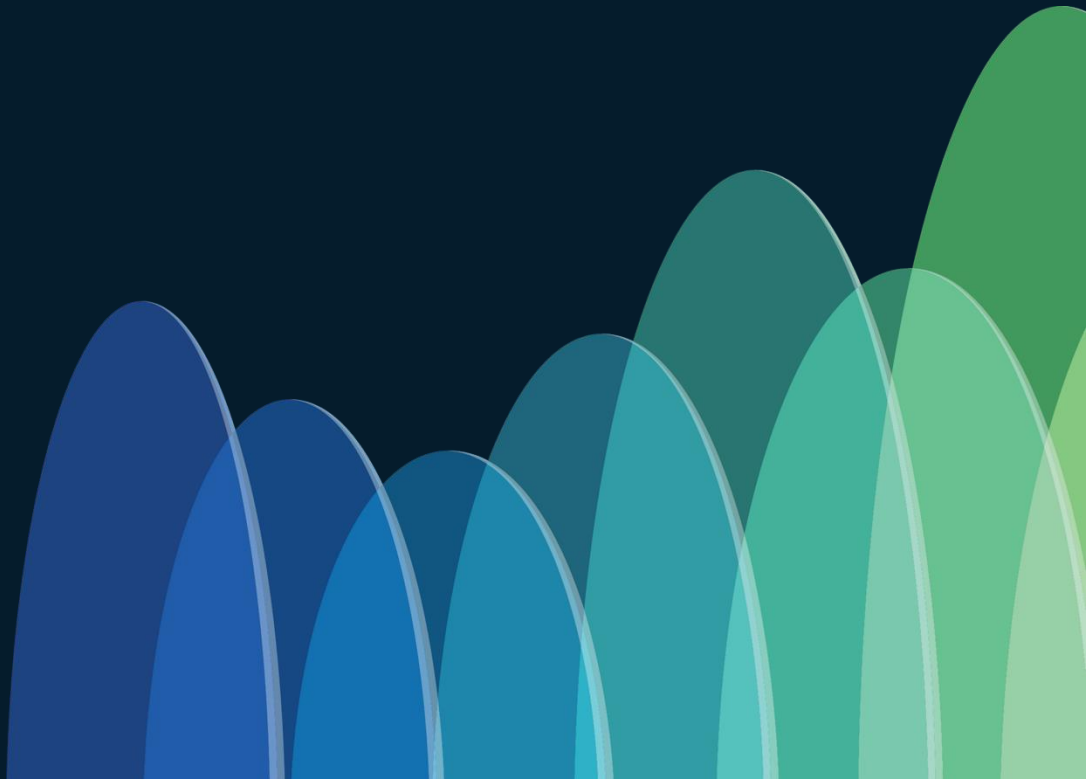


↑ Operational Costs



↓ Sustainability

In an Ideal World...



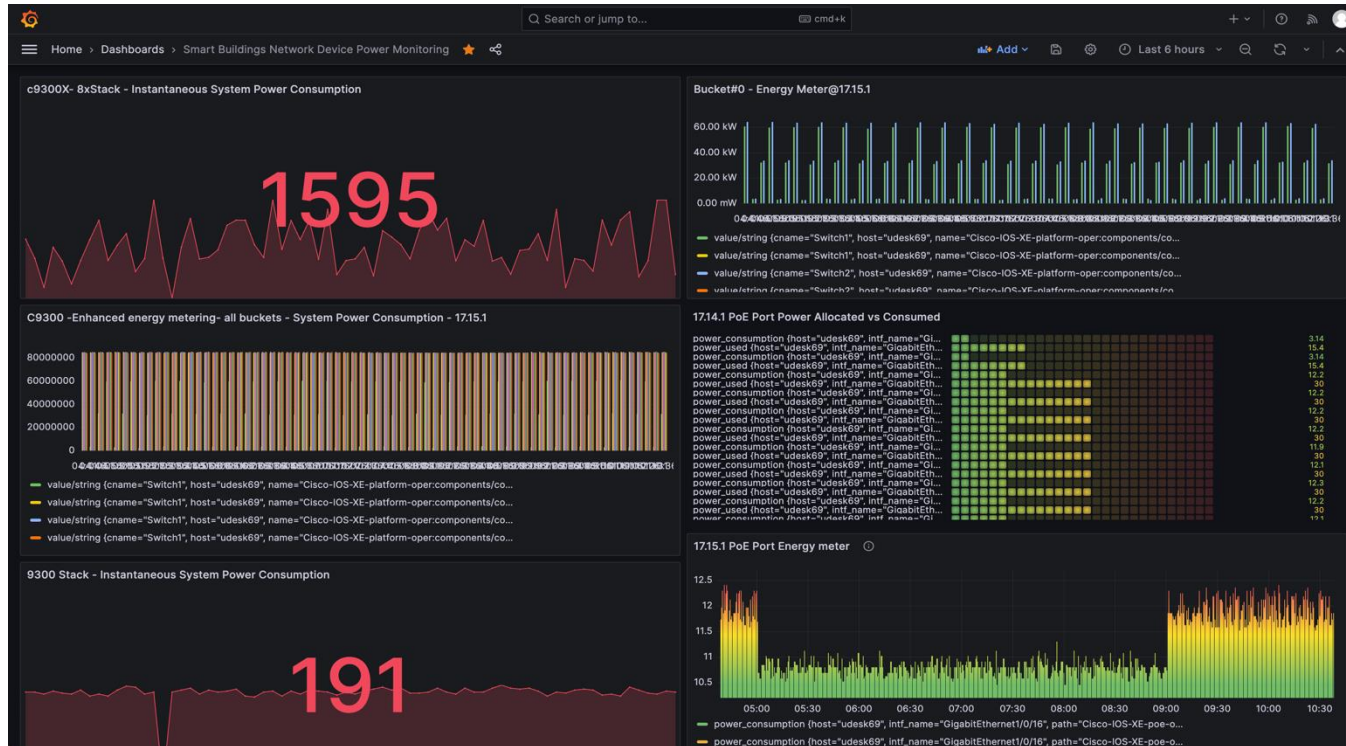
How can we save energy and cost?



How can we maintain more sustainable networks?

In our ideal world

Visibility into the amount of energy we are consuming





Solutions

Programmability with IOS XE
Model Driven Telemetry (MDT)

Why do we use programmability?



Programmability solutions for complex networks



Efficient

Managing infrastructure as code while lowering OPEX



Consistent

Repeatable and precise intent-based outcomes



Scalable

Reduce time and effort with large network automation requirements



Secure

Secure APIs and integrations



Seamless

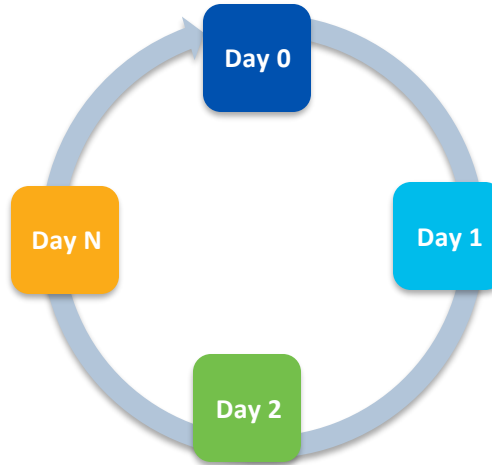
Create value integrating with 3rd party platforms



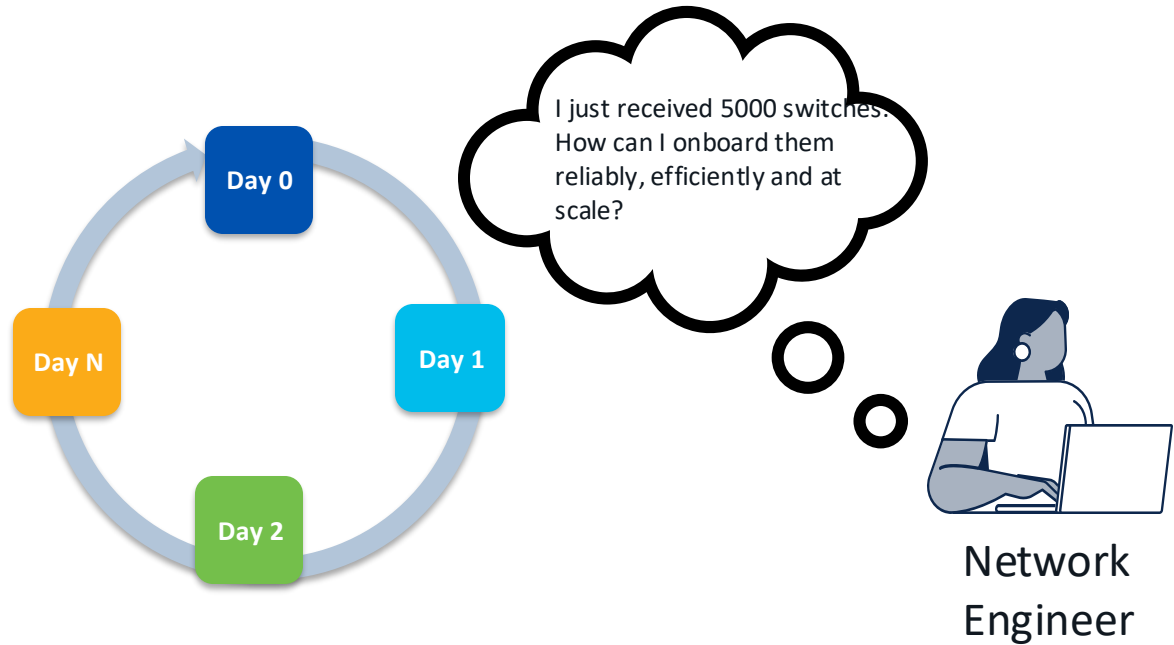
Smart

Real time event notifications, AI/ML capabilities, and streaming telemetry

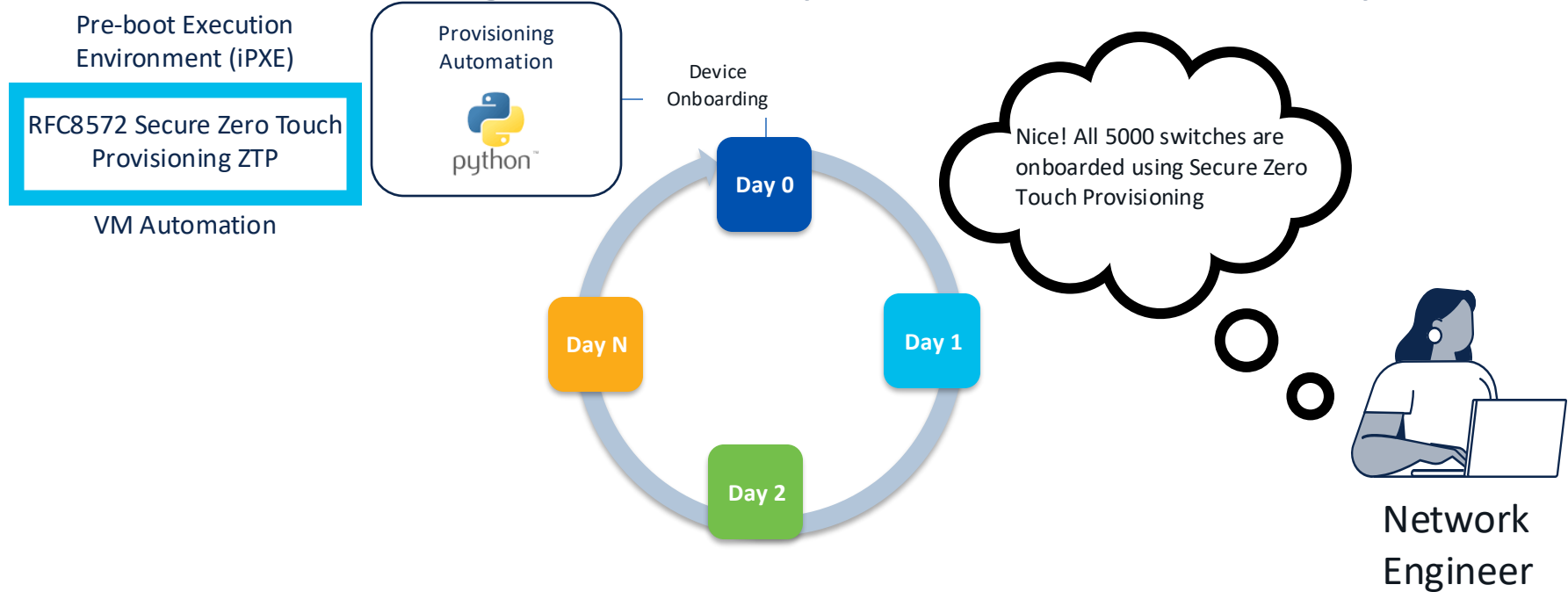
Cisco IOS XE Programmability & Automation Lifecycle



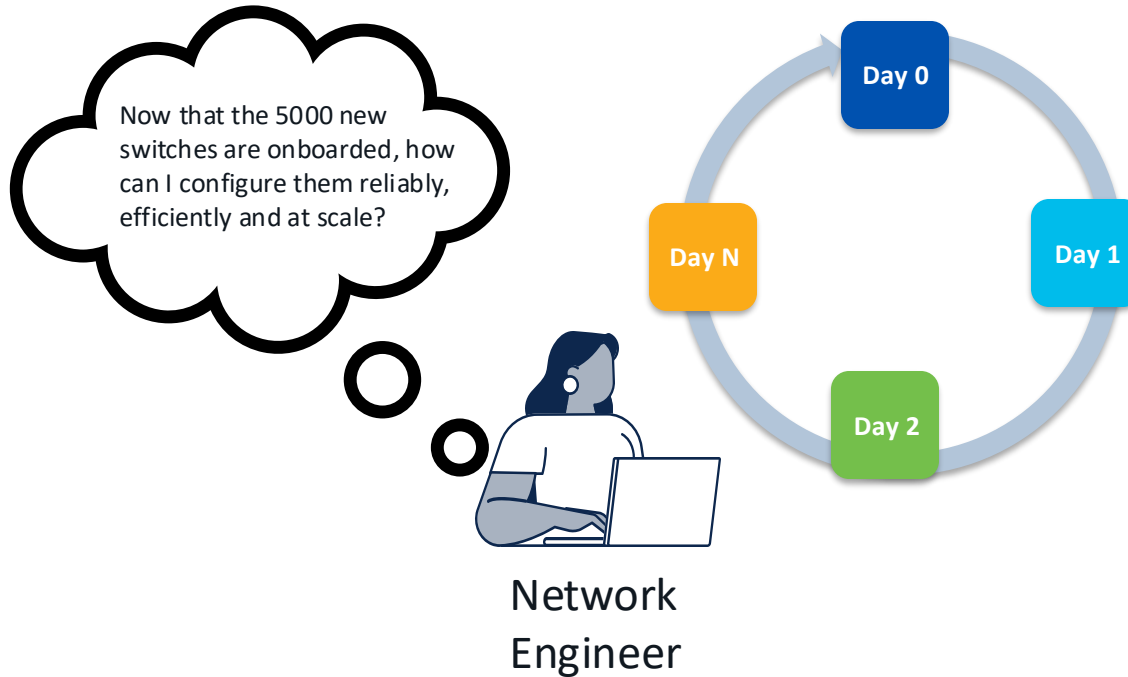
Cisco IOS XE Programmability & Automation Lifecycle



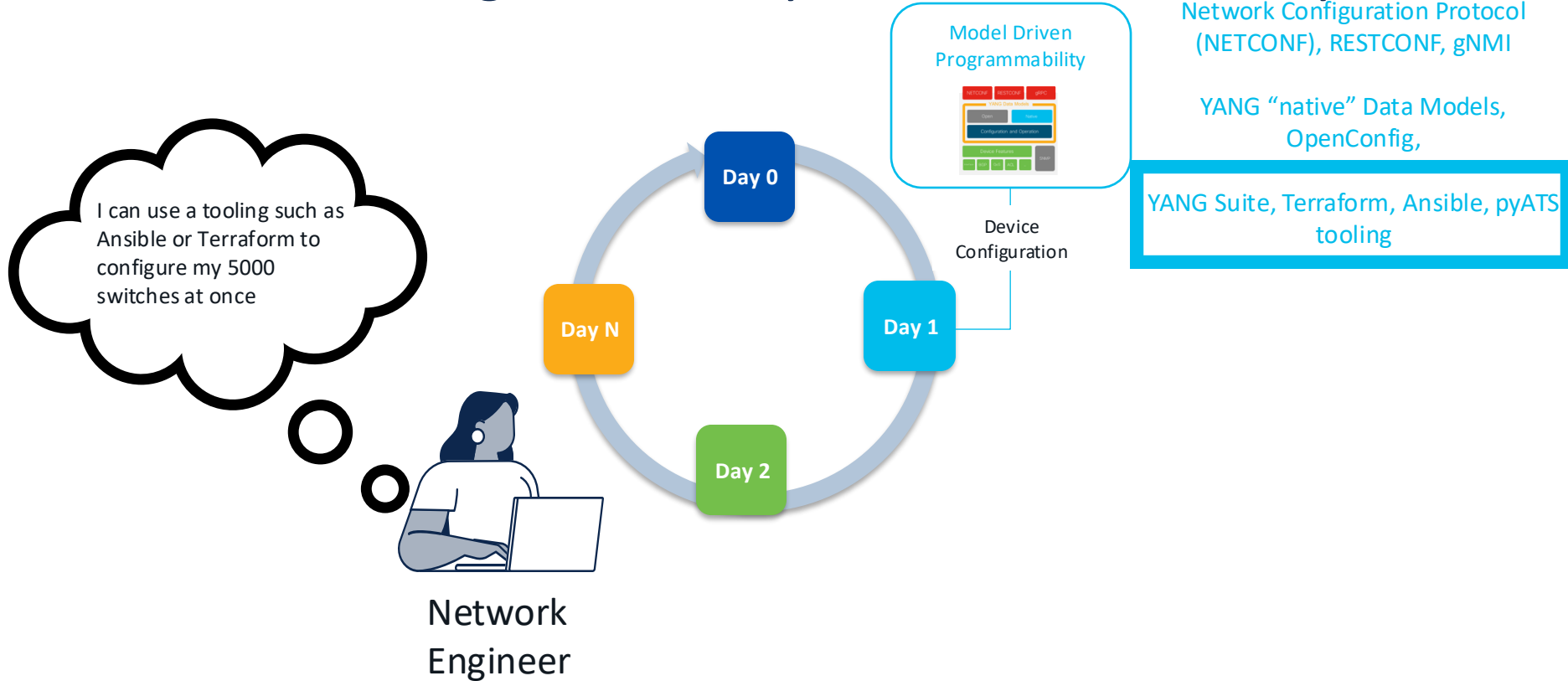
Cisco IOS XE Programmability & Automation Lifecycle



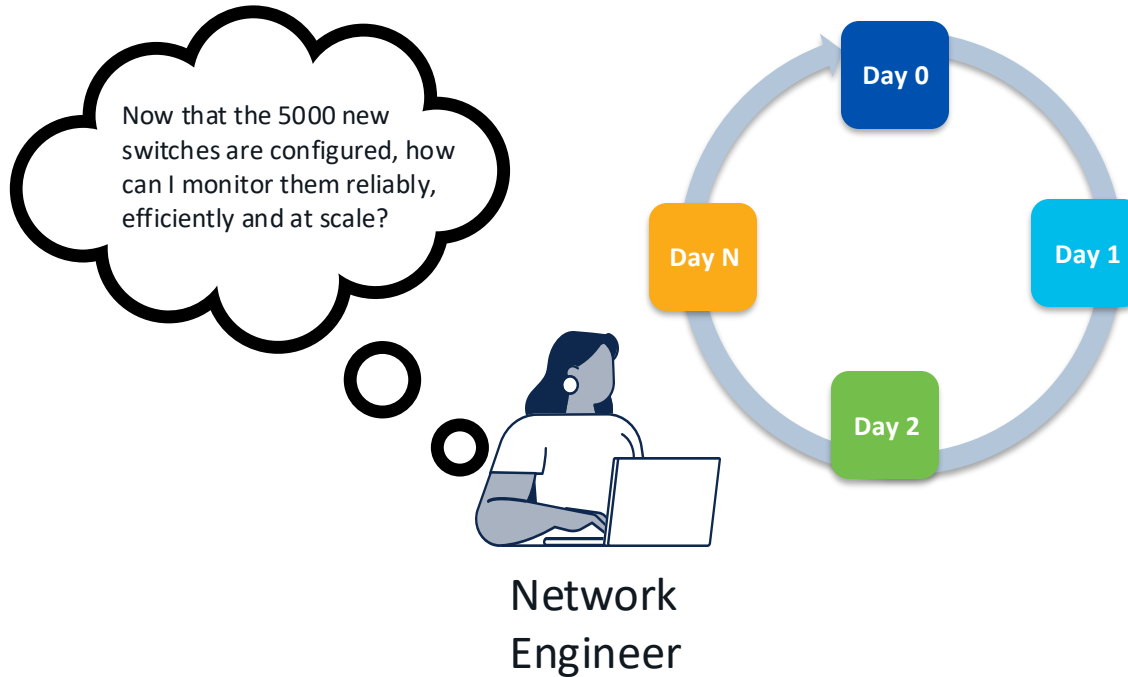
Cisco IOS XE Programmability & Automation Lifecycle



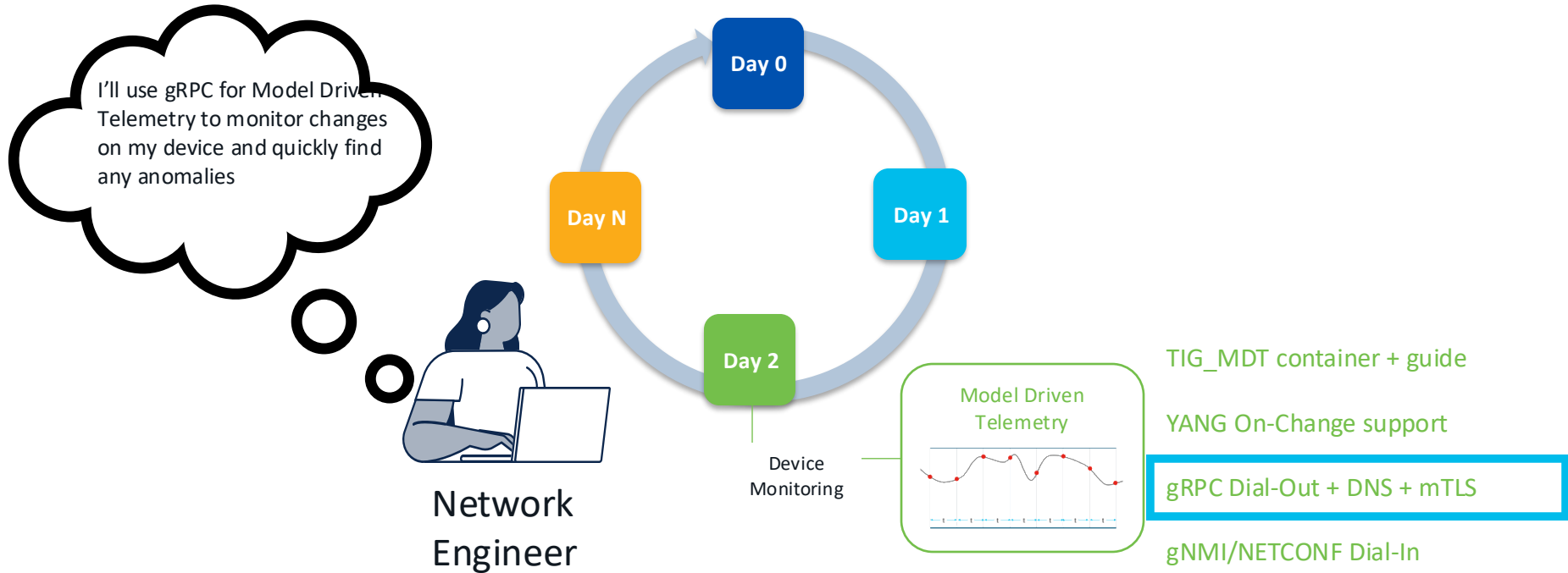
Cisco IOS XE Programmability & Automation Lifecycle



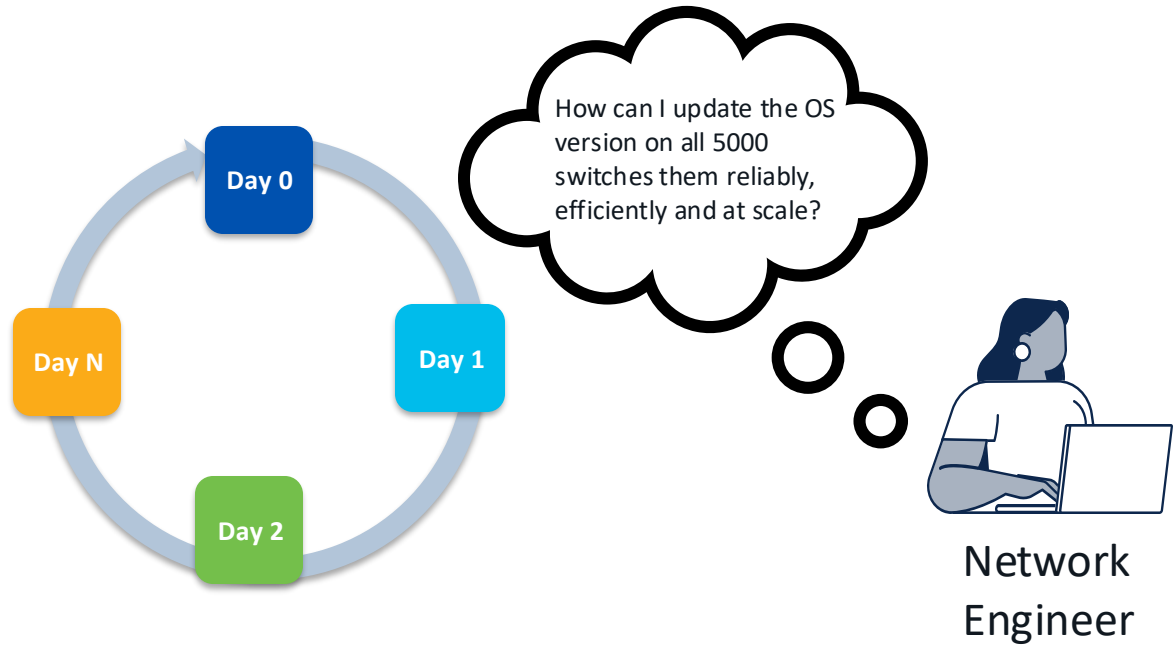
Cisco IOS XE Programmability & Automation Lifecycle



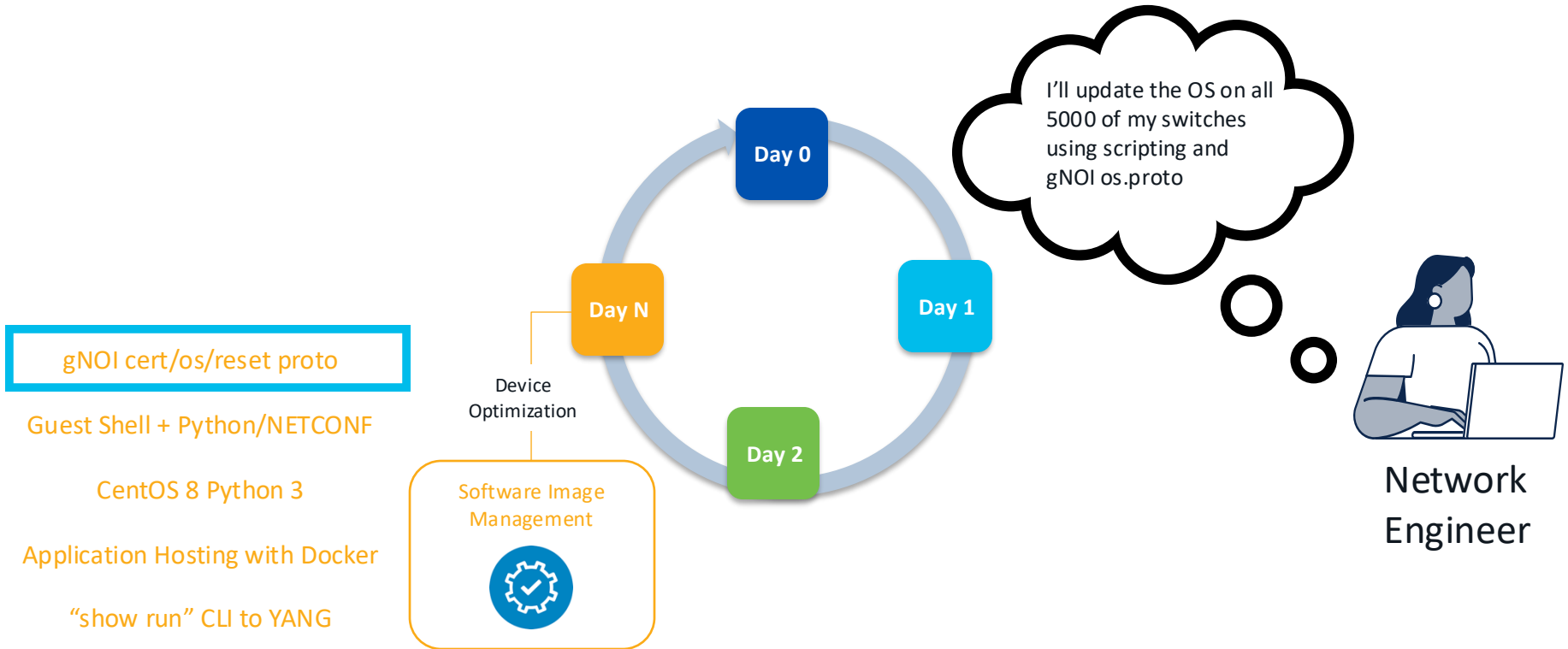
Cisco IOS XE Programmability & Automation Lifecycle



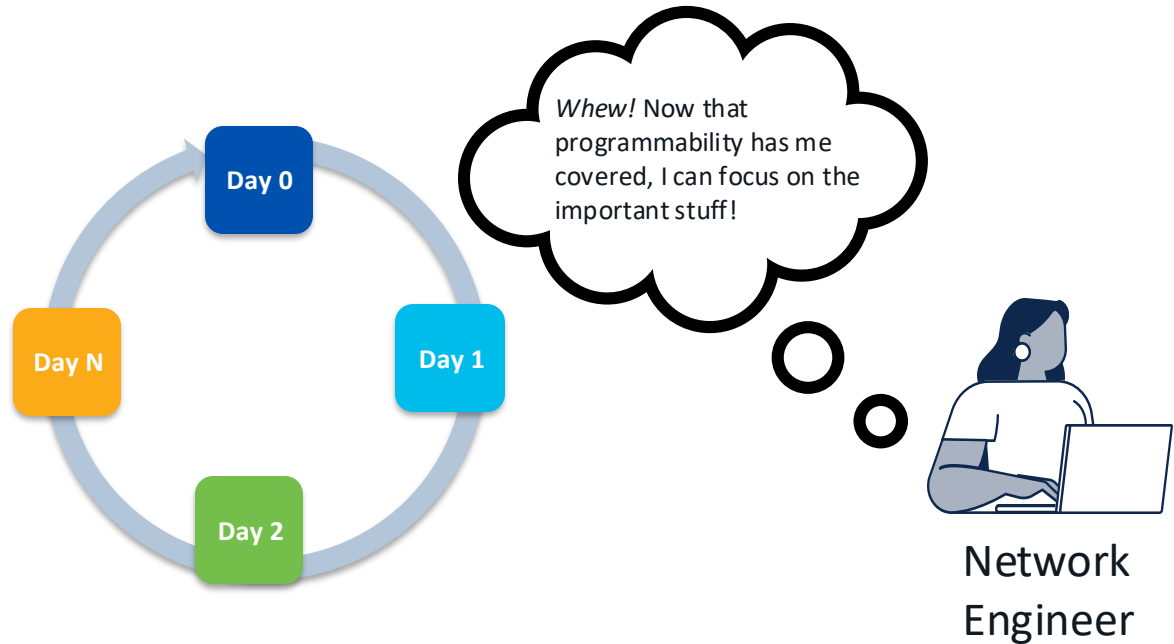
Cisco IOS XE Programmability & Automation Lifecycle



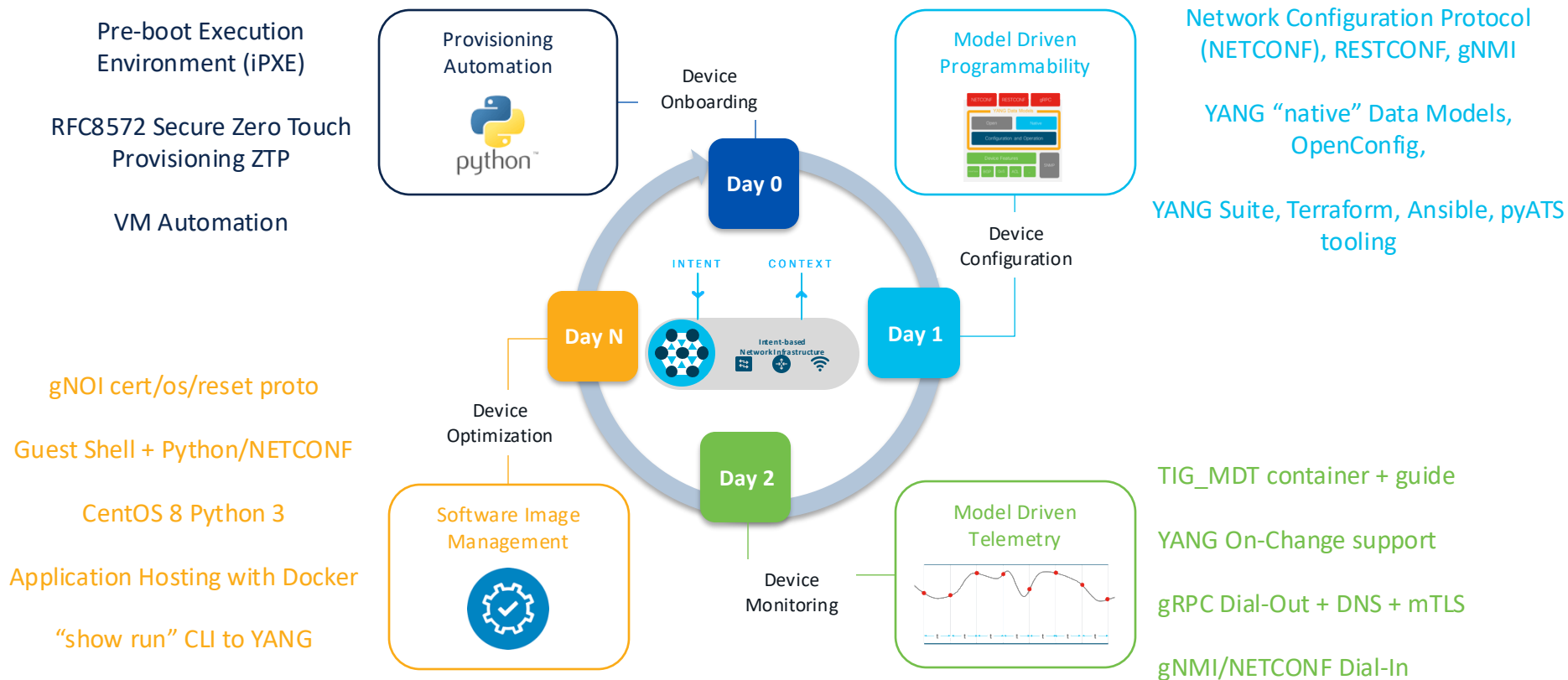
Cisco IOS XE Programmability & Automation Lifecycle



Cisco IOS XE Programmability & Automation Lifecycle



Cisco IOS XE Programmability & Automation Lifecycle



IOS XE programmability and Telemetry “Stack”

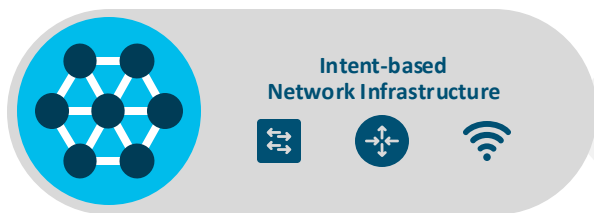
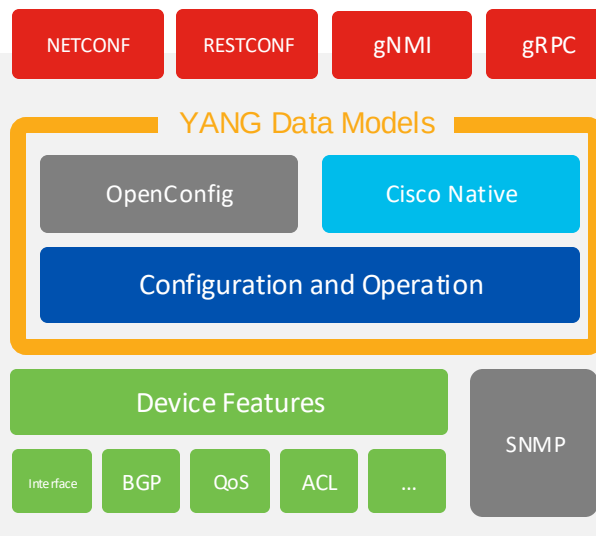
CLI

SNMP

WebUI

The NETCONF, RESTCONF, gNMI and gRPC are programmatic interfaces that provide **additional** methods for interfacing with the IOS XE device – Just like the CLI, SNMP, and WebUI is used for configuration changes and operational metrics so can the programmatic interfaces of NETCONF, RESTCONF, gNMI, and gRPC.

YANG data models define the data that is available for configuration and streaming telemetry



Model Driven Programmability Interface Comparison

Network architecture, security posture and policy, YANG data modules, tools and language preferences are some considerations when leveraging the various MDP interfaces

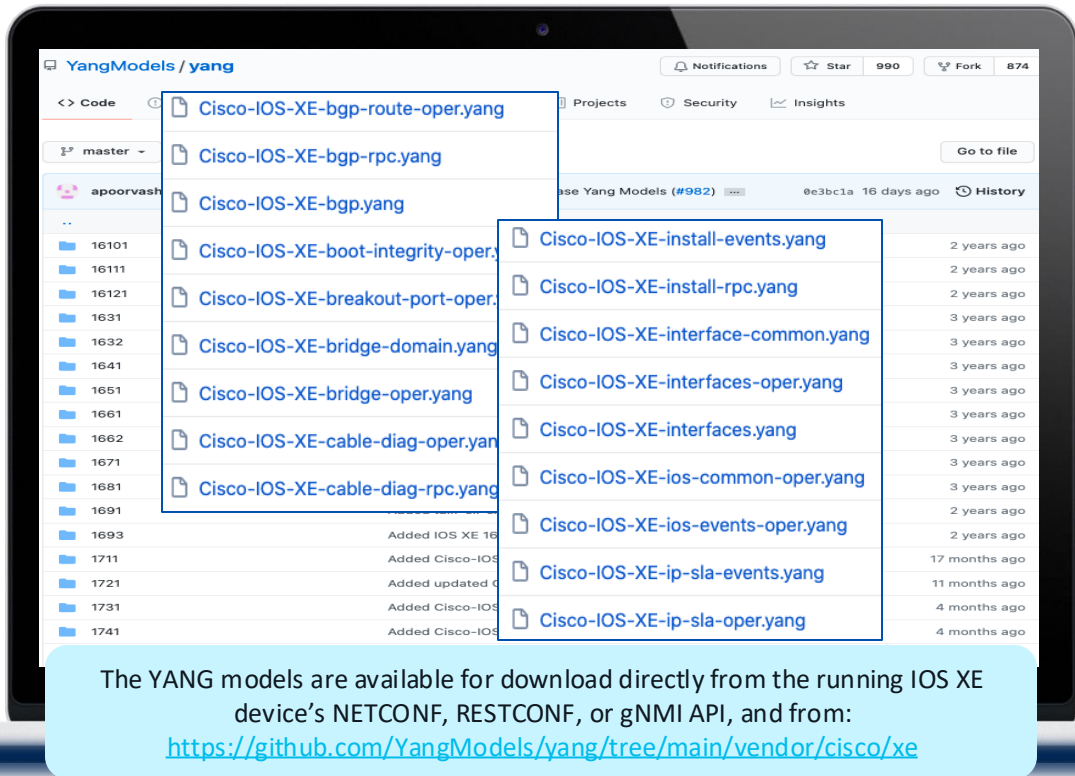
	NETCONF	RESTCONF	gNMI
Minimum IOS XE Version	16.6	16.7	16.8
Recommended Version	17.6	17.6	17.7
Default Port	830	443	9339
Operations	<get>,<get-config>,<edit-config>,<establish-subscription>	GET, POST, PUT, PATCH, DELETE	GET, SET, SUBSCRIBE
Encoding	XML	XML or JSON	RFC7951 JSON_IETF
Security	SSH + PKI certificate or password	HTTPS user/pass	TLS certificate with user authentication
Transport Protocol	SSH	HTTPS	HTTP/2
Tooling	YANG Suite, ncclient, Netconf-console	YANG Suite*, Postman, python	YANG Suite*, gnmic, gnmi_cli
Content	YANG	YANG	YANG + Protobuf



Cisco IOS XE - YANG model API documentation

- There are 10 types of data models including config, oper, actions, and deviations
- RFC 7950 YANG data modelling language are the API definitions for IOS XE
- The YANG modules are available for download from the API and are also published on Github.com
- Notable modules are listed below for the running-config, feature oper, actions and event notifications

TYPE	YANG module name.yang	Description
1 Native configuration model	Cisco-IOS-XE-native	running-config
2 Operation	Cisco-IOS-XE-(feature)-oper	Feature operational data
3 Configuration	Cisco-IOS-XE-(feature)-cfg	Feature configuration, independent of "native" model
4 Events	Cisco-IOS-XE-(feature)-events	Telemetry Events that can be triggered
5 RPC	Cisco-IOS-XE-(feature)-rpc	Actions that can be performed
6 Deviation	Cisco-IOS-XE-(feature)-deviation	Device Implementation deviation from module
7 Types	Cisco-IOS-XE-(feature)-types	Types - Imported by other modules
8 Obsolete	Cisco-IOS-XE-(feature)-obsolete	Obsolete should not be implemented
9 Common	Cisco-IOS-XE-(feature)-common	Common - Imported by other modules
10 Abstractions	OpenConfig-(feature) & Cisco-evpn-service	abstraction for EVPN, OpenConfig config & oper



The YANG models are available for download directly from the running IOS XE device's NETCONF, RESTCONF, or gNMI API, and from:

<https://github.com/YangModels/yang/tree/main/vendor/cisco/x>

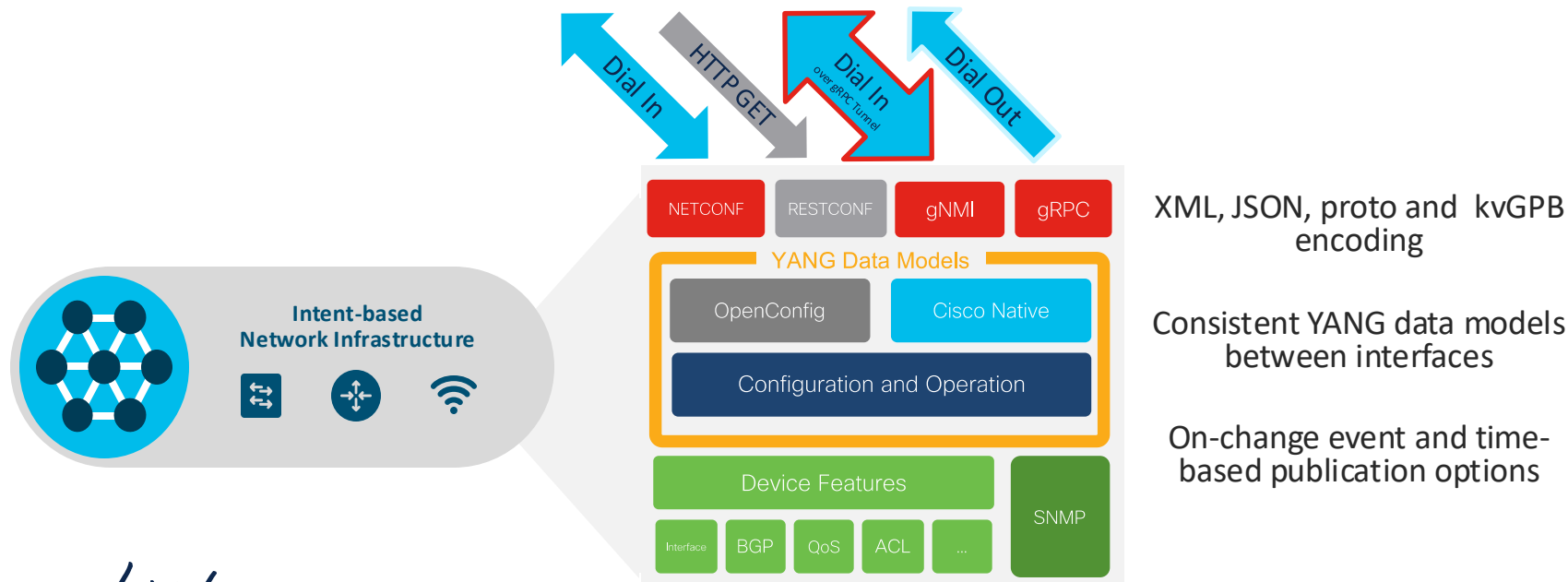
Day 2: Device Monitoring

Model Driven Telemetry Interfaces

↔ Dial In: Collector establishes a connection to the device then subscribes to telemetry (pub/sub)

↔ Dial Out: Telemetry is pushed from the device to the collector based off configuration (push)

Publication / Subscription

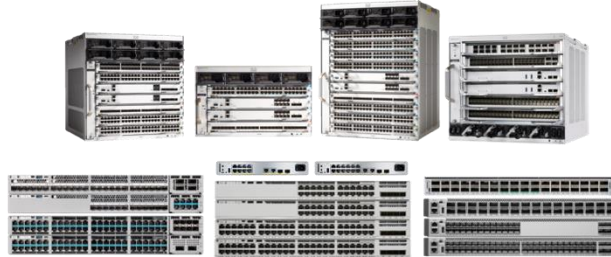


How do we get telemetry data
from our Catalyst
devices?



Telemetry Data Flow

Cisco IOS XE Devices



Collector/Receiver
Decodes to text



Storage
Time Series Database



Monitoring
and Visualisations



https://hub.docker.com/r/jeremycohoe/tig_mdt

<https://github.com/jeremycohoe/cisco-ios-xe-mdt>

https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/179/b_179_programmability_cg/m_179_prog_iETF_telemetry.html

cisco *Live!*



Cisco Telemetry Data Broker (Telegraf)

Cisco Telemetry Broker provides many benefits include brokering, filtering, and transforming data. It provides the ability to replicate telemetry data.



PRODUCTS	UNIT LIST PRICE	STATUS	QUANTITY	ACTION
Virtual License				
Cisco Telemetry Broker Essential License - 100GB/Day TB-ESS-100GB	\$10.00 Per 100GB/Day/12 Month	Added	20 100GB	Delete
Support - Virtual License				
Enhanced Support for Cisco Telemetry Broker SVS-TB-SUP-E	202.50 Per Each/Month	Added	1 Each	Delete
Premium Support for Cisco Telemetry Broker SVS-TB-SUP-P	Take an action to see the List Price Per Each/Month	Not Added	1 Each	22 Swap

► Brokering Data:

The ability to route and replicate telemetry data from a source location to multiple destination consumers.

Quickly onboard new telemetry-based tools!

► Filtering Data:

The ability to filter data that is being replicated to consumers for fine grain control over what consumers are able to see and analyze.

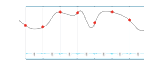
Save money sending data to expensive tools!

► Transforming Data:

The ability to transform data protocols from the exporter to the consumer's protocol of choice.

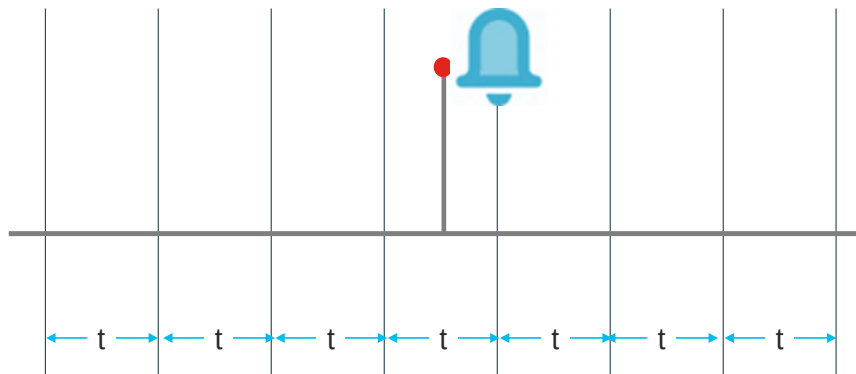
Enable tools to consume multiple data formats!

<https://cs.co/telemetrybroker> aka <https://www.cisco.com/c/en/us/products/security/telemetry-broker/index.html>
<https://blogs.cisco.com/security/taking-full-control-of-your-telemetry-with-the-intelligent-telemetry-plane>



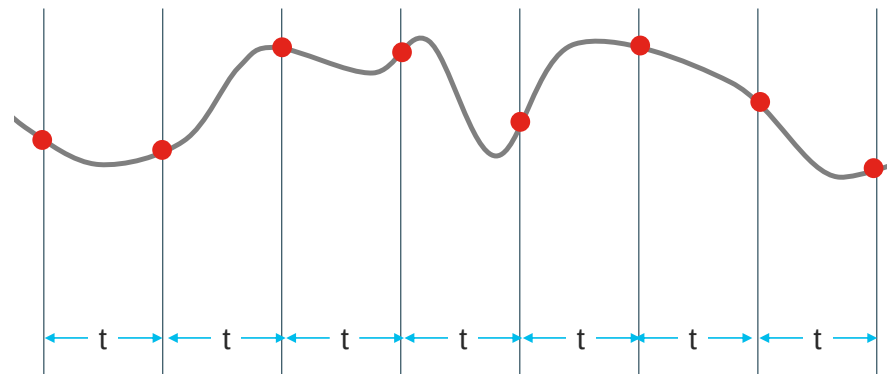
Publication options

On-Change



Feature Model "On-Change" Notifications
Event Notifications (failed login, optic fault, etc)
State and Configuration

Periodic



Feature Model "Periodic" Notifications
Time based publication
Minimum interval 100 centiseconds (1s)

OpenConfig YANG with GNXI, not NETCONF

Monitoring today for each network device

CLI

```
cat9300x-pod29a#show processes cpu
```

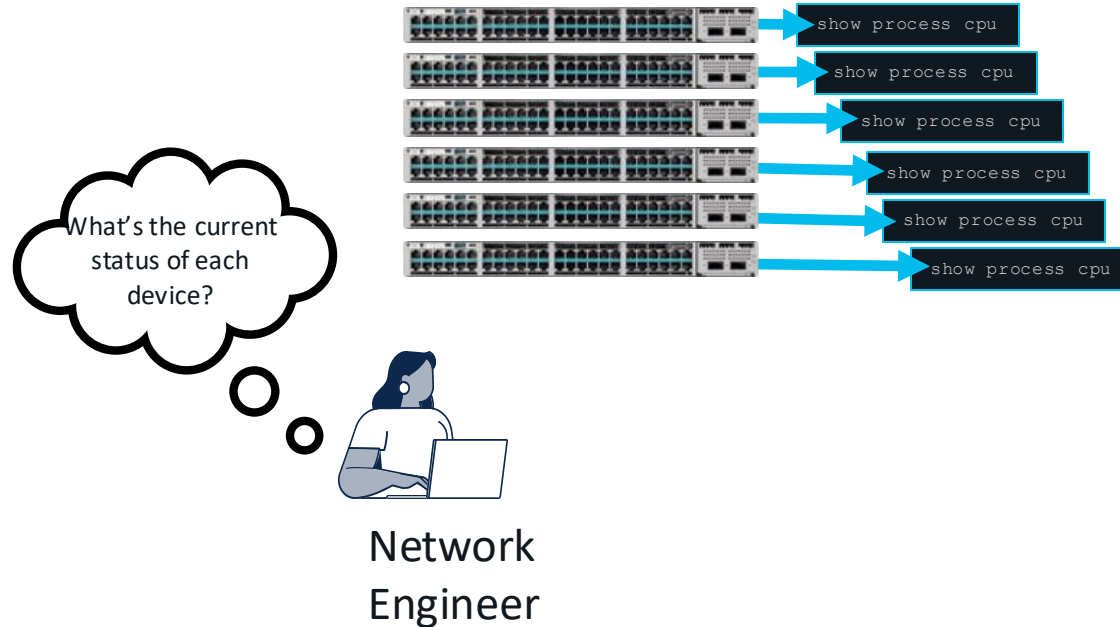
```
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
```

PID	Runtime(ms)	Invoked	uSecs	5Sec	1Min	5Min	TTY	Process
1	0	19	0	0.00%	0.00%	0.00%	0	Chunk Manager
2	429509	2209233	194	0.00%	0.00%	0.00%	0	Load Meter
3	0	2	0	0.00%	0.00%	0.00%	0	PKI Trustpool
4	378	17	22235	0.00%	0.00%	0.00%	0	RF Slave Main Th
5	0	1	0	0.00%	0.00%	0.00%	0	Retransmission o
6	0	1	0	0.00%	0.00%	0.00%	0	IPC ISSU Dispatc
7	0	1	0	0.00%	0.00%	0.00%	0	RO Notify Timers
8	1920	368205	5	0.00%	0.00%	0.00%	0	VIDB BACKGD MGR
9	4460802	1502068	2969	0.00%	0.03%	0.00%	0	Check heaps
10	27588	184104	149	0.00%	0.00%	0.00%	0	Pool Manager
11	0	1	0	0.00%	0.00%	0.00%	0	DiscardQ Backgro
12	0	2	0	0.00%	0.00%	0.00%	0	Timers
13	11	1623	6	0.00%	0.00%	0.00%	0	WATCH_AFS
14	6955	472437	14	0.00%	0.00%	0.00%	0	DB Lock Manager
15	69695	11046159	6	0.00%	0.00%	0.00%	0	GraphIt
16	0	3	0	0.00%	0.00%	0.00%	0	DB Notification

x6 devices

Using Telemetry to Track Devices

Show CPU on each device



Now, let's work more efficiently

Cisco devices use structured data called YANG models

CLI

```
cat9300x-pod29a#show processes cpu
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
PID Runtime(ms)   Invoked  uSecs  5Sec  1Min  5Min  TTY Process
1      0             19      0      0.00% 0.00% 0.00% 0 Chunk Manager
2    429509      2209233  194    0.00% 0.00% 0.00% 0 Load Meter
3      0             2       0      0.00% 0.00% 0.00% 0 PKI Trustpool
4     378        17      22235  0.00% 0.00% 0.00% 0 RF Slave Main Th
5      0             1       0      0.00% 0.00% 0.00% 0 Retransmission o
6      0             1       0      0.00% 0.00% 0.00% 0 IPC ISSU Dispatc
7      0             1       0      0.00% 0.00% 0.00% 0 RO Notify Timers
8     1920      368205   5      0.00% 0.00% 0.00% 0 VIOB BACKGD MGR
9   4460802     1502068 2969    0.00% 0.03% 0.00% 0 Check heaps
10    27588     184104  149    0.00% 0.00% 0.00% 0 Pool Manager
11     0             1       0      0.00% 0.00% 0.00% 0 DiscardQ Backgro
12     0             2       0      0.00% 0.00% 0.00% 0 Timers
13     11        1623   6      0.00% 0.00% 0.00% 0 WATCH_AFS
14    6955     472437  14      0.00% 0.00% 0.00% 0 DB Lock Manager
15   69695     11046159 6      0.00% 0.00% 0.00% 0 GraphIt
16     0             3       0      0.00% 0.00% 0.00% 0 DB Notification
```

CPU data

YANG Model

The screenshot shows the YANG Suite interface with the following details:

- Header:** YANG SUITE, YANG Suite / Exploring YANG / YANG set "IOS XE Switching" / Modules
- Section:** Explore YANG Models
- Filters:** Select a YANG set: IOS XE Switching; Select YANG module(s): Cisco-IOS-XE-process-cpu-oper
- Actions:** Icon legend, Search XPath(s), Search nodes, Expand all nodes
- Tree View:** Cisco-IOS-XE-process-cpu-oper
 - cpu-usage
 - cpu-utilization
 - five-seconds
 - five-seconds-intr
 - one-minute
 - five-minutes
 - cpu-usage-processes

Node Properties	
Name	cpu-utilization
Node type	container
Description	Data nodes for Total CPU Utilization Statistics.
Module	Cisco-IOS-XE-process-cpu-oper
Revision	2017-02-07
Xpath	/cpu-usage/cpu-utilization
Prefix	process-cpu-ios-xe-oper
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-process-cpu-oper
Access	read-only
Operations	• "get"
Schema Node Id	/cpu-usage/cpu-utilization

Using Telemetry to Track Devices

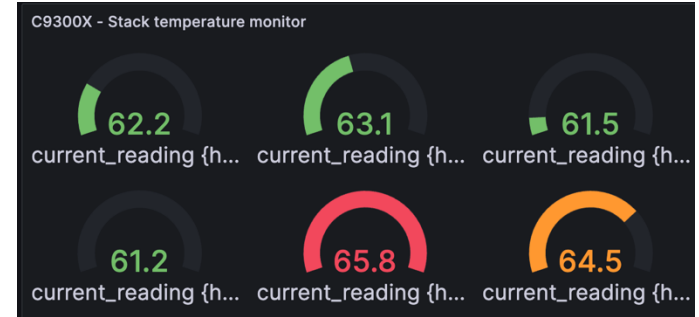
Configure Telemetry Subscription One Time



Monitor devices through a single pane of glass



We need to track the status of all these devices at scale



Telemetry from a Cisco device allows us to know its current status

We can collect this data at scale!

Cisco YANG Suite



YANG API Testing and Validation Environment

Construct and test YANG based APIs over NETCONF,
RESTCONF, gRPC and gNMI

IOS XE / IOS XR / NX OS platforms

Get hands-on using the new learning lab!

<https://developer.cisco.com/learning/labs/intro-yangsuite/>

cisco *Live!*

YANG Suite / Explore YANG Models

Select a YANG set: C9300 | Select YANG module(s): Cisco-IOS-XE-interfaces-oper

Icon legend | Search XPath | Search nodes | Expand all nodes

Display schema nodes only | Display all nodes

Node Properties

Name	Value
statistics	statistics
container	container
Description	A collection of interface-related statistics objects
Module	Cisco-IOS-XE-interfaces-oper
Revision	2020-07-01
Xpath	/interfaces/interface/statistics
Prefix	interfaces-ios-xe-oper
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-interfaces-oper

YANG Suite / NETCONF

YANG Set: C9300 | Module(s): Cisco-IOS-XE-interfaces-oper

NETCONF Operation: get | Device: JCOHOE-DMZ-C9300

Build RPC | Replays | RPC Options... | Run RPC(s) | Clear RPC(s)

Nodes

- Cisco-IOS-XE-interfaces-oper
 - interfaces
 - interface
 - name
 - interface-type
 - admin-status
 - oper-status
 - last-change
 - if-index
 - phys-address
 - higher-layer-if
 - lower-layer-if
 - speed
 - statistics

Value

string

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
    <filter>
      <interfaces xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-interfaces-oper">
        <interface>
          <name/>
          <statistics/>
        </interface>
      </filter>
    </get>
  </rpc>
```



developer.cisco.com/yangsuite

github.com/CiscoDevNet/yangsuite

YANG Suite Capabilities

- Initial Release:
 - Plugin and YANG File Manager, Datasets and diffs
 - Device Manager
 - NETCONF (Python), gRPC Telemetry
 - Docker install support with HTTPS
- Second Release:
 - RESTCONF
 - gNMI
 - Python Integrations
- Third Release:
 - gRPC Telemetry with TLS Support
 - SNMP OID to YANG Xpath Mapping
 - Ansible Integrations
 - Pip install support

Core plugins

Additional plugins

Cisco YANG Suite

Admin

- Manage users
- Manage plugins
- View logs

Setup

- YANG files and repositories
- YANG module sets
- Device profiles

Analytics

- Datasets and diffs
- SNMP to YANG Mapping

Protocols

- gNMI
- gRPC telemetry
- NETCONF
- RESTCONF

Explore

- YANG

Model Metadata with YANG Suite

The CLI config to enable MDT requires the “filter xpath” which defines which YANG data to publish

Use YANG Suite to find the prefix and xpath from the YANG models to use in the configuration

Configuration Guide:

https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/1610/b_1610_programmability_cg/model_driven_telemetry.html

filter xpath / PREFIX : XPATH

The screenshot shows the YANG Suite interface with the following details:

- Top navigation: YANG Suite / Exploring YANG / YANG set "IOS XE Switching" / Modules
- Search bar: Explore YANG Models
- Filters: Select a YANG set (IOS XE Switching), Select YANG module(s) (Cisco-IOS-XE-process-cpu-oper), Load module(s)
- Actions: Icon legend, Search XPath(s), Search nodes, Expand all nodes
- Display options: Display schema nodes only (selected), Display all nodes
- Tree view: Cisco-IOS-XE-process-cpu-oper > cpu-usage > **cpu-utilization** (selected)
- Node Properties table:

Node Properties	
Name	cpu-utilization
Nodetype	container
Description	Data nodes for Total CPU Utilization Statistics.
Module	Cisco-IOS-XE-process-cpu-oper
Revision	2017-02-07
Xpath	/cpu-usage/cpu-utilization
Prefix	process-cpu-ios-xe-oper
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-process-cpu-oper
Access	read-only
Operations	• "get"
Schema Node Id	/cpu-usage/cpu-utilization

filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization

Configure a gRPC Telemetry Subscription

Configuring telemetry subscriptions like the following to collect CPU data over time

On Cisco IOS XE Device:

```
configure terminal
telemetry ietf subscription 1
  encoding encode-kvgpb
  filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilisation
  stream yang-push
  update-policy periodic 60000
  receiver ip address 10.1.1.3 57500 protocol grpc-tcp
```

See more examples at: <https://github.com/jeremycohoe/cisco-ios-xe-mdt>

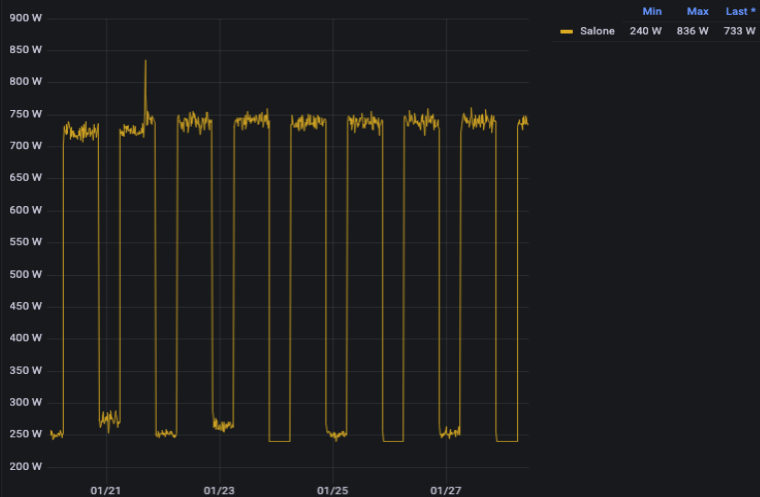
How does this relate to
our sustainability
efforts?



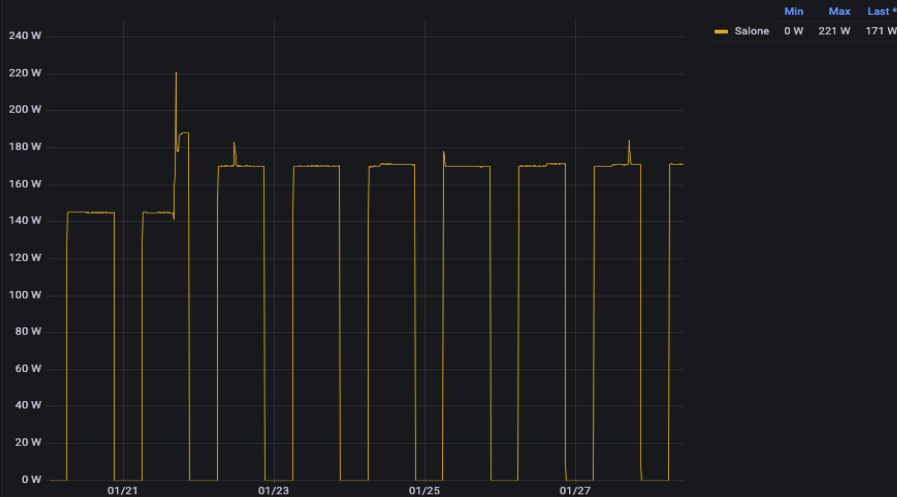


Interface All switch SB-Salone1-C9324H

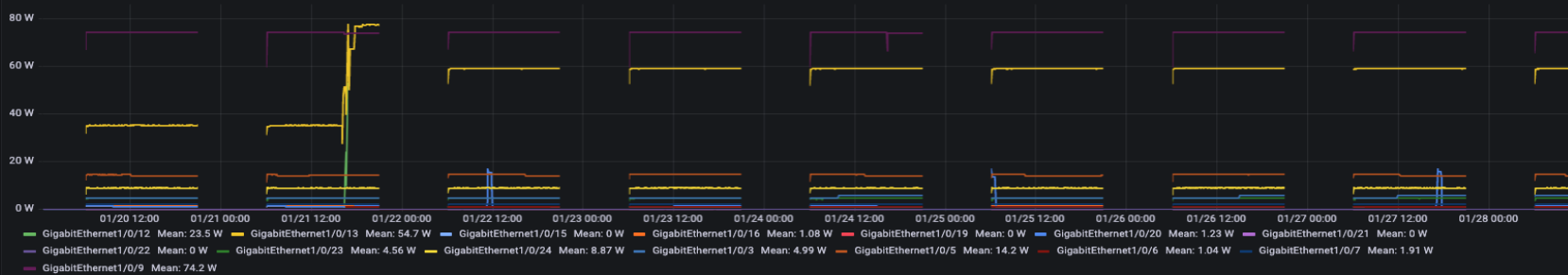
Total Power Allocated - POE + System



Realtime POE Power Consumption



Per interface power consumption



CISCO *Live!*

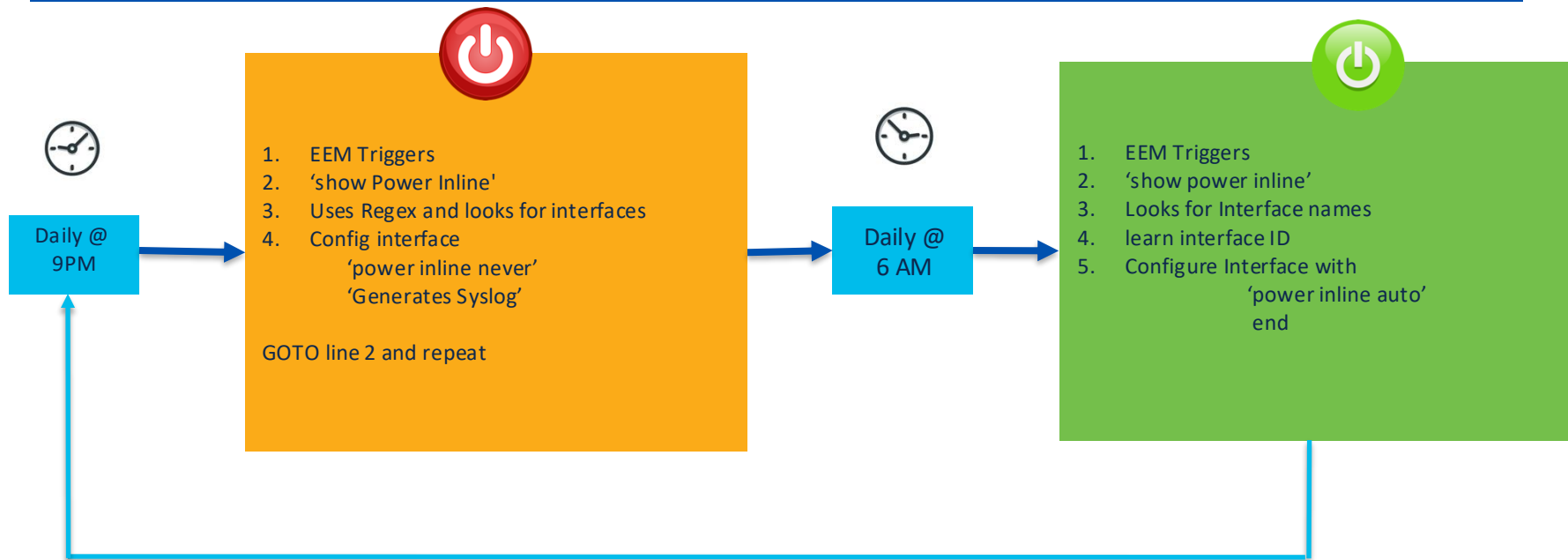
```
no telemetry ietf subscription 69003
telemetry ietf subscription 69003
filter xpath /poe-oper-data/poe-stack
receiver ip address 10.85.134.66 57508 protocol grpc-tcp
source-address 10.85.134.70
source-vrf Mgmt-vrf
stream yang-push
update-policy periodic 3000
encoding enc-de-kva pb
```



BR KAPP-2245

Embedded Event Manager – Applet Flow

Cisco IOS XE Embedded Event Manager (EEM) is a powerful and flexible subsystem
It provides real-time network event detection and onboard automation.
It provides the ability to adapt the behavior of the network devices to better align with business needs.



POE power management with EEM

EEM is used to toggle the power inline auto / never at 9 PM and 6 AM

Power on/off POE ports on a once daily time schedule

Trigger manually by setting event to none and sending CLI:

C9300-SB# event manager run SelectivePowerOn

C9300-SB# event manager run SelectivePowerOff

```
! EEM POE example SelectivePowerOff
no event manager applet SelectivePowerOff
event manager applet SelectivePowerOff
! Turn *OFF* POE power to the ports daily at 9PM: 0 21 * * *
event timer cron name SelectivePowerOff cron-entry "0 21 * * *"
! or
! event none
!
action 0.0 cli command "enable"
action 0.1 cli command "show power inline"
action 0.2 foreach line "$_cli_result" "\n"
action 1.1 regexp "^[^[:space:]]*[^[:space:]]*[^[:space:]]*[^[:space:]]*on.*$" "$line" temp interface
action 1.2 if $_regexp_result eq 1
action 1.3 cli command "conf t"
action 1.4 cli command "interface $interface"
action 1.5 cli command "power inline never"
action 1.6 syslog msg "Turned off PoE on $interface"
action 1.7 end
action 2.1 end
```

Examples @ <https://github.com/jeremycoho/cisco-catalyst-eem-examples>

NETCONF/RSTCONF/gNMI YANG API: <https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/17101/Cisco-IOS-XE-eem.yang>

Example source: <https://elennmatthys.wordpress.com/2014/08/24/intermediary-eem-scripting-more-fun-with-power-over-ether-net/>

EEM <https://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-xe-16/216091-best-practices-and-useful-scripts-for-ee.html>

<https://www.cisco.com/c/en/us/td/docs/ios-xm/ios/eem/command/eem-cr-book/eem-cr-e2.html>

Set power on daily

```
! EEM POE example SelectivePowerOn
no event manager applet SelectivePowerOn
event manager applet SelectivePowerOn
! Turn **ON** POE power to the ports daily at 6AM: 0 6 * * *
event timer cron name SelectivePowerOn cron-entry "0 6 * * *"
!
! or
! event none
!
action 0.0 cli command "enable"
action 0.1 cli command "show power inline"
action 0.2 foreach line "$_cli_result" "\n"
action 1.1 regexp "^[^[:space:]]*[^[:space:]]*off.*$" "$line" temp interface
action 1.2 if $_regexp_result eq 1
action 1.3 cli command "conf t"
action 1.4 cli command "interface $interface"
action 1.5 cli command "power inline auto"
action 1.6 syslog msg "Turned on PoE on $interface"
action 1.7 end
action 2.1 end
```

Benefits

Up to **25%**  energy usage **reduction** on POE switchports with IOT endpoints turned off at TOD (Time of a Day)

How is this applicable to Cisco controller-based solutions for Wireless Access Points?

Catalyst 9800 WLC Calendar Template scheduling

To enable **power save mode** on Cisco Catalyst Access Points

<https://github.com/jeremycohoe/c9800-ap-power-save/>

The screenshot shows the Cisco Catalyst 9800-40 Wireless Controller configuration interface. The left sidebar contains navigation links: Dashboard, Monitoring, Configuration, Administration, Licensing, and Troubleshooting. The main content area is titled 'Edit AP Join Profile'. It features a 'Calendar Profile - Power Profile Map' section with a table showing 'Off Work' with a daily recurrence from 00:00:00 to 23:30:00. Below this is the 'Power Profile Detailed' section, which includes a table of power-saving parameters for various interfaces.

Sequence	Interface	Interface ID	Parameter	Parameter Value
0	Radio	6 GHz	State	Disabled
1	Radio	Secondary 5 GHz	State	Disabled
2	Radio	2.4 GHz	State	Disabled
3	USB	USB 0	State	Disabled
4	Radio	5 GHz	State	Disabled

```
wireless profile power "Off Work Hours"  
0 radio 6ghz state shutdown  
1 radio secondary-5ghz state shutdown  
2 usb 0 state disable  
3 radio 5ghz state shutdown
```

```
wireless profile calendar-profile name "Off Work 5PM to  
Midnight"  
day monday  
day tuesday  
day wednesday  
day thursday  
day friday  
recurrence weekly  
start 17:00:00 end 23:59:59
```

```
wireless profile calendar-profile name "Off Work  
Midnight to 8AM"  
day monday  
day tuesday  
day wednesday  
day thursday  
day friday  
recurrence weekly  
start 00:00:00 end 08:00:00
```

```
ap profile default-ap-profile  
calendar-profile "Workday 5pm to Midnight"  
action power-saving-mode power-profile "Off Work  
Hours"  
calendar-profile "Workday Midnight to 8am"  
action power-saving-mode power-profile "Off Work  
Hours"
```



Monitoring & Automation Impact

Up to **25%**  of Energy usage **reduction** on POE switchports with IOT endpoints turned off ~ 9hrs/Day

Minimum of **12%**  Energy usage **reduction** for controller-based Cisco Catalyst Access Points

100 %  Visibility on POE Power Consumption

*based on use case and data presented in demos

Enhanced Energy Metering

What it is

Enables energy meter for Switch System Power and PoE switchport power consumption
Platforms: C9200, C9300, C9400

What it does

System Energy is power consumed by the system for a specific duration, measured in unit of Watt Second. Macro-metered window size of 3-hours with 12 x 15-minute micro-meters

Gives the visibility within the windows of consumption that can be further analysed and considered for carbon intensity reporting, density-based usage analysis and so on.

What it receives

Ledger on Energy consumption for System Power and PoE Port with a bucketised data of 15 minutes each.

Meter start time 2024-10-22 10:36:34 PST
Energy Data For Last 180 Minute

System Energy System Meter Update			
Mod	Model No	(MilliWattSec)	Time
1	C9300-24H	234894600	2024-10-22 11:32:39 PST

PoE Port Meter

SB-Salone1-C9324H-sho power inline meter
Module Available Used Remaining
(Watts) (Watts) (Watts)

1 860.0 120.0 740.0

Interface	Meter Update Time	Hourly Metered Value (MilliWattSec)	Metered Energy in MilliWattSec (15min Buckets)
Gi1/0/1	2024-10-22 11:21:09 PST	0	0-0-0-0-0-0-0-0-0-0-0-0-0
Gi1/0/2	2024-10-22 11:21:09 PST	10507380	1465368-3553224-3558320-1930468-0-0-0-0-0-0-0-0

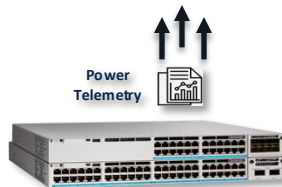
15 minutes/bucket #

0	15 min
1	30 min
2	45 min
3	60 min
4	75 min
5	90 min
6	105 min
7	120 min
8	135 min
9	150 min
10	165 min
11	180 min

Hour-1

Hour-2

Hour-3



What's changed with power in the 17.15 release?

Previously

- Instantaneous power only

Today

- Instantaneous power
- Power over time
 - Smart Buildings use case:
 - 48 PoE ports create too much data, now we can use buckets & can collect data at less frequent intervals

Enhanced Energy Metering

CLI with micro-metering

Energy Metering System Power

```
Meter start time 2024-10-22 10:36:34 PST
Energy Data For Last 180 Minute
```

Mod	Model No	System Energy (MilliWattSec)	System Meter Update Time
1	C9300-24H	234894600	2024-10-22 11:32:39 PST

Energy Metering PoE Port Power

```
SB-Salone1-C9324H#sho power inline meter
```

Module	Available (Watts)	Used (Watts)	Remaining (Watts)
1	860.0	120.0	740.0

Interface	Meter Update Time	Hourly Metered Value (MilliWattSec)	Metered Energy in MilliWattSec (15min Buckets)
Gi1/0/1	2024-10-22 11:21:09 PST	0	0-0-0-0-0-0-0-0-0-0
Gi1/0/2	2024-10-22 11:21:09 PST	10507380	1465368-3553224-3558320-1930468-0-0-0-0-0-0
Gi1/0/3	2024-10-22 11:21:09 PST	0	0-0-0-0-0-0-0-0-0-0
Gi1/0/4	2024-10-22 11:21:09 PST	0	0-0-0-0-0-0-0-0-0-0
Gi1/0/5	2024-10-22 11:21:09 PST	0	0-0-0-0-0-0-0-0-0-0
Gi1/0/6	2024-10-22 11:21:09 PST	0	0-0-0-0-0-0-0-0-0-0

```
SB-Salone1-C9324H#sho ver | i time
SB-Salone1-C9324H uptime is 45 minutes
Uptime for this control processor is 46 minutes
```

Enhanced Energy Metering

Data models updated for Power Telemetry

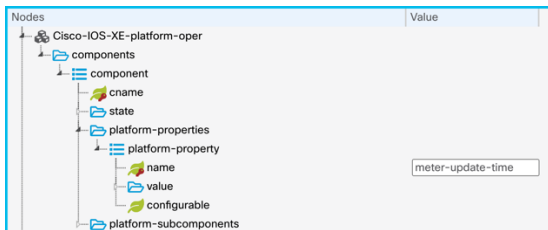
Energy Metering - System Power

Name	cname
Nodetype	leaf
Datatype	string
Description	References component name
Module	Cisco-IOS-XE-platform-oper
Revision	2023-03-01
Xpath	<u>/components/component/cname</u>
Prefix	<u>platform-ios-xe-oper</u>
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-platform-oper
Schema Node Id	/components/component/cname
Key	true
Access	read-only
Operations	<ul style="list-style-type: none">• "get"

Energy Metering – PoE Port Power

Name	power-consumption
Nodetype	leaf
Datatype	decimal64
Description	Actual Power consumed by power device
Module	Cisco-IOS-XE-poe-oper
Revision	2024-03-01
Xpath	<u>/poe-oper-data/poe-port-detail/power-consumption</u>
Prefix	<u>poe-ios-xe-oper</u>
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper
Schema Node Id	/poe-oper-data/poe-port-detail/power-consumption
Units	Watts
Min	-92233720368547758.08
Max	92233720368547758.07
Fraction Digits	2
Access	read-only
Operations	<ul style="list-style-type: none">• "get"

Energy Metering using the API



Show same 2 things as CLI

Get system energy + time

NETCONF GET operation

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
    <filter>
      <components xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-platform-oper">
        <component>
          <platform-properties>
            <platform-property>
              <name>meter-update-time</name>
            </platform-property>
          </platform-properties>
        </component>
      </components>
    </filter>
  </get>
</rpc>
```

Response

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
  message-id="urn:uuid:820e9cb4-fc8d-4f3d-bad9-4b09bab635e2">
  <data>
    <components xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-platform-oper">
      <component>
        <platform-properties>
          <platform-property>
            <name>meter-update-time</name>
            <value>
              <string>1730240928747</string>
            </value>
            <configurable>false</configurable>
          </platform-property>
        </platform-properties>
      </component>
    </components>
  </data>
</rpc-reply>
```

Epoch time

= Wednesday October 30, 2024
Melbourne, Australia

PoE API

YANG Suite / Exploring YANG / YANG set "smartbuilding-salone1-default-yangset" / Modules

admin ?

Explore YANG Models

Select a YANG set

smartbuilding-salone1-default-yangset

Select YANG module(s)

Cisco-IOS-XE-poe-oper x

Load module(s)

Icon legend

Search XPath

Search nodes

Expand all nodes

Display schema nodes only

Display all nodes

Cisco-IOS-XE-poe-oper

poe-oper-data

poe-port

poe-port-detail

intf-name

power-used

pd-class

device-detected

device-name

police

power-admin-max

power-from-pse

power-to-pd

power-consumption

max-power-drawn

oper-state

admin-state

oper-power

admin-police

oper-police

cutoff-power-police

power-negotiation-used

four-pair-poe-supported

four-pair-poe-enabled

four-pair-pd-arch

over-current-counter

short-current-counter

power-denied-counter

Node Properties

Name	poe-port-detail
Nodetype	list
Description	List of PoE interfaces, keyed by interface name
Module	Cisco-IOS-XE-poe-oper
Revision	2024-07-01
Xpath	/poe-oper-data/poe-port-detail
Prefix	poe-ios-xe-oper
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper
Schema Node Id	/poe-oper-data/poe-port-detail
Keys	<ul style="list-style-type: none">"intf-name"
Access	read-only
Operations	<ul style="list-style-type: none">"get"

Reference URL: <https://tools.ietf.org/html/rfc6020#section-7.8>

7.8. The list Statement

The "list" statement is used to define an interior data node in the schema tree. A list node may exist in multiple instances in the data tree. Each such instance is known as a list entry. The "list" statement takes one argument, which is an identifier, followed by a block of substatements that holds detailed list information.

A list entry is uniquely identified by the values of the list's keys, if defined.

PoE API (cont.) NETCONF GET

YANG Set: smartbuilding-salone1-default-yangset | Module(s): Cisco-IOS-XE-poe-oper | Load Module(s)

NETCONF Operation: get | Device: SB-salone2 | Edit Device | Open Device Window | Run RPC(s) | Clear RPC(s)

YANG Tree | Replays | RPC Options... | Build RPC

Nodes

- Cisco-IOS-XE-poe-oper
 - poe-oper-data
 - poe-port
 - poe-port-detail

Value

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
    <filter>
      <poe-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper">
        <poe-port-detail/>
      </poe-oper-data>
    </filter>
  </get>
</rpc>
```

NETCONF GET operation

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
    <filter>
      <poe-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper">
        <poe-port-detail/>
      </poe-oper-data>
    </filter>
  </get>
</rpc>
```

Response

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <poe-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-poe-oper">
      <poe-port-detail>
        <intf-name>GigabitEthernet 1/0/2</intf-name>
        <power-use>30.0</power-use>
        <pd-class>pd-ieee 4</pd-class>
        <device-detected>true</device-detected>
        <device-name>IEEE PD</device-name>
        <police>false</police>
        <power-admin-max>90.0</power-admin-max>
        <power-from-pse>30.0</power-from-pse>
        <power-to-pd>30.0</power-to-pd>
        <power-consumption>3.91</power-consumption>
        <max-power-drawn>5.08</max-power-drawn>
      </poe-port-detail>
    </poe-oper-data>
  </data>
</rpc-reply>
```

Subscription to the new platform data

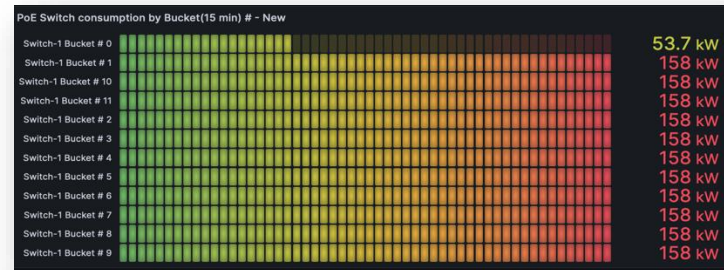
```
telemetry ietf subscription 79007
encoding encode-kvgpb
filter xpath /platform-ios-xe-oper:components/component/platform-properties/platform-property
source-address 172.26.202.111
stream yang-push
update-policy periodic 300000
receiver ip address 172.26.202.69 57000 protocol grpc-tcp
```

```
telemetry ietf subscription 79008
encoding encode-kvgpb
filter xpath /poe-ios-xe-oper:poe-oper-data/poe-port-detail
source-address 172.26.202.111
stream yang-push
update-policy periodic 30000
receiver ip address 172.26.202.69 57000 protocol grpc-tcp
```

```
telemetry ietf subscription 79007
encoding encode-kvgpb
filter xpath /platform-ios-xe-oper:components/component/platform-properties/platform-property
source-address 172.26.202.111
stream yang-push
update-policy periodic 300000
receiver ip address 172.26.202.69 57000 protocol grpc-tcp
```

```
telemetry ietf subscription 79008
encoding encode-kvgpb
filter xpath /poe-oper-data/poe-port-detail
source-address 172.26.202.111
stream yang-push
update-policy periodic 30000
receiver ip address 172.26.202.69 57000 protocol grpc-tcp
```

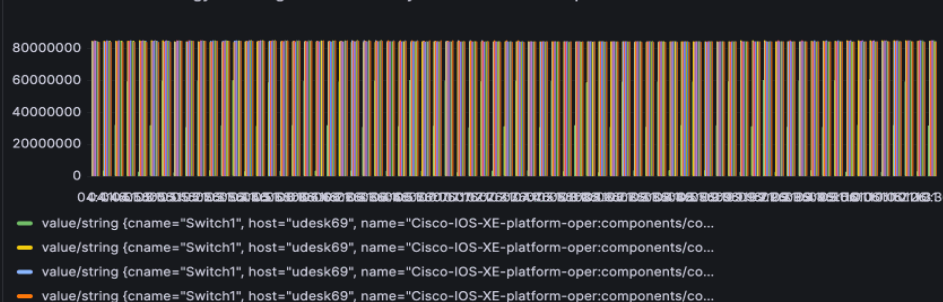
What can we do with Power Telemetry



c9300X- 8xStack - Instantaneous System Power Consumption



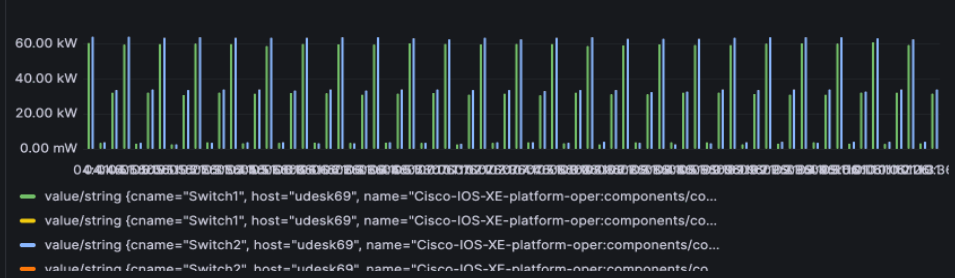
C9300 -Enhanced energy metering- all buckets - System Power Consumption - 1715.1



9300 Stack - Instantaneous System Power Consumption



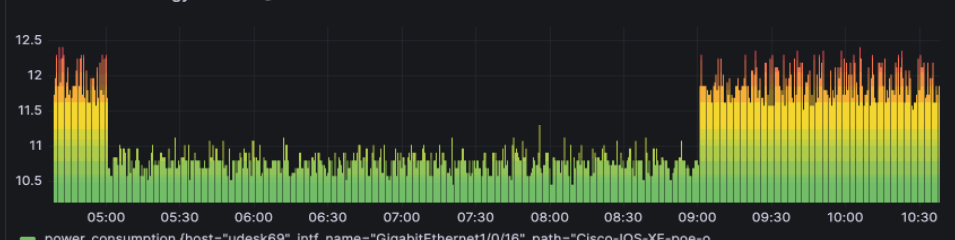
Bucket#0 - Energy Meter@17.15.1



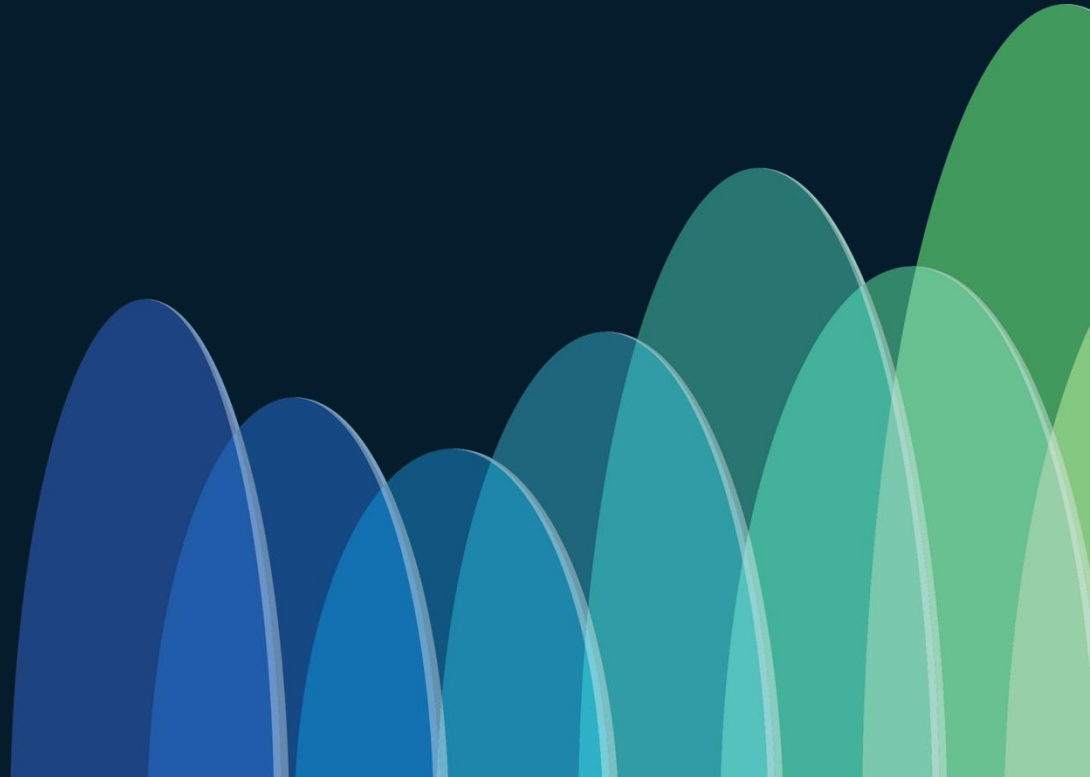
17.14.1 PoE Port Power Allocated vs Consumed



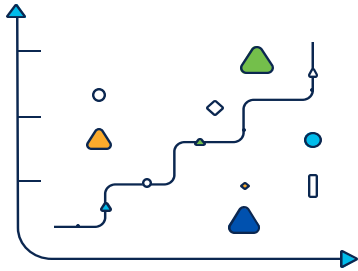
17.15.1 PoE Port Energy meter ⓘ



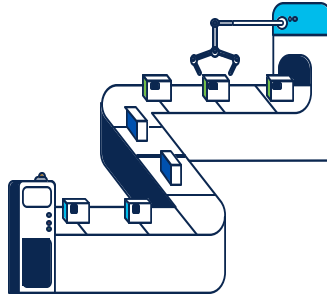
Benefits



Now using programmability for a Sustainability Use Case



↓ Energy Consumption



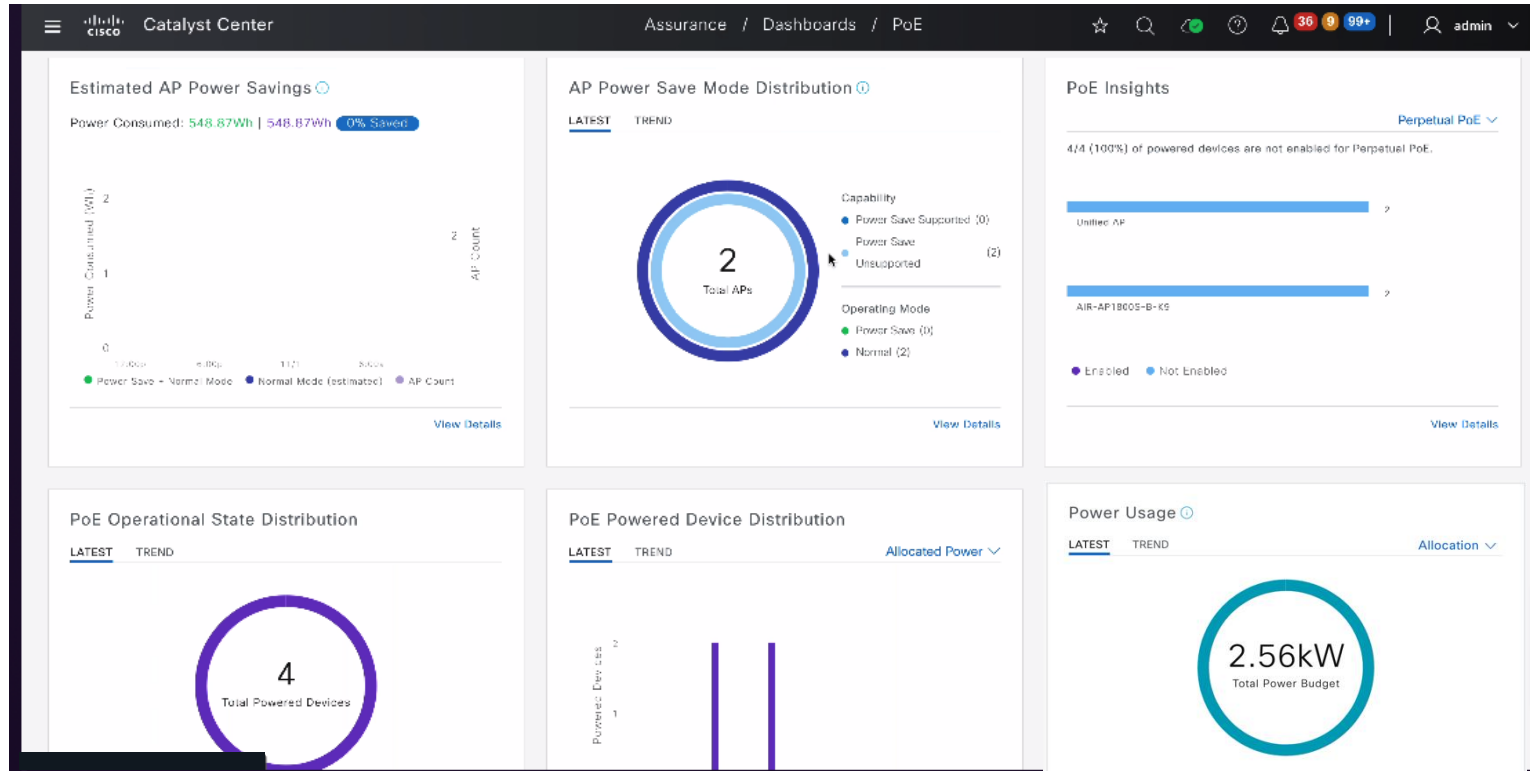
↓ Operational Costs



↑ Sustainability

Sustainability on other Platforms

PoE Dashboard on Catalyst Center



Device Inventory for PoE Devices

 Catalyst Center



142

 Story 

All Devices / LO-BN

 LO-BN  [Run Commands](#) [View 360](#)

Last Updated: 7 minutes ago 

 Reachable  Managed | IP Address: 10.93.141.28 | Device Model: Cisco Catalyst 9300 Switch | Device Role: DISTRIBUTION | Uptime: 195 days 4 hrs 36 mins | Site: Global/France/ParisOlympics/Floor-0

DETAILS

Summary

System >

Interfaces >

Layer 2 Configuration BETA >

Browse Configurations >

User Defined Fields

REP Rings

Wireless Info

SECURITY

Advisories

FIELD NOTICES

Field Notices

Potential Field Notices

Color Code: Status 



 Connected  Error Disabled  Admin Disabled  Not Connected  PoE Enabled

System

Image (Version)	17.9.4a	Fan Tray	3
IP Address	10.93.141.28	Power Supply	2
MAC Address	90:77:ee:ac:ab:80	SFP Modules	0
Platform	C9300-24U	Serial Number	FOC2426U0VX

[View System Details](#)

cisco *Live!*

#CiscoLiveAPJC

BRKAPP-2245

© 2024 Cisco and/or its affiliates. All rights reserved. Cisco Public

73

Power drawn by a device

The screenshot shows the Cisco Catalyst Center interface for a device named LO-BN. The left sidebar contains navigation menus for DETAILS, SECURITY, FIELD NOTICES, and COMPLIANCE. The main content area shows the configuration for GigabitEthernet1/0/17. A table lists system details:

System	
Type	Physical
Access VLAN	WIRELESS (65)
Voice VLAN	NA
MAC Address	90:77:ee:ac:ab:91
Link	FullDuplex
PoE	Enabled
Speed	1000 Mbps
MTU	1500 Bytes
Last Input	Nov 12, 2024 9:12 PM
Last Output	Nov 12, 2024 9:14 PM
Admin Status	Up
Operational Status	Connected

A large blue box highlights the 'Power Drawn' section, which currently shows '3 Watts'.

PoE APIs

Cisco

Catalyst Center

Platform / Developer Toolkit

☆

🔍

🟢

?

🔔 14 2

|

👤 Story

▼

APIs

Integration Flows

Event Notifications

🔍 Search

Know Your Network

▼

Devices

🔍 poe

×

Know Your Network

Know your Network APIs can be used to discover details about clients, sites, topology and devices. It also provides programmatic REST APIs to add devices to the network and export device data.

Devices

Method	Name	Description	URL	Actions
GET	Returns POE interface details for the device.	Returns POE interface details for the device, where deviceuid is mandatory & accepts comma separated interface names which is optional and returns information for that particular interfaces where(operStatus = operationalStatus)	/network-device/\${deviceUid}/interface/poe-detail	⋮ ▼
GET	POE details	Returns POE details for device.	/network-device/\${deviceUid}/poe	⋮ ▼

Return the PoE Interface details for the provided device

Returns POE interface details for the device.



GET [https://10.93.141.45/dna/intent/api/v1/network-device/\\$deviceUuid/interface/poe-detail](https://10.93.141.45/dna/intent/api/v1/network-device/$deviceUuid/interface/poe-detail)

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

[Cisco DevNet API Guide](#)

Parameters Responses Code Preview

Request URL Parameters

Name	Description	DataType
deviceUuid	uuid of the device	string

Request Query Parameters

Name	Description	DataType
interfaceNameList	comma seperated interface names	string

Try 'Returns POE interface details for the device.'

Method: **GET** Public URL: <https://10.93.141.45/dna/intent/api/v1/network-device/db05ad49-2c8d-4094-9df6-3a03f9b28069/interface/poe-detail>

PARAMETERS

PATH PARAMETERS

deviceUuid

QUERY PARAMETERS

☐ Select All

☐ interfaceNameList

Response Headers Status Code: 200

```
1: {
2:   "response": [
3:     {
4:       "interfaceName": "GigabitEthernet1/0/4",
5:       "operStatus": "OFF",
6:       "adminStatus": "AUTO",
7:       "allocatedPower": "8.0",
8:       "maxPortPower": "60.0",
9:       "portPowerDrawn": "0"
10:    },
11:    {
12:       "interfaceName": "GigabitEthernet1/0/22",
13:       "operStatus": "OFF",
14:       "adminStatus": "AUTO",
15:       "allocatedPower": "8.0",
16:       "maxPortPower": "60.0",
17:       "portPowerDrawn": "0"
18:    },
19:    {
20:       "interfaceName": "GigabitEthernet1/0/11",
21:       "operStatus": "OFF",
22:       "adminStatus": "AUTO",
23:       "allocatedPower": "8.0",
24:       "maxPortPower": "60.0",
25:       "portPowerDrawn": "0"
26:    },
27:    {
28:       "interfaceName": "GigabitEthernet1/0/3",
29:       "operStatus": "OFF",
30:       "adminStatus": "AUTO",
31:       "allocatedPower": "8.0",
32:       "maxPortPower": "60.0",
33:       "portPowerDrawn": "0"
34:    },
35:    {
36:       "interfaceName": "GigabitEthernet1/0/23",
37:       "operStatus": "OFF",
38:       "adminStatus": "AUTO",
39:       "allocatedPower": "8.0",
40:       "maxPortPower": "60.0",
41:       "portPowerDrawn": "0"
42:    }
43:   ]
44: }
```

Get PoE details

POE details

×

GET https://10.93.141.45/dna/intent/api/v1/network-device/\${deviceUuid}/poe

Returns POE details for device.

[Cisco DevNet API Guide](#)

Parameters Responses Code Preview

Request URL Parameters

Name	Description
deviceUuid	UUID of the device

Try 'POE details '

×

Method: **GET** Public URL :https://10.93.141.45/dna/intent/api/v1/network-device/db05ad49-2c8d-4094-9df6-3a03f9b28069/poe

PARAMETERS

PATH PARAMETERS

deviceUuid*
db05ad49-2c8d-4094-6

⋮

Response

Headers

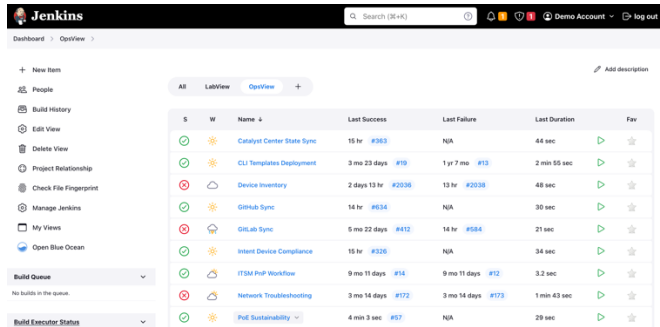
Status Code: 200

```
1 {  
2   "response": {  
3     "powerAllocated": "1195",  
4     "powerConsumed": "98",  
5     "powerRemaining": "1105"  
6   },  
7   "version": "1.0"  
8 }
```

Reset

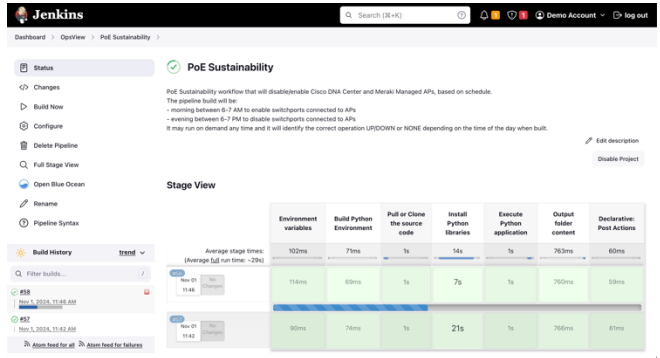
Run

Catalyst Center + Jenkins



The screenshot shows the Jenkins Dashboard. The left sidebar contains navigation links: New Item, People, Build History, Edit View, Delete View, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, and Open Blue Ocean. The main area displays a table of build jobs. The 'PoE Sustainability' job is highlighted.

S	W	Name	Last Success	Last Failure	Last Duration	Fav
✓	🔔	Catalyst Center State Sync	15 hr #363	N/A	44 sec	▶ ☆
✓	🔔	CLI Templates Deployment	3 mo 23 days #19	1 yr 7 mo #13	2 min 55 sec	▶ ☆
✓	☁	Device Inventory	2 days 13 hr #2036	13 hr #2038	48 sec	▶ ☆
✓	🔔	GitLab Sync	14 hr #634	N/A	30 sec	▶ ☆
✗	🔔	GitLab Sync	6 mo 22 days #412	14 hr #584	21 sec	▶ ☆
✓	🔔	Intent Device Compliance	15 hr #326	N/A	34 sec	▶ ☆
✓	☁	ITSM PoP Workflow	9 mo 11 days #14	9 mo 11 days #12	3.2 sec	▶ ☆
✗	☁	Network Troubleshooting	3 mo 14 days #172	3 mo 14 days #173	1 min 43 sec	▶ ☆
✓	🔔	PoE Sustainability	4 min 3 sec #57	N/A	29 sec	▶ ☆



The screenshot shows the Jenkins Pipeline View for the 'PoE Sustainability' pipeline. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Open Blue Ocean, Rename, and Pipeline Syntax. The main area displays the 'Stage View' with a table of stage durations.

Environment variables	Build Python Environment	Pull or Clone the source code	Install Python libraries	Execute Python application	Output build content	Declarative Post Actions
103ms	73ms	1s	14s	1s	763ms	60ms
114ms	69ms	1s	7s	1s	760ms	69ms
90ms	76ms	1s	21s	1s	766ms	61ms

App PoE_Sustainability run:.....

[Pipeline] withEnv

[Pipeline] {

[Pipeline] echo

Verify Working Path:.....

[Pipeline] sh

+ pwd

/var/lib/jenkins/workspace/PoE Sustainability

[Pipeline] dir

Running in /var/lib/jenkins/workspace/PoE Sustainability/dnacentr_jenkins_automations

[Pipeline] {

[Pipeline] sh

+ python3 poe_sustainability.py

INFO:root:2024-11-01 11:46:57 - App "poe_sustainability.py" Start

INFO:root:2024-11-01 11:46:57 - Cisco DNA Center PoE Sustainability

INFO:root:2024-11-01 11:46:57 - Local time hour: 11

INFO:root:2024-11-01 11:46:57 - No PoE operation performed at this time, during business hours

What about Meraki? What can it show me?

Use Case: software updates on cash registers in retail stores: Wake On LAN use case



Did you know?

Powering off 1,000 PoE devices (10 W) for 12 hours per day can lead to 50% energy savings or ~43,800 kWh energy / ~\$7,095 cost (\$0.162 per kWh) / ~16.5 metric tonnes CO2 emissions* savings per year.

<https://meraki.cisco.com/product-collateral/meraki-mis-sustainability-whitepaper/?file>

Meraki Mt40 Smart Power Controller PoE



Did you know?

The Cisco Meraki MT40 smart power controller helps power on/off non-PoE devices as well as measure and report their power consumption after implementing energy-saving measures.

<https://meraki.cisco.com/product-collateral/meraki-ms-sustainability-whitepaper/?file>

Try out MDT in your own network!



Updated TIG_MDT container now available!

```
docker pull jeremycohoe/tig_mdt
docker run -ti -p 3000:3000 -p 57500:57500 jeremycohoe/tig_mdt
```

Cisco IOS XE Devices



Collector/Receiver
Decodes to text

Storage
Time Series Database

**Monitoring
and Visualisations**



Upgrade coming to Telegraf, Influx, and Grafana Model Driven Telemetry (TIG_MDT) Docker container
Making it easier to consume telemetry in production

Upgraded Telegraf, InfluxDB, and Grafana tools
Additional dashboards for

Device Health, Wireless Client, Wireless AP, RFetc
Examples for device CLI configuration for telemetry
Details of scale and data storage requirements



https://hub.docker.com/r/jeremycohoe/tig_mdt

https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/179/b_179_programmability_cg/m_179_prog_ielf_telemetry.html

<https://github.com/jeremycohoe/cisco-ios-xe-mdt>

cisco *Live!*



Terraform for Model Driven Telemetry

Enable gRPC Dial-Out telemetry subscriptions for the POE and Basic Device Monitoring use cases

```
auto@pod22-xelab:~$ terraform plan -var-file="var.tfvars"
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# iosxe_mdt_subscription.cpu_subs["100"] will be created
+ resource "iosxe_mdt_subscription" "cpu_subs" {
  encoding      = "encode-kvgpb"
  filter_xpath  = "/process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds"
  id            = (known after apply)
  receivers     = [
    {
      address = "10.1.1.3"
      port    = 57500
      protocol = "grpc-tcp"
    },
  ],
  stream        = "yang-push"
  subscription_id = 100
  update_policy_periodic = 3000
}

# iosxe_mdt_subscription.cpu_subs["101"] will be created
+ resource "iosxe_mdt_subscription" "cpu_subs" {
  encoding      = "encode-kvgpb"
  filter_xpath  = "/process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/one-minute"
  id            = (known after apply)
  receivers     = [
    {
      address = "10.1.1.3"
      port    = 57500
      protocol = "grpc-tcp"
    },
  ],
  stream        = "yang-push"
  subscription_id = 101
  update_policy_periodic = 3000
}
```

Lab Guide Steps:

1. Introduction
2. IOS XE CLI pre-req
3. Terraform Install & headers, device variables
4. Variables and config for MDT subscription
5. Terraform configuration for device monitoring
6. TF Workflow: init, plan, apply & destroy
7. Validation with TF and CLI
8. Conclusion

iosxe_mdt_subscription (Resource)

This resource can manage the MDT Subscription configuration.

Example Usage

```
resource "iosxe_mdt_subscription" "example" {
  subscription_id = 101
  stream          = "yang-notif-native"
  encoding        = "encode-kvgpb"
  source_vrf      = "Mgmt-vrf"
  source_address  = "1.2.3.4"
  update_policy_on_change = true
  filter_xpath    = "/ios-events-ios-xe-oper"
  receivers = [
    {
      address = "5.6.7.8"
      port    = 57600
      protocol = "grpc-tcp"
    }
  ]
}
```

https://registry.terraform.io/providers/CiscoDevNet/iosxe/latest/docs/resources/mdt_subscription

https://registry.terraform.io/providers/CiscoDevNet/iosxe/latest/docs/data-sources/mdt_subscription

<https://github.com/jeremycohoe/cisco-ios-xe-panda-lab-terraform>

DEMO – install docker container and use Terraform files to configure telemetry subscriptions

```
auto@pod27-xelab: ~ (ssh)
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
auto@pod27-xelab:~$
auto@pod27-xelab:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
auto@pod27-xelab:~$
auto@pod27-xelab:~$

auto@pod27-xelab: ~ (ssh)
c9300-pod27# sh run | s tel
telemetry ietf subscription 6041337
encoding encode-kvgpb
filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds
stream yang-push
update-policy periodic 30000
receiver ip address 10.1.1.3 57500 protocol grpc-tcp
c9300-pod27#
c9300-pod27#
Oct 16 21:55:56.082: %HA_EM-6-LOG: catchall: show running-config
```

Github.com and Grafana.com documentation

README.md

Cisco Catalyst POE

Details about monitoring the POE and power consumption and utilization on Cisco Catalyst IOS XE.

This use case relies on the TIG_MDT Docker container which has the Telegraf, InfluxDB, and Grafana toolset available from [The cisco-ios-xe-mdt repository](#) on Github

Prerequisites

You need Cisco Catalyst 9300 POE

Some POE devices connected too

Minimum IOS XE Software release: ____ ?

POE YANG

The YANG module that has details about Power Over Ethernet operational data is the [Cisco-IOS-XE-poe-oper.YANG](#) which has contains a lot of information about the feature.

There are two specific containers that have data that is relevant to POE and power monitoring: **poe-switch** which models port specific details, and the **poe-port-detail** container which models device level power information.

The **poe-switch** container has ...

 poe-switch	Name	poe-switch
 switch-num	Nodetype	list
 power-budget	Description	List of PoE switches, keyed by switch number

Grafana Labs Products Open source Solutions Learn Company

← All dashboards

Cisco Catalyst POE

Cisco Catalyst Model Driven Telemetry (MDT) Power Over Ethernet (POE) dashboard

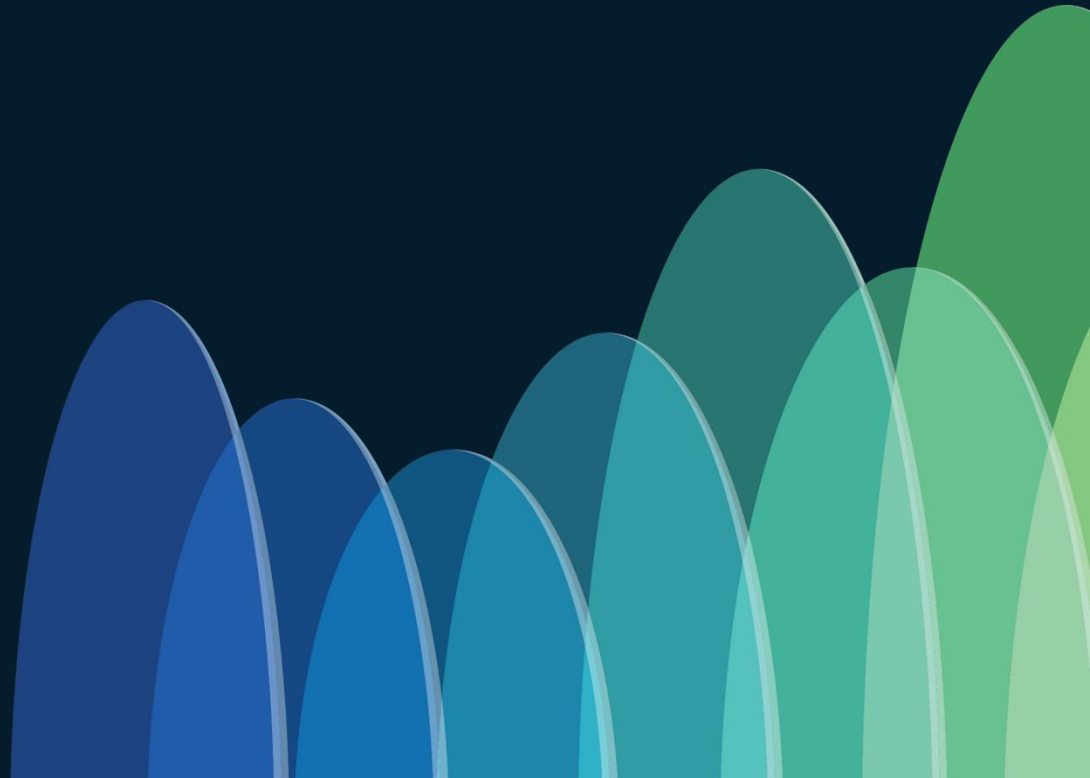
Overview Revisions Reviews Edit →



See details at <https://github.com/jeremycohoe/cisco-mdt-poe/>

<https://github.com/jeremycohoe/cisco-mdt-poe/>

Tooling



YANG Suite Capabilities

- Initial Release:
 - Plugin and YANG File Manager, Datasets and diffs
 - Device Manager
 - NETCONF (Python), gRPC Telemetry
 - Docker install support with HTTPS
- Second Release:
 - RESTCONF
 - gNMI
 - Python Integrations
- Third Release:
 - gRPC Telemetry with TLS Support
 - SNMP OID to YANG Xpath Mapping
 - Ansible Integrations
 - Pip install support

Core plugins

Additional plugins

Cisco YANG Suite

Admin

- Manage users
- Manage plugins
- View logs

Setup

- YANG files and repositories
- YANG module sets
- Device profiles

Analytics

- Datasets and diffs
- SNMP to YANG Mapping

Protocols

- gNMI
- gRPC telemetry
- NETCONF
- RESTCONF

Explore

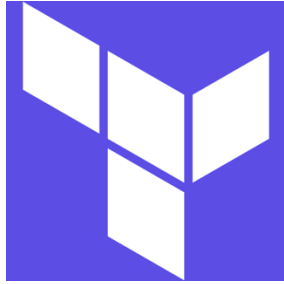
- YANG

What is Ansible?

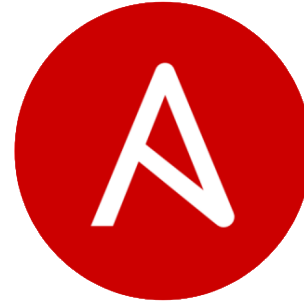


- Open-source configuration management tool
- Commercially supported by Red Hat
- Declarative and idempotent
- Ansible can be Imperative when needed
- Can manage a wide range of systems:
VMs, network devices, cloud instances, etc.
- Agentless: no requirement for installation of application software to run
- Has Python server-side dependencies

They can coexist... it's not an either/or story



- Terraform keeps state locally
- Terraform knows what is configured vs desired end-state
- Terraform can automatically destroy/recreate resources



- Ansible mutates the infrastructure
- Ansible does not keep state (mostly)
- Terraform can call Ansible to perform tasks post resource deployment (e.g., on VMs deployed)

Create a MDT Subscription using Ansible

CLI

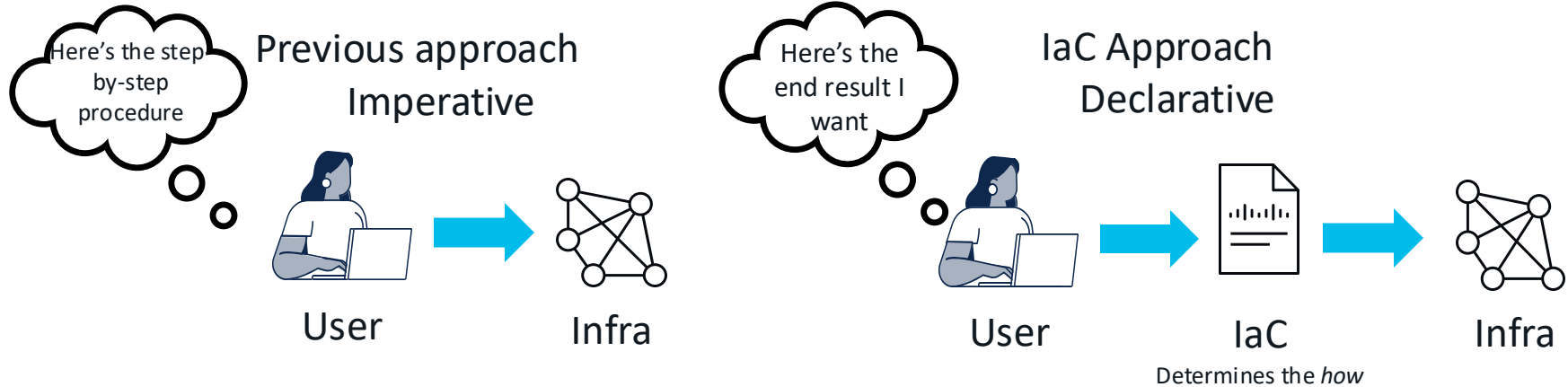
```
configure terminal
telemetry ietf subscription 1
encoding encode-kvgpb
filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization
stream yang-push
update-policy periodic 60000
receiver ip address 10.1.1.3 57500 protocol grpc-tcp
```

```
1  - name: add telemetry subscriptions
2    hosts: c9300
3    connection: netconf
4    gather_facts: no
5
6    tasks:
7      - name: establish subscription
8        netconf_config:
9          xml: |
10             <config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
11               <mdt-config-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-cfg">
12                 <mdt-subscription>
13                   <subscription-id>1</subscription-id>
14                   <base>
15                     <stream>yang-push</stream>
16                     <encoding>encode-kvgpb</encoding>
17                     <source-address>10.1.1.5</source-address>
18                     <source-vrf>Mgmt-vrf</source-vrf>
19                     <period>2000</period>
20                     <xpath>/process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds</xpath>
21                   </base>
22                   <mdt-receivers>
23                     <address>10.1.1.3</address>
24                     <port>57500</port>
25                     <protocol>grpc-tcp</protocol>
26                   </mdt-receivers>
27                 </mdt-subscription>
28               </mdt-config-data>
29             </config>
```

https://github.com/jeremycohoe/ansible-config-samples/blob/master/add_sub_17.7.yaml

What's IaC?

Infrastructure as Code (IaC) is the process of managing changes through code, rather than a manual process



Learn more about IaC here:

<https://developer.cisco.com/iaac/#:~:text=Adopting%20Infrastructure%20as%20Code%20allows,data%20center%20to%20the%20edge>

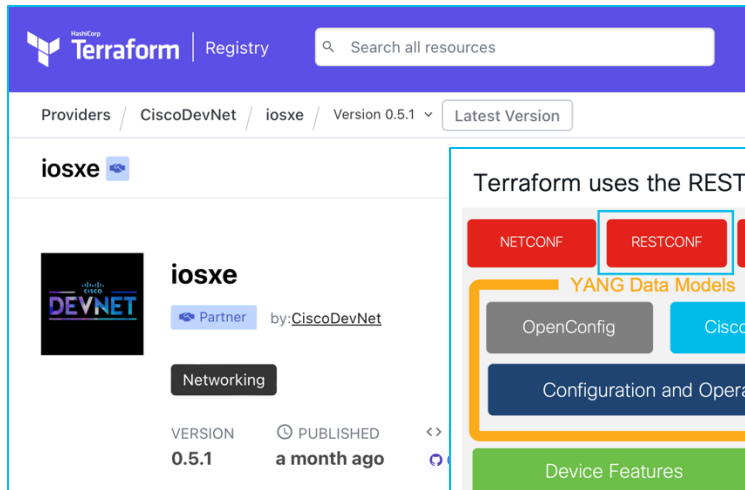
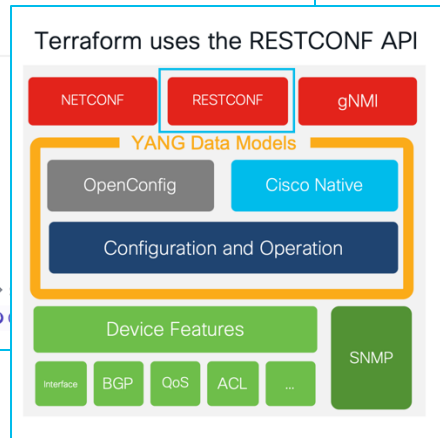
Cisco IOS XE Terraform Provider

Infrastructure as Code (IaC) Software Tool providing a consistent CLI workflow to manage hundreds of cloud services.
Terraform codifies cloud APIs into declarative configuration files.

Documentation and details about the provider are available on the Hashicorp Registry

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

Source code in GitHub Repository: <https://github.com/CiscoDevNet/terraform-provider-iosxe/>



- iosxe provider
 - > Guides
 - > AAA
 - > BFD
 - > BGP
 - > CTS
 - > Crypto
 - > EVPN
 - > General
 - > Interface
 - > MDT
 - > MPLS
 - > Management
 - > Multicast
 - > OSPF
 - > QoS
 - > Routing
 - > Switching
 - > System
 - > VRF
- > General
- > Resources
 - iosxe_cli
 - iosxe_restconf
 - iosxe_save_config
- > Data Sources
 - iosxe_restconf

Terraform for Model Driven Telemetry

Enable gRPC Dial-Out telemetry subscriptions for the POE and Basic Device Monitoring use cases

```
auto@pod22-xelab:~/ndt_examples$ terraform plan -var-file="var.tfvars"
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# iosxe_mdt_subscription.cpu_subs["100"] will be created
+ resource "iosxe_mdt_subscription" "cpu_subs" {
  + encoding      = "encode-kvgpb"
  + filter_xpath  = "/process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds"
  + id            = (known after apply)
  + receivers     = [
    + {
      + address = "10.1.1.3"
      + port    = 57500
      + protocol = "grpc-tcp"
    },
  ]
  + stream        = "yang-push"
  + subscription_id = 100
  + update_policy_periodic = 100
}

# iosxe_mdt_subscription.cpu_subs["101"] will be created
+ resource "iosxe_mdt_subscription" "cpu_subs" {
  + encoding      = "encode-kvgpb"
  + filter_xpath  = "/process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/one-minute"
  + id            = (known after apply)
  + receivers     = [
    + {
      + address = "10.1.1.3"
      + port    = 57500
      + protocol = "grpc-tcp"
    },
  ]
  + stream        = "yang-push"
  + subscription_id = 101
  + update_policy_periodic = 3000
}
```

Lab Guide Steps:

1. Introduction
2. IOS XE CLI pre-req
3. Terraform Install & headers, device variables
4. Variables and config for MDT subscription
5. Terraform configuration for device monitoring
6. TF Workflow: init, plan, apply & destroy
7. Validation with TF and CLI
8. Conclusion

iosxe_mdt_subscription (Resource)

This resource can manage the MDT Subscription configuration.

Example Usage

```
resource "iosxe_mdt_subscription" "example" {
  subscription_id = 101
  stream          = "yang-notif-native"
  encoding        = "encode-kvgpb"
  source_vrf      = "Mgmt-vrf"
  source_address  = "1.2.3.4"
  update_policy_on_change = true
  filter_xpath    = "/ios-events-ios-xe-oper"
  receivers = [
    {
      address = "5.6.7.8"
      port    = 57600
      protocol = "grpc-tcp"
    }
  ]
}
```

https://registry.terraform.io/providers/CiscoDevNet/iosxe/latest/docs/resources/mdt_subscription

https://registry.terraform.io/providers/CiscoDevNet/iosxe/latest/docs/data-sources/mdt_subscription

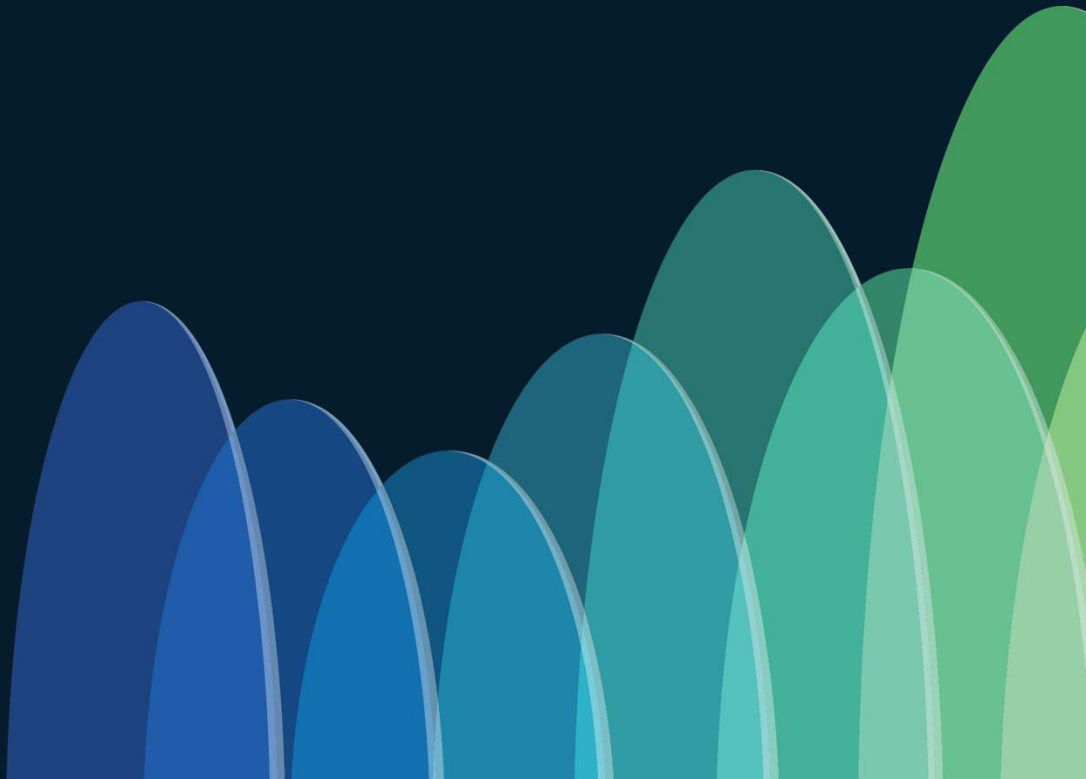
<https://github.com/jeremycohoe/cisco-ios-xe-panda-lab-terraform>

DEMO – install docker container and use Terraform files to configure telemetry subscriptions

```
auto@pod27-xelab: ~ (ssh)
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$
auto@pod27-xelab:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
auto@pod27-xelab:~$
auto@pod27-xelab:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
auto@pod27-xelab:~$
auto@pod27-xelab:~$

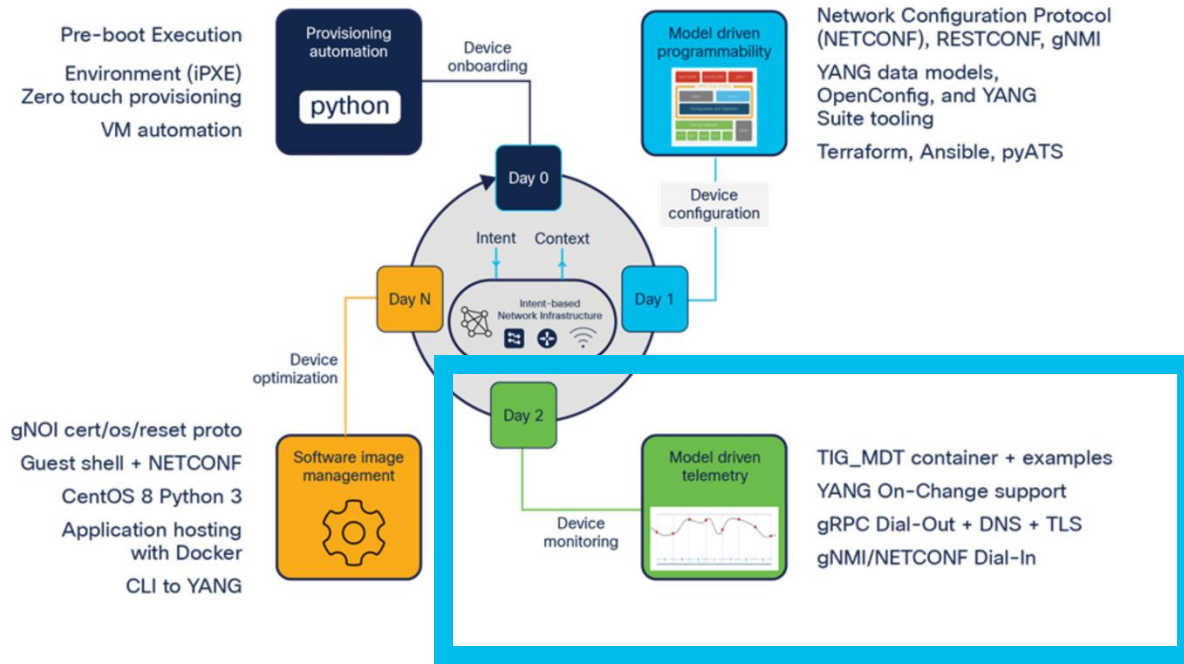
auto@pod27-xelab: ~ (ssh)
c9300-pod27# sh run | s tel
telemetry ietf subscription 6041337
encoding encode-kvgpb
filter xpath /process-cpu-ios-xe-oper:cpu-usage/cpu-utilization/five-seconds
stream yang-push
update-policy periodic 30000
receiver ip address 10.1.1.3 57500 protocol grpc-tcp
c9300-pod27#
c9300-pod27#
Oct 16 21:55:56.082: %HA_EM-6-LOG: catchall: show running-config
```

Resources



MDT White Paper

The Model Driven Telemetry White Paper includes examples, use cases and tooling related to telemetry.
This paper is now available online and in PDF form!



Website



PDF



View online: <https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-9300-series-switches/model-driven-telemetry-wp.html>
View as PDF: <https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-9300-series-switches/model-driven-telemetry-wp.pdf>

cisco *Live!*

Introduction to Cisco IOS ... ^
Table of Contents
Introduction to Cisco IOS XE
Introduction to telemetry
Benefits of model driven tele...
Network monitoring challeng...
Architecture and databases
Dial-in and dial-out MDT +
Publication notification optio...
YANG data modeling language +
Benchmarking and comparis... +
Cisco controller solutions +
Cloud solutions +
Tooling +
Dashboarding and validation +
Configuration examples +
Telemetry configuration man... +
Troubleshooting and validati... +
Best practices and lessons l... +
Conclusion
Resources

Programmability and Telemetry Deployment Guide

Cisco Catalyst 9800 Wireless LAN Controller

<https://www.cisco.com/c/en/us/products/collateral/wireless/catalyst-9800-series-wireless-controllers/catalyst-9800-programmability-telemetry-deployment-guide.html>

Table of Contents

Feature overview

Telemetry history

Cisco device compatibility

Configuration overview

NETCONF telemetry and progr...

RESTCONF programmatic interf...

gRPC Dial-Out model driven tel...

gNMI

Cisco YANG Suite

Ansible automation

Guest Shell

Conclusion

Additional resources

Reference

Questions?

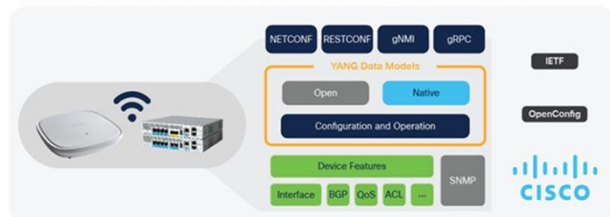
Feature overview

The world of programmability has been evolving for years, and with the latest Cisco® IOS XE releases, we've included new Yet Another Next Generation (YANG) models to bring additional automation to wireless technology. With the use of APIs, interacting with devices and retrieving data got much easier. Back in the day, we used many commands sent from a Command Line Interfaces (CLIs) to communicate with the software. In addition, the Simple Network Management Protocol (SNMP) is frequently used for network management. Fast forward to today, we use the new way to interact with the software commonly called an API. Even though CLIs and SNMP are widely used, they are not as efficient and scalable as APIs.

This document will dive into the different programmable interfaces used to communicate with the various Cisco devices and specifically target the Catalyst® 9800 Wireless LAN Controller. We will discuss the pros and cons of using Network Configuration Protocol (NETCONF), Representational State Transfer Configuration (RESTCONF), and the gRPC Network Management Interface/Google Remote Procedure Call (gNMI/gRPC) protocols, and the main differences between them.

The Catalyst 9800 IOS XE-based Wireless LAN Controller (WLC) has several options for programmatic configuration. Traditional methods for configuring the WLC include the CLI and WebUI, but that has now been expanded to include the programmable interfaces. These programmable interfaces include NETCONF, RESTCONF, and the gNMI/gRPC protocols. YANG data models define what data is accessible over the programmable interfaces, and they come in several varieties. Cisco IOS XE features are defined within the Native data models, while standard and vendor agnostic features are defined within the open data models that are mapped to Native data models. Either model can be used for many tasks; however, features specific to IOS XE are available only in the Native models.

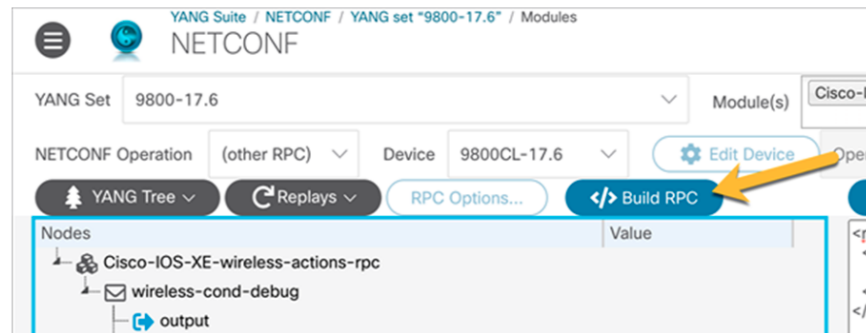
Models created by Cisco are referred to as Native data models since they are specifically created for devices and software with which they are associated. The Native data models provide most comprehensive and operational coverage for device functionality.



Enable wireless conditional debugging

Based on either the MAC or IP address of a device we can generate debug messages using the `Cisco-IOS-XE-wireless-actions-rpc.yang` model. This is equivalent to entering the following CLI command:

```
debug platform condition feature wireless <mac/ip> <MAC/IP Address>
```



Updated TIG_MDT container now available!

```
docker pull jeremycohoe/tig_mdt
docker run -ti -p 3000:3000 -p 57500:57500 jeremycohoe/tig_mdt
```

Cisco IOS XE Devices



Collector/Receiver

Decodes to text



Storage

Time Series Database



Monitoring

and Visualisations



Upgrade coming to Telegraf, Influx, and Grafana Model Driven Telemetry (TIG_MDT) Docker container

Making it easier to consume telemetry in production

Upgraded Telegraf, InfluxDB, and Grafana tools
Additional dashboards for

Device Health, Wireless Client, Wireless AP, RF etc

Examples for device CLI configuration for telemetry

Details of scale and data storage requirements



https://hub.docker.com/r/jeremycohoe/tig_mdt

<https://github.com/jeremycohoe/cisco-ios-xe-mdt>

https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/179/b_179_programmability_cg/m_179_prog_ielf_telemetry.html

cisco *Live!*



Learning Lab and Blog: IOS XE MDT

https://developer.cisco.com/learning/modules/iosxe_telemetry

<https://blogs.cisco.com/developer/model-driven-telemetry-sandbox>

Learning Labs

DevNet Express Tracks **Modules** Labs Challenges Help Feedback

IOS XE Model Driven Telemetry

Cisco IOS XE is the Network Operating System for the Enterprise. It runs on switches like the Catalyst 9000, routers like the AIR-CT5500, Catalyst 9000, and ISR 4000 and 4000T, Catalyst 9000 Wireless LAN controllers, as well as a few other devices in IoT and Cable product lines. Since the IOS XE 16.6 release, there has been support for model-driven telemetry, which provides network operators with additional options for getting information from their network.

4 Learning Labs
1 Hour 20 Minutes

Choose a learning lab to start learning

Introduction to Telemetry on IOS XE 20 min

Introduction to Telemetry on IOS XE

Telemetry Python GraphQL SDN REST Network Automation

Start Lab

Enabling Telemetry On IOS XE 20 min

Enabling Telemetry On IOS XE

Telemetry Python GraphQL SDN REST Network Automation

Start Lab

Yang Explorer 20 min

Yang Explorer

Telemetry Python GraphQL SDN REST Network Automation

Start Lab

Building Grafana with increase Telemetry 20 min



Developer

Explore Model-Driven Telemetry

Stuart Clark

New learning labs and sandbox

As our journey through network automation grows, so does the need for our network tools. Network Engineers have always been considered the absolute escalation point for any performance difficulties and problems, irrespective whether the root cause is really the network, server, or application. Network Engineers are expected to have the knowledge and tools to isolate and identify the issue, collaborating with other teams such as SRE / AppDev to bring it to resolution and often present this in an RCA (root cause analysis).

One of these great tools which can really help is telemetry. In software, telemetry is used to gather data on the use and performance of applications and application components, e.g. how often certain features are used, measurements of start-up time and processing time, hardware, application crashes, and general usage statistics and/or user behavior.

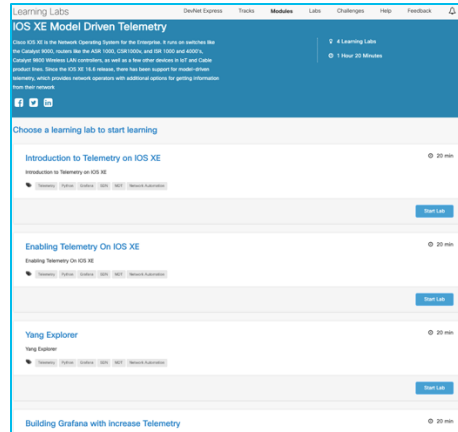
Learning Lab and Blogs: Telemetry

https://developer.cisco.com/learning/modules/iosxe_telemetry

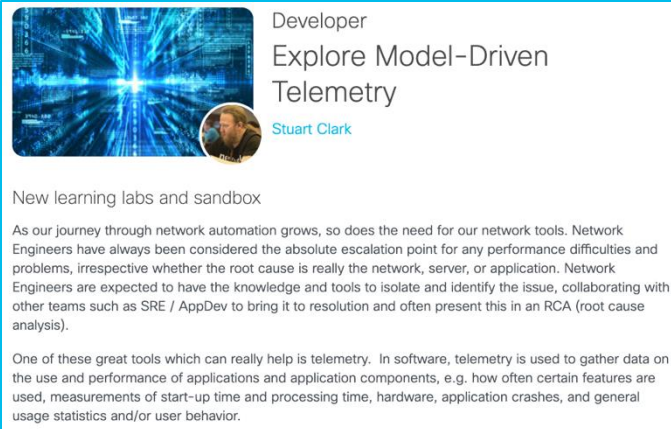
<https://blogs.cisco.com/developer/model-driven-telemetry-sandbox>

<https://blogs.cisco.com/developer/getting-started-with-model-driven-telemetry>

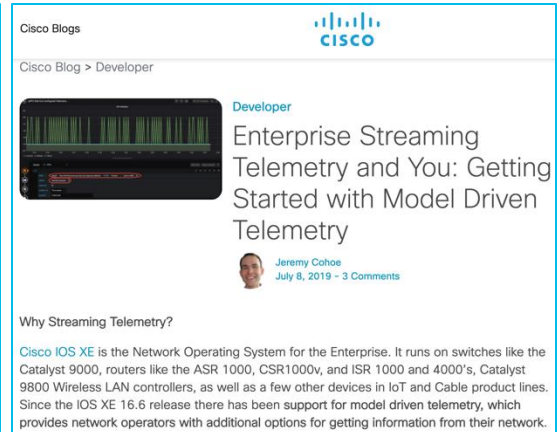
<https://youtu.be/QwwZakkWBng>



The screenshot shows the Cisco Learning Labs interface. At the top, it says "Learning Labs" and "IOS XE Model Driven Telemetry". Below this, there's a section "Choose a learning lab to start learning" with a list of labs. The first lab is "Introduction to Telemetry on IOS XE" with a duration of 22 min. The second lab is "Enabling Telemetry On IOS XE" with a duration of 22 min. The third lab is "Yang Explorer" with a duration of 22 min. The fourth lab is "Building Grafana with Increase Telemetry" with a duration of 22 min. Each lab has a "Start Lab" button.



The image shows a blog post from the Cisco Developer site. The title is "Explore Model-Driven Telemetry" by Stuart Clark. The post features a blue abstract image with a person's profile. The text discusses the need for network tools as automation grows and introduces new learning labs and a sandbox. It mentions that network engineers have always been considered the absolute escalation point for performance difficulties and that network engineers are expected to have the knowledge and tools to isolate and identify the issue, collaborating with other teams such as SRE / AppDev to bring it to resolution and often present this in an RCA (root cause analysis). It also states that in software, telemetry is used to gather data on the use and performance of applications and application components, e.g. how often certain features are used, measurements of start-up time and processing time, hardware, application crashes, and general usage statistics and/or user behavior.



The image shows a blog post from the Cisco Blog. The title is "Enterprise Streaming Telemetry and You: Getting Started with Model Driven Telemetry" by Jeremy Cohoe. The post features a Cisco logo and a screenshot of a network diagram. The text discusses the need for network tools as automation grows and introduces new learning labs and a sandbox. It mentions that network engineers have always been considered the absolute escalation point for performance difficulties and that network engineers are expected to have the knowledge and tools to isolate and identify the issue, collaborating with other teams such as SRE / AppDev to bring it to resolution and often present this in an RCA (root cause analysis). It also states that in software, telemetry is used to gather data on the use and performance of applications and application components, e.g. how often certain features are used, measurements of start-up time and processing time, hardware, application crashes, and general usage statistics and/or user behavior.

Cisco YANG Suite



YANG API Testing and Validation Environment

Construct and test YANG based APIs over NETCONF,
RESTCONF, gRPC and gNMI

IOS XE / IOS XR / NX OS platforms

Get hands-on using the new learning lab!

<https://developer.cisco.com/learning/labs/intro-yangsuite/>

Docker container innovation 1 container

cisco *Live!*

YANG Suite

YANG Suite / Explore YANG Models

Select a YANG set: C9300 Select YANG module(s): Cisco-IOS-XE-interfaces-oper

Icon legend Search XPath Search nodes Expand all nodes

Display schema nodes only Display all nodes

Node Properties

Name	statistics
container	
Description	A collection of interface-related statistics objects
Module	Cisco-IOS-XE-interfaces-oper
Revision	2020-07-01
Xpath	/interfaces/interface/statistics
Prefix	interfaces-ios-xe-oper
Namespace	http://cisco.com/ns/yang/Cisco-IOS-XE-interfaces-oper

YANG Suite

YANG Suite / NETCONF / YANG set "C9300" / Modules

YANG Set: C9300 Module(s): Cisco-IOS-XE-interfaces-oper

NETCONF Operation: get Device: JCOHOE-DMZ-C9300 Edit Device Open Device Window

Build RPC Replays RPC Options... Run RPC(s) Clear RPC(s)

Nodes

Nodes	Value
Cisco-IOS-XE-interfaces-oper	
interfaces	
interface	
name	string
interface-type	
admin-status	
oper-status	
last-change	
if-index	
phys-address	
higher-layer-if	
lower-layer-if	
speed	
statistics	

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
    <filter>
      <interfaces xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-interfaces-oper">
        <interface>
          <name/>
          <statistics/>
        </interface>
      </filter>
    </get>
  </rpc>
```

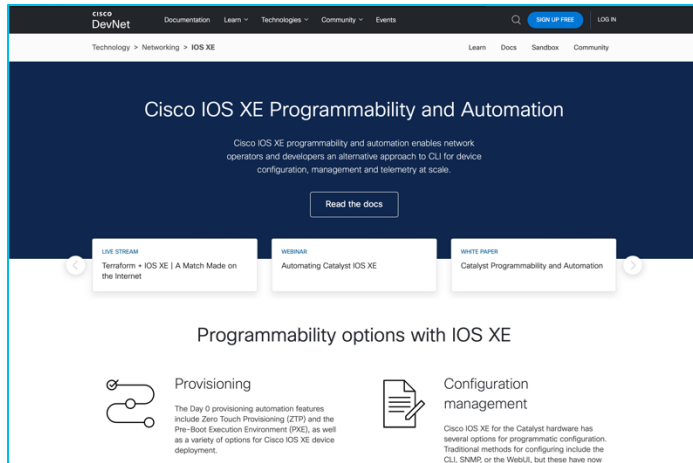
developer.cisco.com/yangsuite

github.com/CiscoDevNet/yangsuite



Programmability Website

The one-stop-shop for Cisco IOS XE Programmability resources including videos, white papers, labs and more!



- Community Forum
- IOS XE FAQ
- White Papers
- Code Exchange
- IOS XE Docs & Guide
- Learning Tracks and Labs
- Sandboxes
- ... and more !



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

API White Paper

Programmability and auto... ^ Q

Table of Contents

Programmability and automatio... -

- Day 0: Provisioning automation
- Day 1: Model-driven programmability
- Day 2: Model-driven telemetry
- Day N: Device optimization
- Cisco IOS XE operational consistency

Yet Another Next Generation (Y... +

- Day 1: Model-driven program... +
- Tooling: Cisco YANG Suite +
- Day 2: Model-driven telemetry +
- Day N: Device optimization +

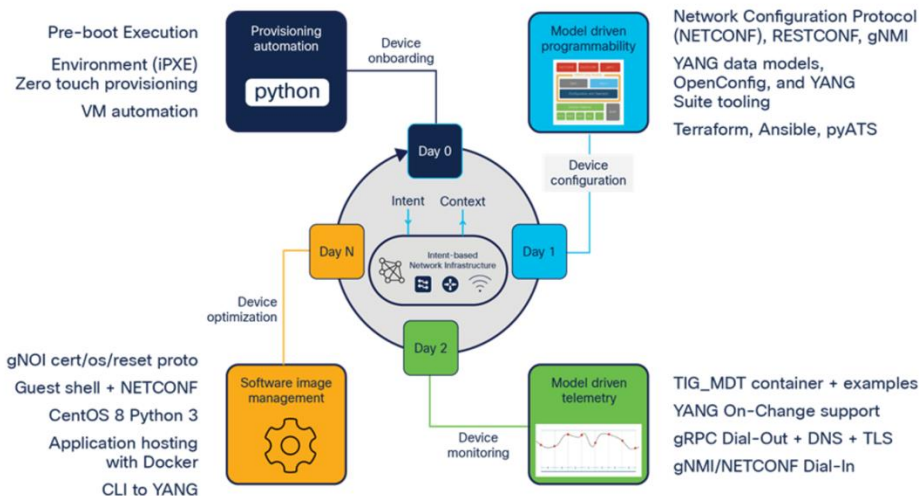
Conclusion

Additional resources +

Blogs

Products & Services / Switches / Campus LAN Switches - Access / Cisco Catalyst 9300 Series Switches /

Catalyst Programmability and Automation



Website: <https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-9300-series-switches/nb-06-catalyst-programmability-automation-wp.html>

PDF: <https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-9300-series-switches/nb-06-catalyst-programmability-automation-wp.pdf>

Webinar with live demos & examples: <https://www.youtube.com/watch?v=Idck5PnPu2I>

cisco *Live!*

<http://cs.co/apiwp>



<http://cs.co/apiwp.pdf>

dCloud Programmability

<https://dcloud.cisco.com>

"Cisco Catalyst 9000 IOS XE Programmability & Automation Lab v1"

<https://dcloud2.cisco.com/demo/catalyst-9000-ios-xe-programmability-automation-lab-v1>

Use Cases:

EVPN:

Ansible with CLI deployment of EVPN solutions
EVPN management over RESTCONF/YANG with Postman
Declarative EVPN fabric management with Terraform

Tooling and Integrations

YANG Suite

- NETCONF/RESTCONF/gNMI API
 - Ansible integration
- NETCONF/gNMI Dial-In Telemetry
- gRPC Dial-Out Telemetry receiver

Telemetry

- TIG stack in Docker
- Grafana dashboard for device health

Postman / RESTCONF

- EVPN fabric API calls

Terraform/RESTCONF

- Declarative EVPN fabric management

Ansible

- EVPN solution enablement using CLI

Model Driven Telemetry

Telemetry configuration with CLI and YANG Suite
Collection with TIG_MDT container and tooling

YANG Programmability

YANG Suite tooling and integrations to YANG API's
Ansible integrations

Ubuntu VM Details:

Syslog receiver from all switches
TFTP config backup
See slide

Windows VM Details

VS Code

Terraform @ folder
Ansible @ folder

Chrome browser

YANG Suite, Grafana

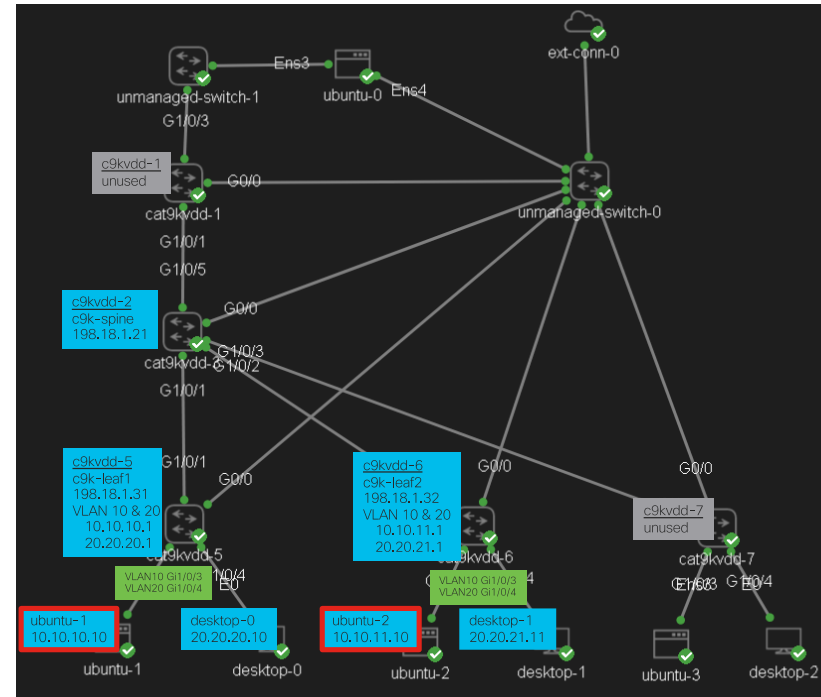
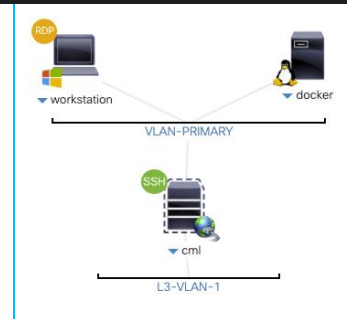
Bash/PS/Cmd shells

SSH into C9K or Ubuntu

Postman

Workspace for EVPN

C9K VM's



VLAN1
c9k-spine
IP: 198.18.1.21
developer / C1sco12345

c9k-leaf1
IP: 198.18.1.31
developer / C1sco12345

c9k-leaf2
IP: 198.18.1.32
developer / C1sco12345

c9kvdd-1 - unconfigured

c9kvdd-7 - unconfigured

cisco *Live!*

Cisco IOS XE - Reservable Virtual Sandbox

This DevNet reservable sandbox has IOS XRv + N9Kv + IOS XE
The Catalyst 8000V within the DevBox is enabled for Day 0/ZTP usecases

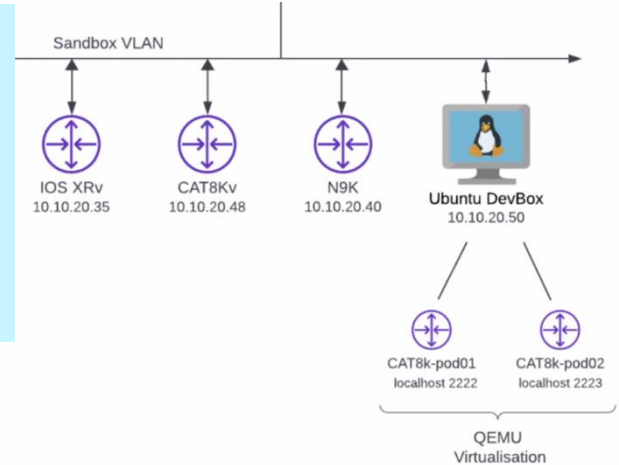
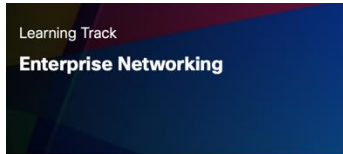
<https://devnetsandbox.cisco.com>

“IOS XE on Cat8kv”

<https://devnetsandbox.cisco.com/DevNet/catalog/IOS%20XE%20on%20Cat8kv>

The “Enterprise Networking” Learning Labs contains guides for the supported usecases

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



IOS XE on Cat8kv

The IOS XE on Catalyst 17.9.2 Sandbox offers developers access to an IOS XE device running the latest IOS XE code release available on cisco.com. Here you can test out the newest programmability...

Networking

reservable



Launch

6 hrs - 4 modules - 12 labs

Sandbox Capabilities:

TIG_MDT Telemetry

YANG Suite API

Terraform + Ansible

ZTP & Guest Shell

... and more

Table of Contents

- Overview
- Network Topology
- Grafana Dashboard
- Access Details
- Additional Resources
- VPN Access

Catalyst 8000v Credentials

host	port	username	password
10.10.20.48	22	developer	C1sco12345
10.10.20.48	830	developer	C1sco12345
10.10.20.48	443	developer	C1sco12345

Developer Box Credentials

host	port	username	password
10.10.20.50	22	developer	C1sco12345
telnet localhost (CAT8k-1)	2222	developer	C1sco12345
telnet localhost (CAT8k-2)	2223	developer	C1sco12345

Cisco IOS XE - Always On Virtual Sandbox

The “IOS XE on Cat8kv Always On” virtual sandbox requires no reservation

The c8kv can be accessed with SSH, NETCONF, and RESTCONF

Hostname: sandbox-iosxe-latest-1.cisco.com

Username: admin Password: C1sco12345

SSH port 22, NETCONF port 830, RESTCONF HTTPS

<https://devnetsandbox.cisco.com>

devnetsandboxiosxe.cisco.com

<https://devnetsandbox.cisco.com/DevNet/catalog/ios-xe-always-on>

The “Enterprise Networking” Learning Labs contains guides for the supported usecases

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

IOS XE on Cat8kv AlwaysOn



This AlwaysOn sandbox offers developers access to an IOS XE device running the latest IOS XE code release available on cisco.com (currently 17.11.x). Here you can test out the newest programmability feature...

Always-On



Launch

Access Details:

Developers and network engineers access the IOS XE on Catalyst 17.11.x sandbox directly using the following information:

- Cat8000v Host
 - Address: `sandbox-iosxe-latest-1.cisco.com`
 - Username: admin
 - Password: C1sco12345
 - NETCONF port: 830
 - gRPC telemetry port: 57500
 - ssh port: 22

Learning Track

Enterprise Networking

Explore YANG data models and NETCONF to query and configure IOS XE network devices. Then use Ansible to manage configurations on your IOS XE devices, covering the fundamentals of Ansible. Lastly, enable...

6 hrs - 4 modules - 12 labs

Cisco University (Cisco U) part of L&D

<https://u.cisco.com>

<https://u.cisco.com/search/tutorial?query=Story%20DeWeese,%20Jeremy%20Coho,%20not%20berry>

Direct link to Tutorial, requires login to u.cisco.com first:

1. <https://ondemandlearning.cisco.com/apollo-alpha/tc-iosxe-ztp/pages/1>
2. <https://ondemandlearning.cisco.com/apollo-alpha/tc-terraform-ios-xe/pages/1>
3. <https://ondemandlearning.cisco.com/apollo-alpha/tc-yangsuite-netconf/pages/1>
4. <https://ondemandlearning.cisco.com/apollo-alpha/tc-yangsuite-restconf/pages/1>

Cisco U.

Welcome to Cisco U.

You're in the right place, whether you're looking to earn a certification or gain new skills. In Cisco U., you'll find courses, community, and learning content to help you reach your goals.

Learn more about Cisco U., or come on in and get started!

Log in with your Cisco Account ID

Create a free account

cisco *Live!*

TUTORIAL

 Jeremy Coho

Cisco IOS XE Zero-Touch Provisioning

In this tutorial, you will learn the basics of the IOS XE ZTP feature.

Beginner 45m

Created October 23, 2023

TUTORIAL

 Story DeWeese

Using YANG Suite with NETCONF

In this tutorial, you'll learn how to use YANG Suite to visualize the NETCONF protocol with network devices.

Intermediate 30m

Created October 24, 2023

TUTORIAL

 Story DeWeese

Using YANG Suite with RESTCONF and gRPC

In this tutorial, you'll learn how to use YANG Suite to visualize the RESTCONF and gRPC protocols with network devices.

Intermediate 10m

Created October 24, 2023

TUTORIAL

 Story DeWeese

Cisco IOS XE Terraform Provider

Get started leveraging the power of Terraform. In this tutorial, we'll use this cloud-native provisioning tool to create a new VLAN.

Beginner 15m

Created February 21, 2023

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



Thank you

CISCO *Live!*

#CiscoLiveAPJC

CISCO *Live!*

GO BEYOND

#CiscoLiveAPJC