The bridge to possible

Benefits of Streaming Telemetry

In monitoring and troubleshooting Cisco NXOS devices

Aakriti Saxena, Network Consulting Engineer
@AakritiSaxena04

DEVNET-3010

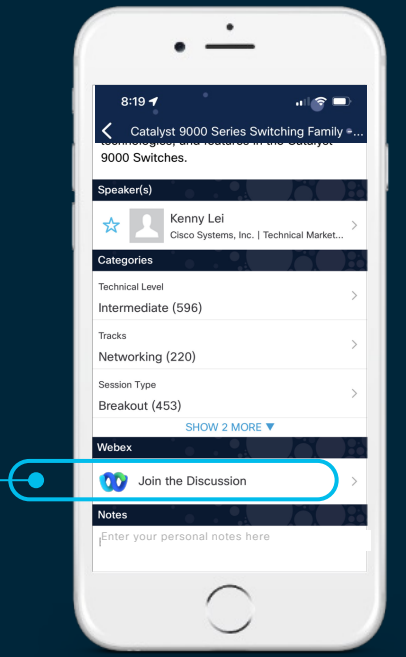
Cisco Webex App

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



https://ciscolive.ciscoevents.com/ciscolivebot/#DEVNET-3010

Webex spaces will be moderated by the speaker until June 17, 2022.

Agenda

- Monitoring Models and Telemetry
- Telemetry Data Collection
- Hardware Telemetry Sources and Platform Telemetry
- Software Telemetry Sources and Detailed Configuration
- Open-Source Tools
- Demo and Conclusion

Session Abstract

The traditional telemetry methods like SNMP, CLI and Syslog have limitations which prevents the scale that data demands at present. SNMP polling can take about 5-10 minutes at times, CLIs are unstructured and can vary as the code grows or the device changes, which is not good for scripts that can be run on the devices.

Cisco NXOS streaming telemetry allows to continuously push data off of the device to a defined endpoint as JSON at a much higher frequency and efficiency. The streaming telemetry feature notifies the client, providing near-real-time access to monitoring data. This data properly stored can be used in troubleshooting and finding the root cause (RCA) of issues that are seen. This will help in faster resolution of problems along with potential prevention of the same.

Collecting data for analysing and troubleshooting has always been an important aspect in monitoring the health of a network. Cisco NXOS streaming telemetry is the perfect solution.

Traditional NX-OS Monitoring Methods

SNMP

Can take 5-10 mins for polling

CLI

Often unstructured

Syslogs

Limitations that restrict automation and scaling

Telemetry

Push Model

- Sends **relevant** data,
  - as Fast
  - as Useful
  - as Easy
  - as Efficient
- Format – JSON or GPB
- Highly available

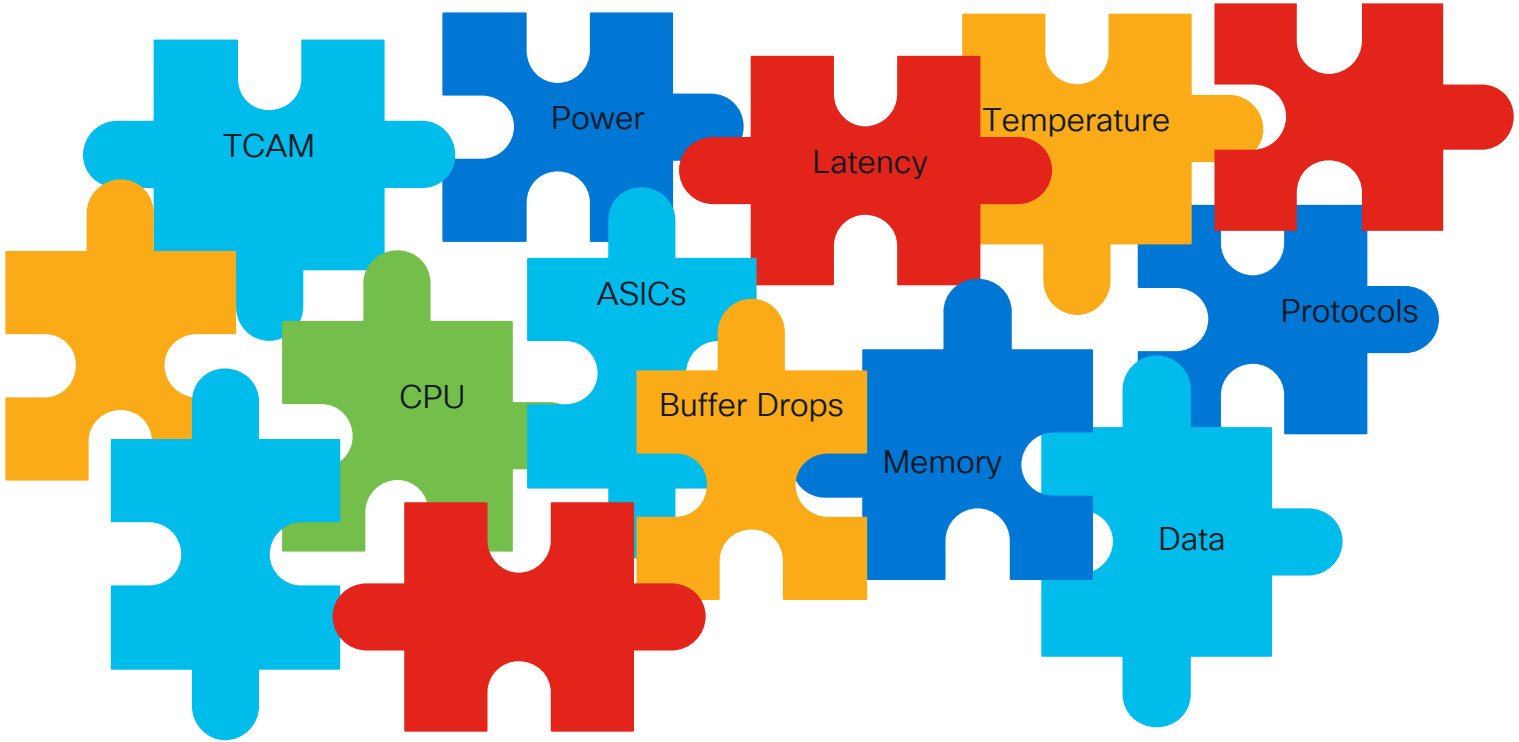
Using Telemetry

- No License
- ‘feature telemetry’
- Up to five receivers are supported.
- Configure via NX-API or CLI after 7.0(3)I5(1)

Components of Telemetry

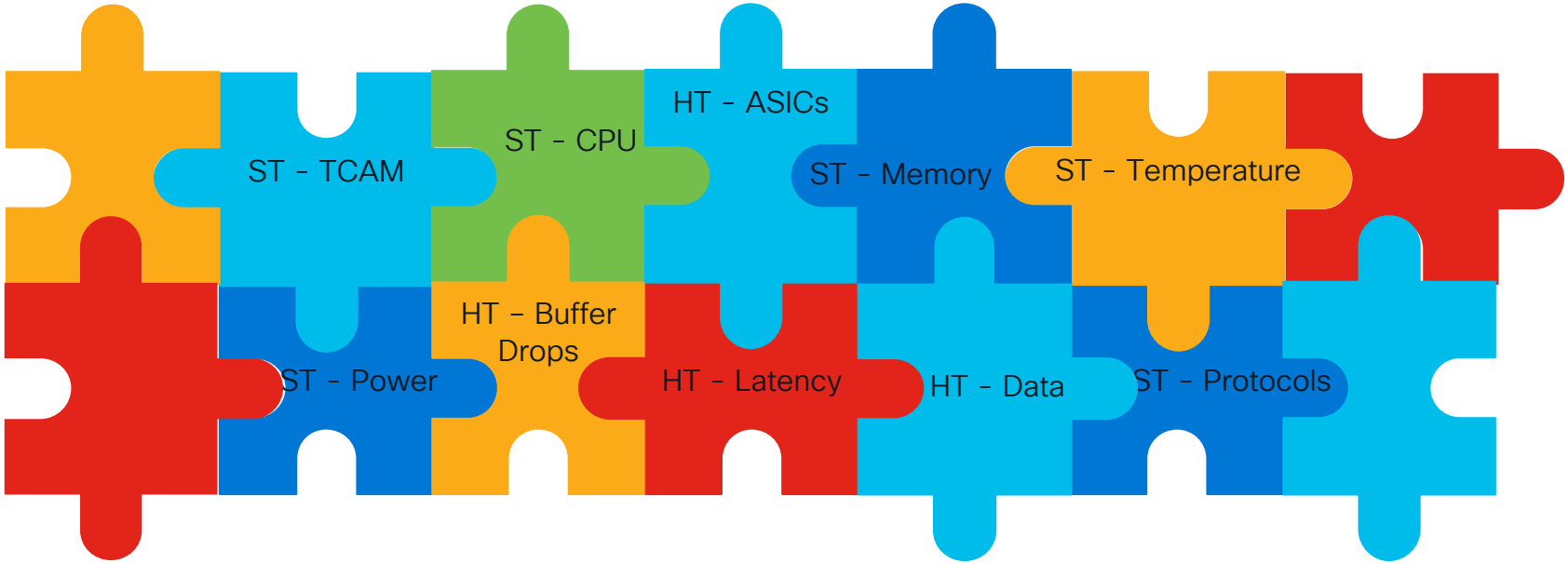
- Data Collection from DME
- Data Encoding in JSON or GPB
- Data Transport using HTTP for JSON encoding and gRPC for GPB encoding
- Telemetry Reciever

Judging health of devices



Telemetry Data Collection Methods – Hardware and SoftwareTelemetry

Hardware and Software Telemetry

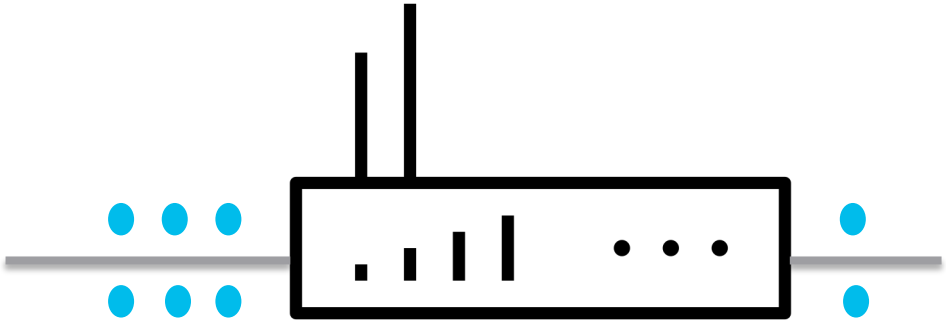


Hardware Telemetry Sources

Streaming Statistics Export (SSX)

User specified monitoring statistics include:

- Queue congestion
  - egress/ingress queue depth,
  - egress queue peak,
  - egress buffer depth
- Queue drops
  - egress/ingress queue drops
- RMON counters
  - ethernet counters



SSX - Configuration Example

```
switch(config)# feature hardware-telemetry
switch(config)# hardware-telemetry ssx
switch(config)# ssx exporter e
switch(config-ssx-collector)# source x.x.x.x
switch(config-ssx-collector)# destination x.x.x.x use-vrf default
switch(config-ssx-collector)# transport udp src-port xx dst-port yy
switch(config-ssx-collector)# mtu 1500
switch(config-ssx-collector)# dscp 0
switch(config)# ssx record all
switch(config-ssx-record)# collect egress queue depth
switch(config-ssx-record)# interval 100
switch(config)# ssx monitor all
switch(config-ssx-monitor)# exporter exporter1
switch(config-ssx-monitor)# record all
switch(config)# ssx system monitor all
switch(config)# ssx system system-id 2
```

- Enable SSX feature
- Configure SSX Exporter
- Configure SSX Record
- Configure SSX Monitor
- Apply SSX

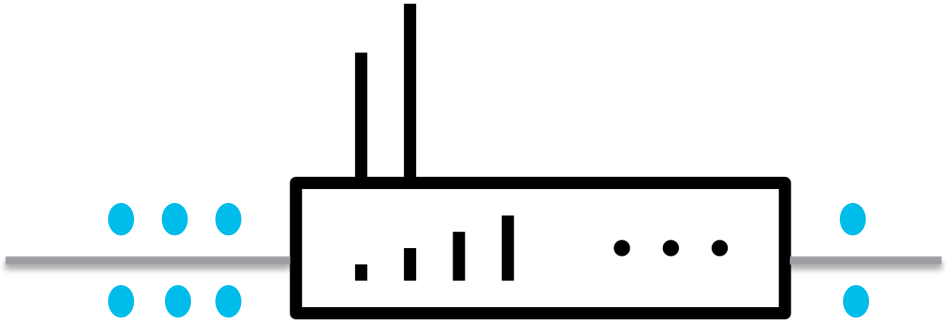
Flow Table Analytics

After Cisco NX-OS 9.3(1) release

Capture all flow details in the packets of a flow to support the analytics feature.

Flow filter (named-ACLs) control the flow to be monitored.

Drop event streams have to be defined in FT filter as without FT filter, drop-events will not export.



Flow Table - Configuration Example

```
switch(config)# feature analytics
switch(config)# flow exporter <exporter1>
```

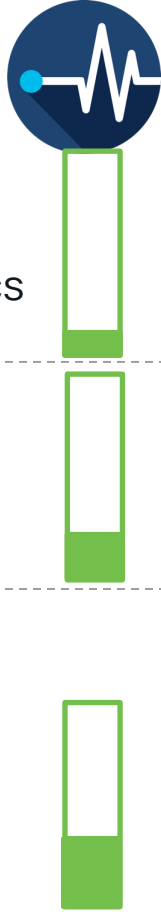
- Enabling the analytics feature

```
switch (config-flow-exporter)# destination x.x.x.x use-vrf <default>
switch (config-flow-exporter)# transport udp <destination_port>
switch (config-flow-exporter)# source loopback
<source_interface_for_destination>
```

- Configuring the flow exporter

```
switch(config)# flow record <record1>
switch (config-flow-record)# match ip source address
switch (config-flow-record)# match ip destination address
switch (config-flow-record)# match transport source-port
switch (config-flow-record)# match transport destination-port
switch (config-flow-record)# collect counter bytes
switch (config-flow-record)# collect counter packets
```

- Configuring the flow record



Flow Table – Configuration Example

```
switch(config)# flow monitor <monitor1>
switch (config-flow-monitor)# record <record1>
switch (config-flow-monitor)# exporter-bucket-id <id : 1-255> <hash range : 0-65535>
switch (config-flow-monitor-eb)# exporter <exporter1>
```

- Configuring the flow monitor

```
switch(config)# feature analytics
switch(config)# flow profile <profile1>
switch (config-flow-profile)# collect interval <100/1000/2000 msec>
switch (config-flow-profile)# source port <port>
switch(config)# feature analytics
```

- Configuring the flow profile

```
switch(config)# flow filter <filter1>
switch (config-flow-filter)# ipv4/6 <named-acl>
```

- Configuring the flow filter (ACLs)

Flow Table - Configuration Example

```
switch(config)# feature analytics
switch(config)# flow system config
switch(config-flow-system)# exporter-id <id>
switch(config-flow-system)# monitor <monitor1> input
switch(config-flow-system)# profile <profile1>
switch(config-flow-system)# filter <filter1>
```

- Applying the flow monitor

```
switch# configure terminal
switch (config)# vrf context <vrfName>
switch (config-vrf)# flow filter <filter1>
```

- Applying the Flow Monitor at VRF Level

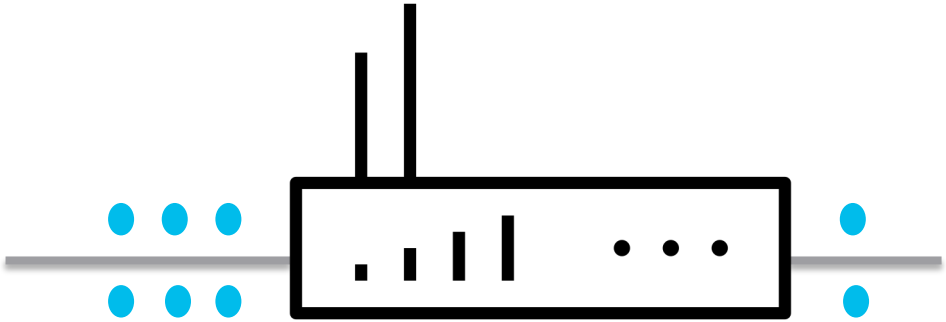
```
switch# configure terminal
switch (config)# int ethernet <intNo.>
switch (config-if)# flow filter <filter1>
```

- Applying the Flow Monitor at Interface Level

Flow Table Events

After Cisco NX-OS 9.2(1) release, capture

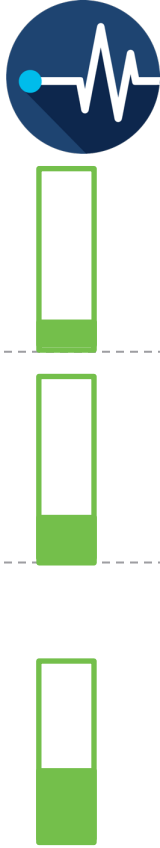
- ACL drops
- Buffer drops
- Forward drops
- Latency
- TTL
- TOS
- IP-DF – Generate event if fragment bit is enabled.



FTE- Configuration Example

```
switch(config)# feature analytics
switch(config)# flow exporter fte-exp
switch(config-flow-exporter)# destination x.x.x.x
switch(config-flow-exporter)# transport udp <DestPortNo>
switch(config-flow-exporter)# events transport udp <eventDestPortNo>
switch (config-flow-exporter)# source loopback <int>
switch(config)# flow record <fteRecord>
switch (config-flow-record)# match ip source address
switch (config-flow-record)# match ip destination address
switch (config-flow-record)# match transport source-port
switch (config-flow-record)# match transport destination-port
switch(config-flow-record)# collect timestamp sys-uptime first
switch(config-flow-record)# collect timestamp sys-uptime last
```

- Enabling the FTE feature
- Configuring the FTE exporter
- Configuring the FTE record



FTE- Configuration Example

```
switch(config)# flow monitor <ftemonitor1>
switch (config-flow-monitor)# record <fterecord1>
switch (config-flow-monitor)# exporter-bucket-id <id : 1-255> <hash range : 0-65535>
switch (config-flow-monitor-eb)# exporter <fteexporter1>
switch(config)# flow profile <fteprofile1>
switch (config-flow-profile)# collect interval <100/1000/2000 msec>
switch (config-flow-profile)# source port <port>
switch(config-flow-profile)# flow event fte-event1
switch(config-flow-event)# group drop-events
switch(config-flow-event-drop-events)# capture buffer-drops
switch(config-flow-event-drop-events)# capture acl-drops
switch(config-flow-event-drop-events)# capture fwd-drops
switch(config-fte-event-drop-events)# group latency-eventsswitch(config-fte-event-latency-events)# capture latency exceeding-threshold 100 micro-sec
```

- Configuring the FTE monitor
- Configuring the FTE profile
- Configuring the FTE Event



FTE- Configuration Example

```
switch(config-flow-event-latency-events)# capture latency exceeding-
threshold 1 milli-sec
```

- Configure FTE Latency Events

```
switch(config-flow-event-packet-events)# flow system
switch(config-flow-system)# exporter-id 3999
switch(config-flow-system)# switch-latency
switch(config-flow-system)# monitor ftemonitor1 input
switch(config-flow-system)# profile fteprofile1
switch(config-flow-system)# event fte-event1
```

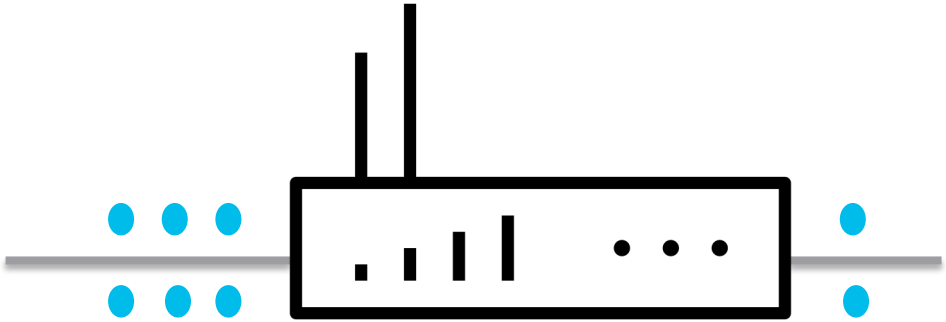
- Configure FTE System

Platform Telemetry

Platform Telemetry

After Cisco NX-OS 10.2(x) release,

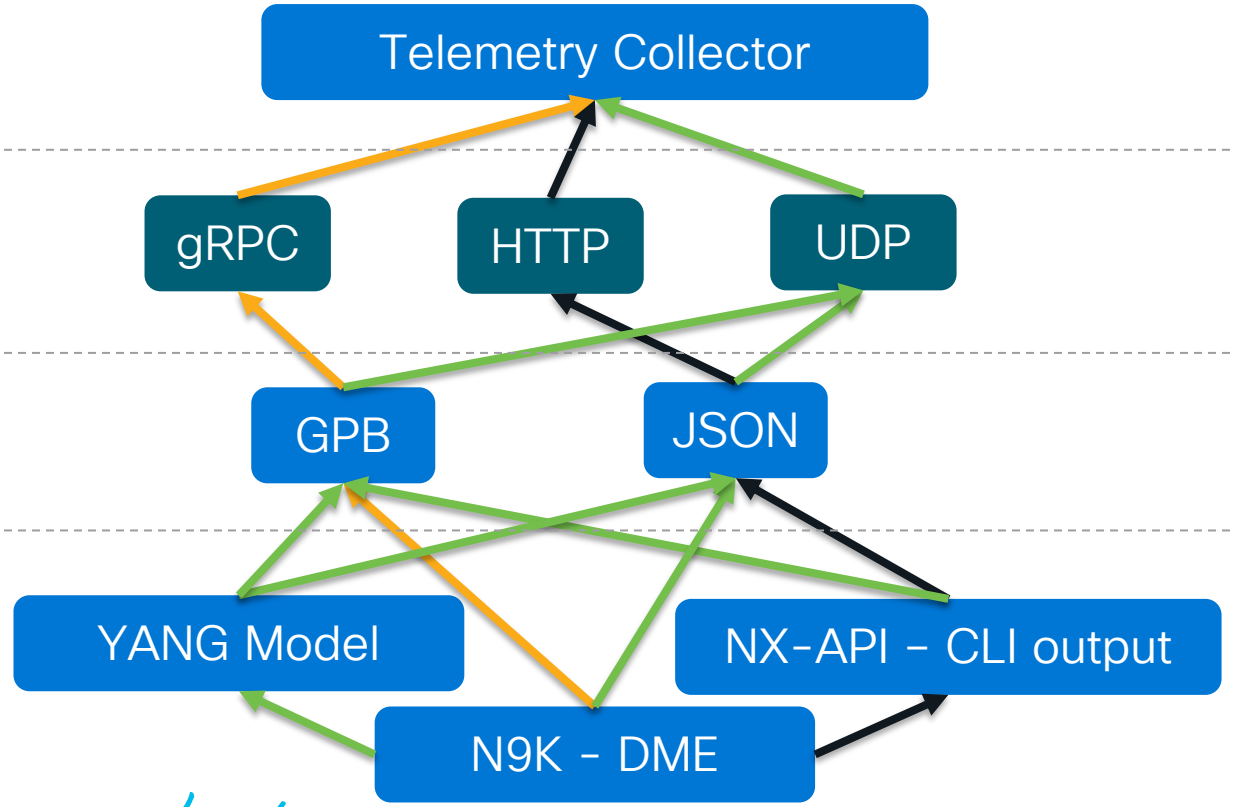
- PSU
- Fans
- Sensors
- CPU and Memory Usage



- show **pie eventdb psu/fan/cpu-usage/mem-usage**

Software Telemetry Sources

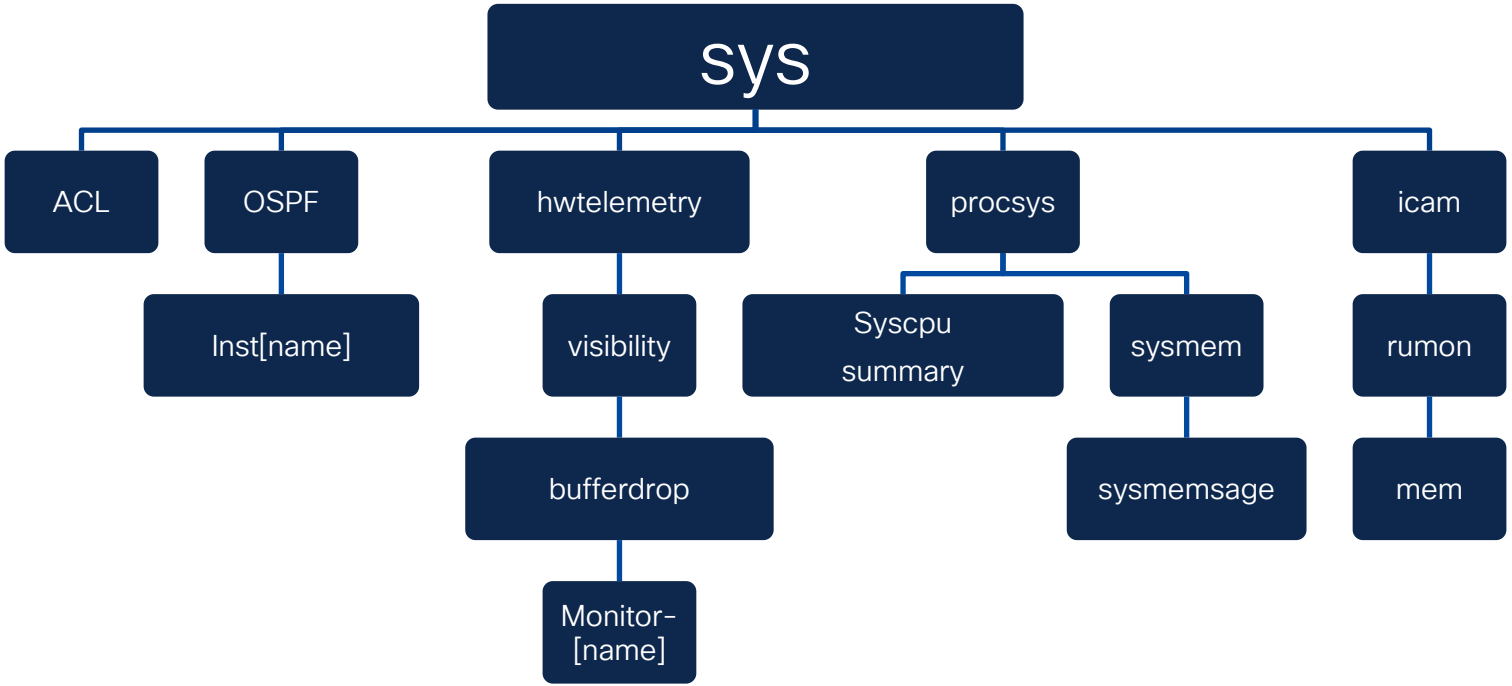
NX-OS Telemetry Data Flow



- Transport Protocol
- Encoding Mechanism
- Data Collection Mechanism

Telemetry Sources - DME

NX-OS release 7.0(3)I5(1)



Data Management Engine

NX-OS release 9.3(1) supports path labels

bgpEntity	
adminSt	enabled
childAction	
dn	sys/bgp
lcOwn	local
modTs	2015-09-15T17:39:48.628+00:00
monPolDn	uni/fabric/monfab-default
name	
operErr	
operSt	enabled
status	
uid	0

bgpInst	
activateTs	2015-09-15T17:39:50.819+00:00
adminSt	enabled
asPathDbSz	0
asn	65501
attribDbSz	100
childAction	
createTs	2015-09-15T17:39:50.129+00:00
ctrl	fastExtFallover
dn	sys/bgp/inst
lcOwn	local
memAlert	normal
modTs	2015-09-15T17:39:48.635+00:00
monPolDn	uni/fabric/monfab-default
name	

bgpDom	
always	disabled
bestPathIntvl	300
bgpCfgFailedBmp	
bgpCfgFailedTs	00:00:00:00.000
bgpCfgState	0
childAction	
clusterId	unspecified
dn	sys/bgp/inst/dom-default
firstPeerUpTs	2015-08-19T20:30:47.037+00:00
holdIntvl	180
kaIntvl	60
lcOwn	local
maxAsLimit	0
modTs	2015-09-15T17:39:51.648+00:00
mode	fabric
monPolDn	uni/fabric/monfab-default
name	default

Telemetry Sources - NXAPI

NX-OS Release 7.0(3)I6(1)

Check whether the show command has NX-API support:

show <command> | json or show <command> | json pretty

- switch(conf-tm-sensor# data-source NX-API
- switch(conf-tm-sensor)# path "show system resources" depth 0
- switch(conf-tm-sensor)# path "show version" depth 0
- switch(conf-tm-sensor)# path "show policy-map vlan" depth 0
- switch(conf-tm-sensor)# path "show ip access-list test" depth 0

Check Support for show command

Refine the show command to filter

Data source = NXAPI

Sensor paths = show commands

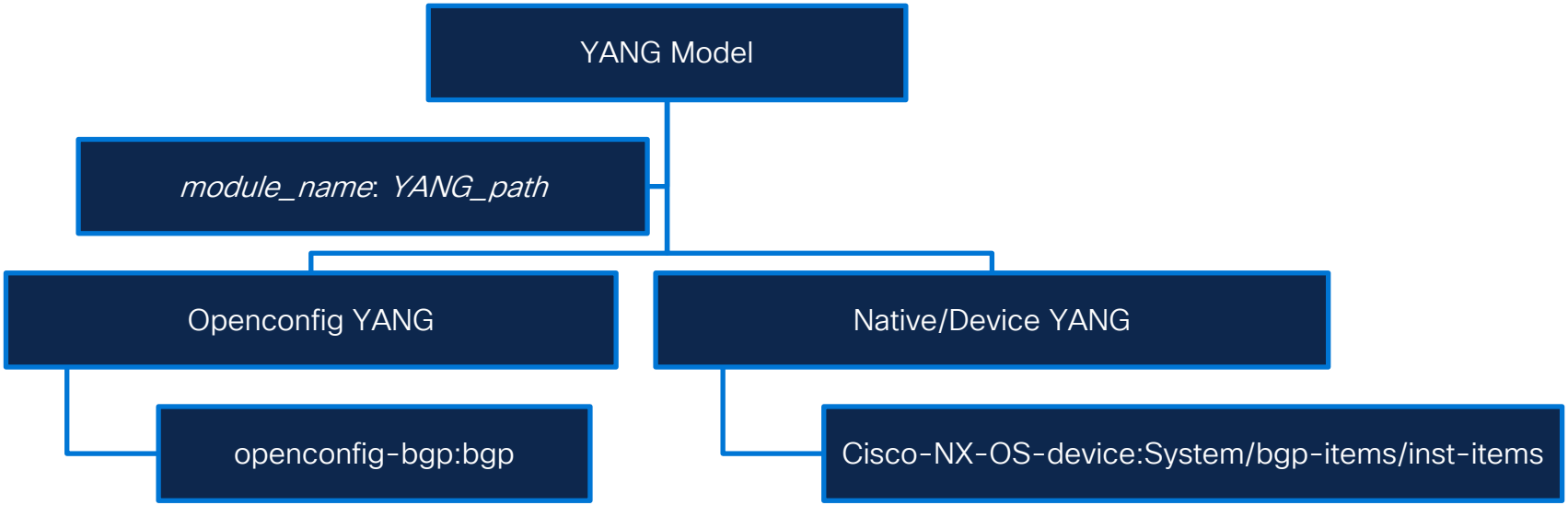
Configure telemetry with a cadence of 5 times the processing time of the respective *show command* to limit CPI usage.

NX-API

```
SPINE-1(config)# show system resources | json
{"load_avg_1min": "0.37", "load_avg_5min": "0.43", "load_avg_15min": "0.47", "pr
ocesses_total": "757", "processes_running": "3", "cpu_state_user": "13.74", "cpu
_state_kernel": "7.12", "cpu_state_idle": "79.13", "TABLE_cpu_usage": {"ROW_cpu_
usage": [{"cpuid": "0", "user": "21.42", "kernel": "4.08", "idle": "74.48"}, {"c
puid": "1", "user": "13.54", "kernel": "8.33", "idle": "78.12"}, {"cpuid": "2",
"user": "7.29", "kernel": "6.25", "idle": "86.45"}, {"cpuid": "3", "user": "12.7
4", "kernel": "9.80", "idle": "77.45"}]}, "memory_usage_total": "16399900", "mem
ory_usage_used": "5458884", "memory_usage_free": "10941016", "current_memory_sta
tus": "OK"}
SPINE-1(config)# sh ip int br | json
{"TABLE_intf": {"ROW_intf": [{"vrf-name-out": "default", "intf-name": "Lo0", "pr
oto-state": "up", "link-state": "up", "admin-state": "up", "iod": "66", "prefix"
: "192.168.1.1", "ip-disabled": "FALSE"}, {"vrf-name-out": "default", "intf-name
": "Lo1", "proto-state": "up", "link-state": "up", "admin-state": "up", "iod": "
67", "prefix": "192.168.2.1", "ip-disabled": "FALSE"}, {"vrf-name-out": "default
", "intf-name": "Eth1/3", "proto-state": "up", "link-state": "up", "admin-state"
: "up", "iod": "7", "prefix": "10.0.10.1", "ip-disabled": "FALSE"}, {"vrf-name-o
ut": "default", "intf-name": "Eth1/4", "proto-state": "up", "link-state": "up",
"admin-state": "up", "iod": "8", "prefix": "10.0.20.1", "ip-disabled": "FALSE"}]
}}
```

Telemetry Sources – YANG Model

NX-OS Release 9.2(1)



Telemetry Sources – YANG Model

NX-OS Release 9.2(1)

```
SPINE-1(config)# show telemetry yang direct-path cisco-nxos-device
1) Cisco-NX-OS-device:System/lldp-items
2) Cisco-NX-OS-device:System/acl-items
3) Cisco-NX-OS-device:System/mac-items
4) Cisco-NX-OS-device:System/intf-items
5) Cisco-NX-OS-device:System/procsys-items/sysload-items
6) Cisco-NX-OS-device:System/ospf-items
7) Cisco-NX-OS-device:System/procsys-items
8) Cisco-NX-OS-device:System/ipqos-items/queuing-items/policy-items/out-items
9) Cisco-NX-OS-device:System/mac-items/static-items
10) Cisco-NX-OS-device:System/ch-items
11) Cisco-NX-OS-device:System/cdp-items
12) Cisco-NX-OS-device:System/bd-items
13) Cisco-NX-OS-device:System/eps-items
14) Cisco-NX-OS-device:System/ipv6-items
SPINE-1(config)#
```

Telemetry Sources – YANG Model

NX-OS Release 9.2(1)

```
SPINE-1# show telemetry control database sensor-groups
Sensor Group Database size = 2
------------------------------------------------------------------------------
--------------------
Row ID     Sensor Group ID  Sensor Group type  Sampling interval(ms)  Linked sub
scriptions  SubID
------------------------------------------------------------------------------
--------------------
1          1                Timer   /YANG      5000      /Running      1
           1
Collection Time in ms (Cur/Min/Max): 20/19/71
Encoding Time in ms (Cur/Min/Max): 2/1/5
Transport Time in ms (Cur/Min/Max): 0/0/0
Streaming Time in ms (Cur/Min/Max): 22/20/8273

Collection Statistics:
  collection_id_dropped      = 0
  last_collection_id_dropped = 0
  drop_count                 = 0

2          2                Timer   /YANG      5000      /Running      1
             1
Collection Time in ms (Cur/Min/Max): 1115/1087/11035
Encoding Time in ms (Cur/Min/Max): 53/53/169
Transport Time in ms (Cur/Min/Max): 1/1/2
Streaming Time in ms (Cur/Min/Max): 1169/1152/11206
```

```
telemetry
  destination-group 1
    ip address 192.0.2.1 port 9000 protocol HTTP encoding JSON
  sensor-group 1
    data-source YANG
    path /Cisco-NX-OS-device:System/procsys-items depth unbounded
  sensor-group 2
    data-source YANG
    path /Cisco-NX-OS-device:System/intf-items/phys-items depth unbounded
  subscription 1
    dst-grp 1
    snsr-grp 1 sample-interval 5000
    snsr-grp 2 sample-interval 5000
```

Telemetry Sources – YANG Model for Syslogs

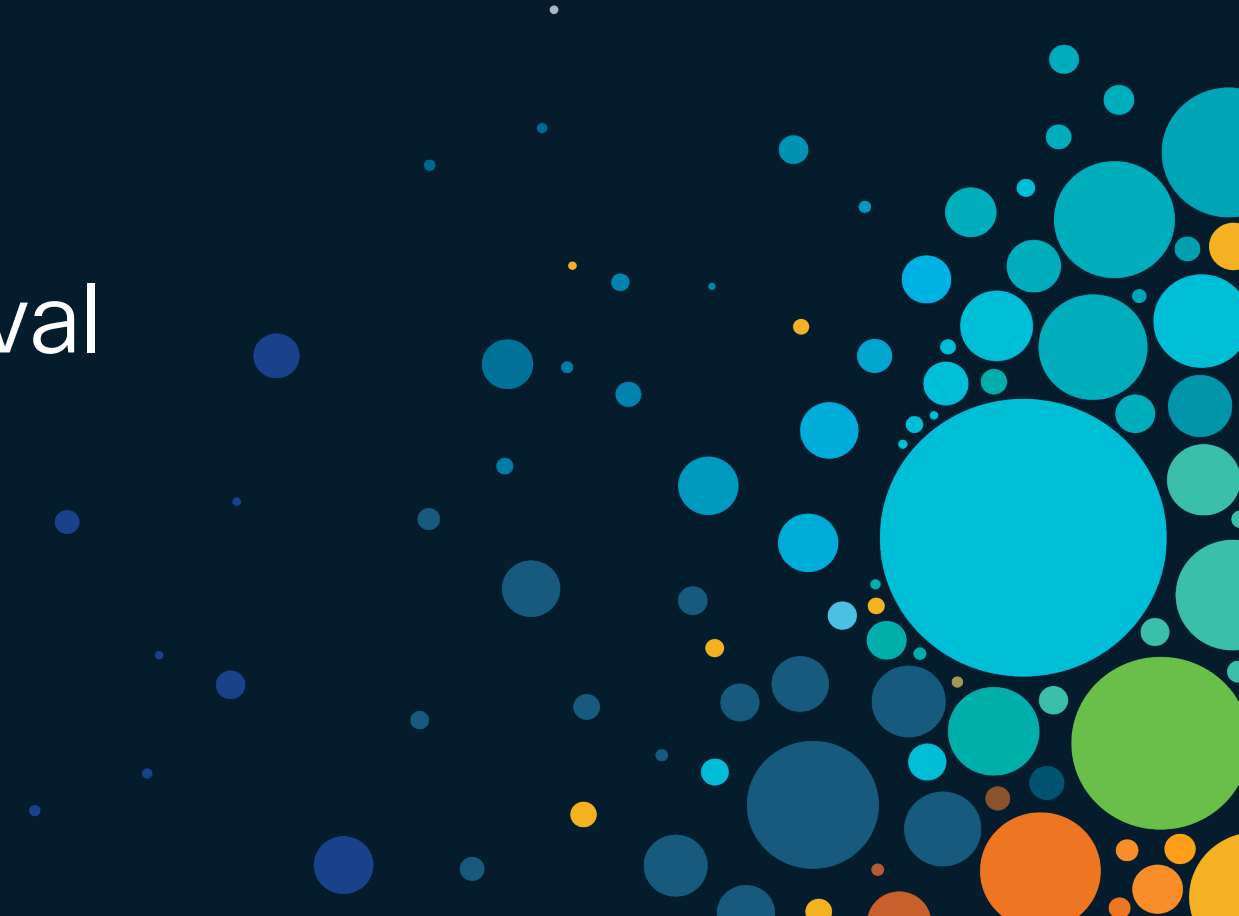
NX-OS Release 9.3(3)

```
telemetry
  destination-group 1
    ip address 192.0.2.1 port 9000 protocol HTTP encoding JSON
  destination-group 6
    ip address 192.0.2.11 port 50001 protocol HTTP encoding JSON
  sensor-group 1
    data-source YANG
    path /Cisco-NX-OS-device:System/procsys-items depth unbounded
  sensor-group 2
    data-source YANG
    path /Cisco-NX-OS-device:System/intf-items/phys-items depth unbounded
  sensor-group 6
    data-source YANG
    path Cisco-NX-OS-Syslog-oper:syslog/messages
  subscription 1
    dst-grp 1
    snsr-grp 1 sample-interval 5000
    snsr-grp 2 sample-interval 5000
  subscription 6
    dst-grp 6
    snsr-grp 6 sample-interval 0
```

Sends event notification for syslog occurred on the switch:

- message-id
- node-name
- time-stamp and time-of-day
- time-zone
- category
- message-name
- severity
- text

Sample Interval Values



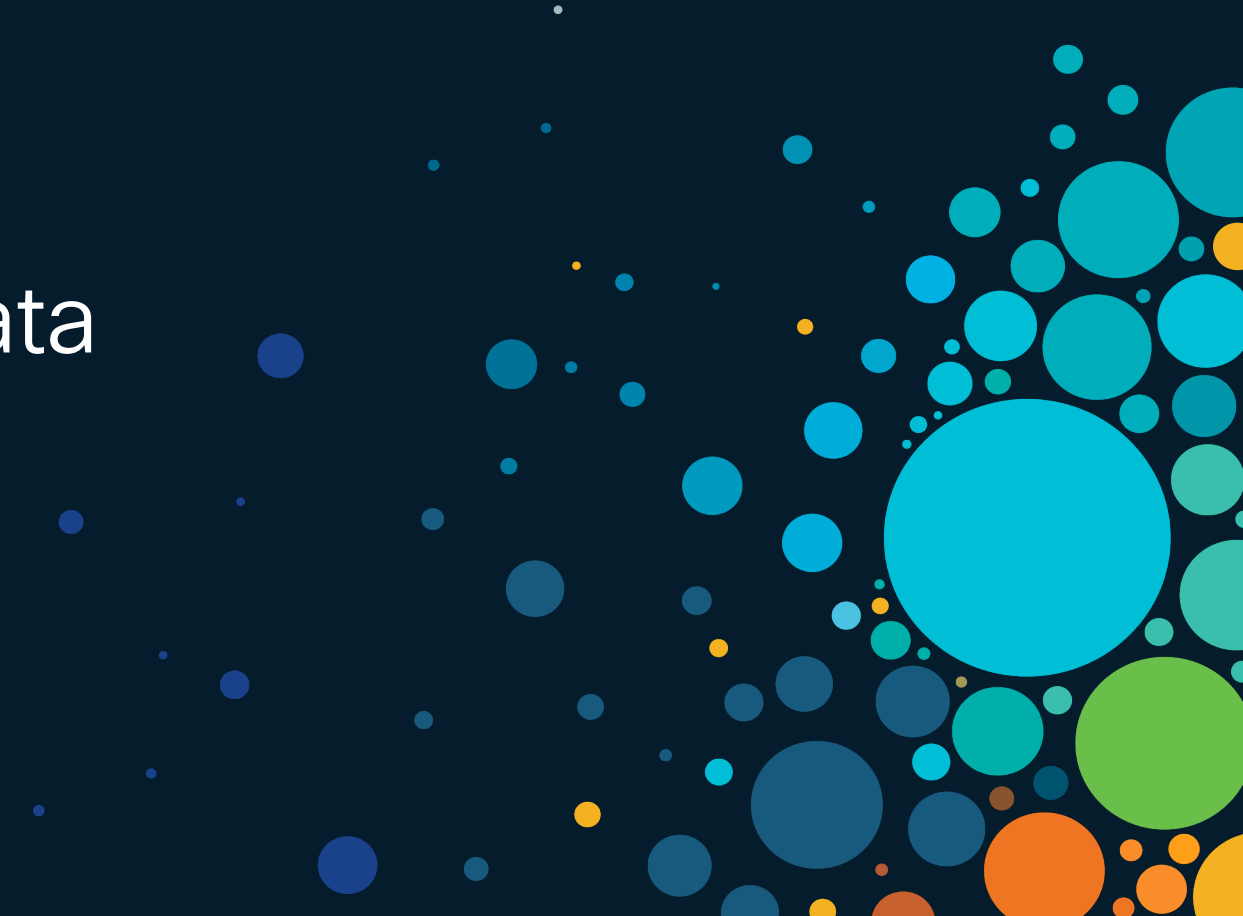
Cadence

Telemetry data can be streamed out in 2 ways

Sample Based – The sample-interval is set in msec.

Event Based – The sample-interval is 0, and telemetry data is event triggered.

Telemetry Data Encoding



Encoding Options

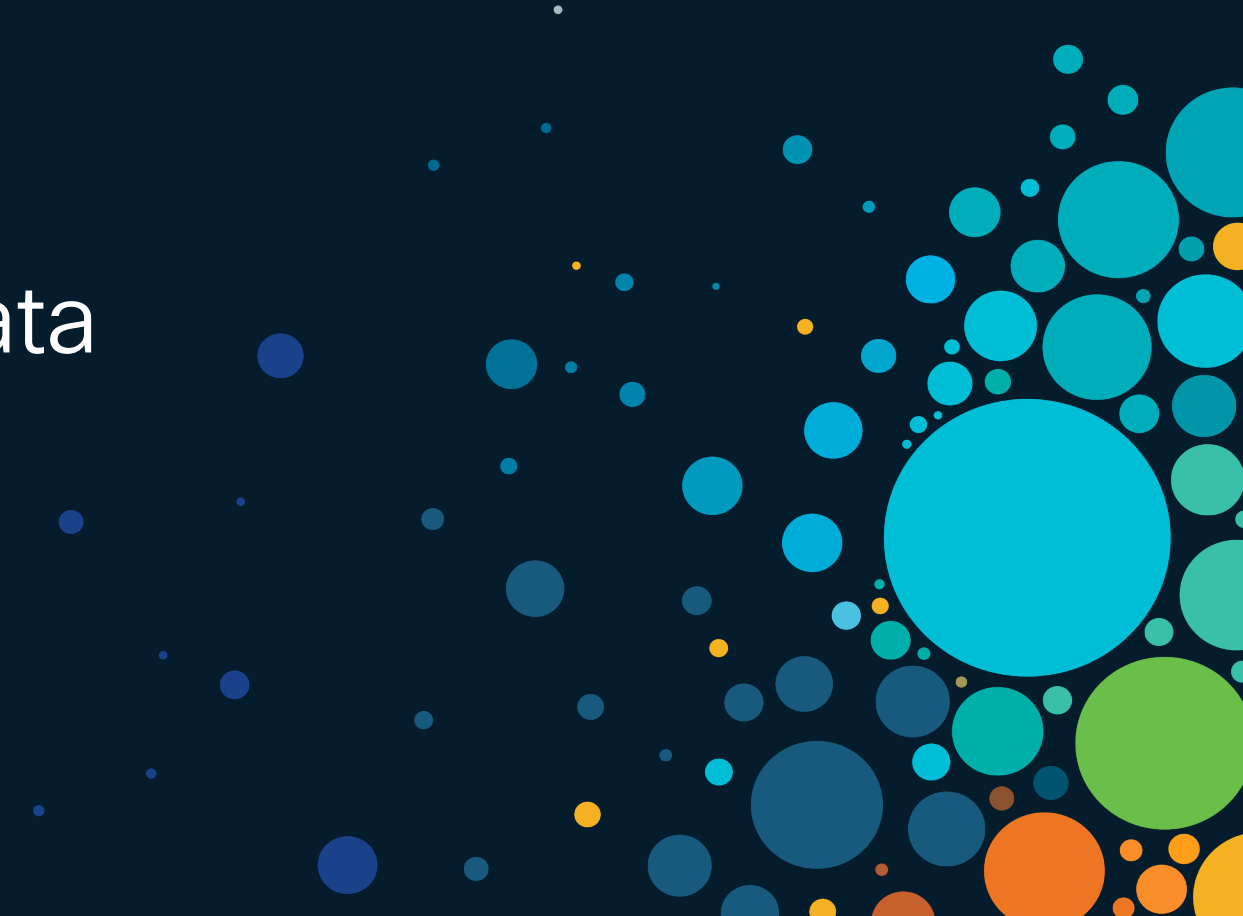
- Google Protocol Buffers – GPB
  - Data Structure – .proto file
  - Compiler – .protoc
  - Machine readable

```
message TelemetryField
{
  uint64         timestamp = 1;
  oneof value_by_type
  {
    bytes          bytes_value = 4;
    string         string_value = 5;
    bool           bool_value = 6;
    uint32         uint32_value = 7;
    uint64         uint64_value = 8;
    sint32         sint32_value = 9;
    sint64         sint64_value = 10;
    double         double_value = 11;
    float          float_value = 12;
  }
}
```

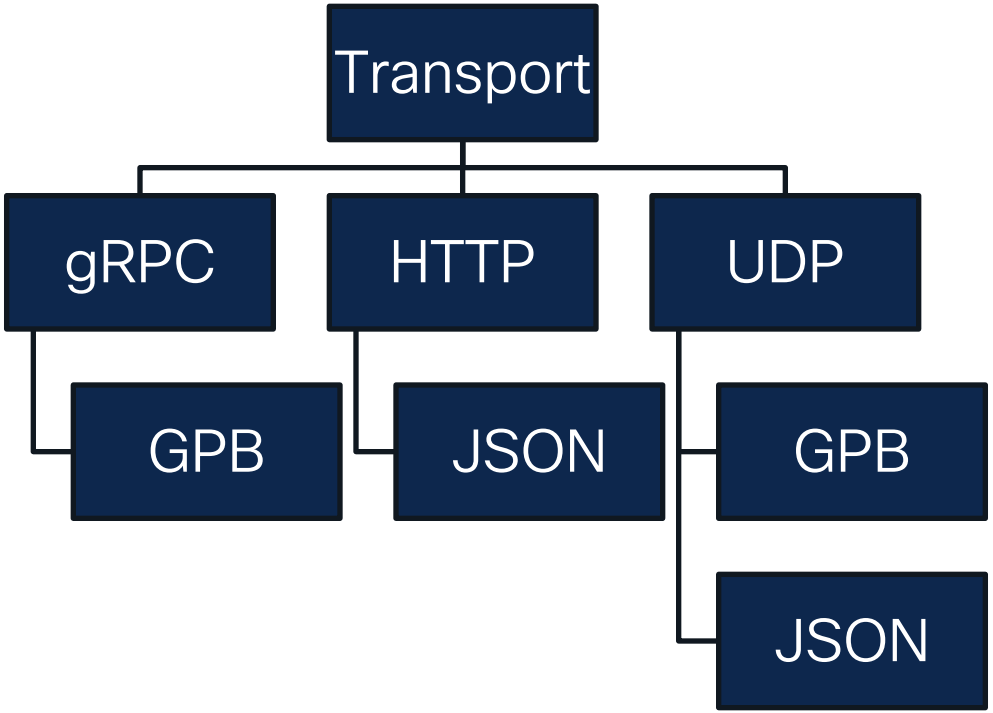
- JSON
  - Structured Data
  - Open standard
  - Human Readable


```
[
  [
    {
      "message-id": 420
    },
    {
      "category": "ETHPORT",
      "group": "ETHPORT",
      "message-name": "IF_UP",
      "node-name": "task-n9k-1",
      "severity": 5,
      "text": "Interface loopback10 is up ",
      "time-of-day": "Dec 3 2019 11:38:51",
      "time-stamp": "1575401931000",
      "time-zone": ""
    }
  ]
]
```

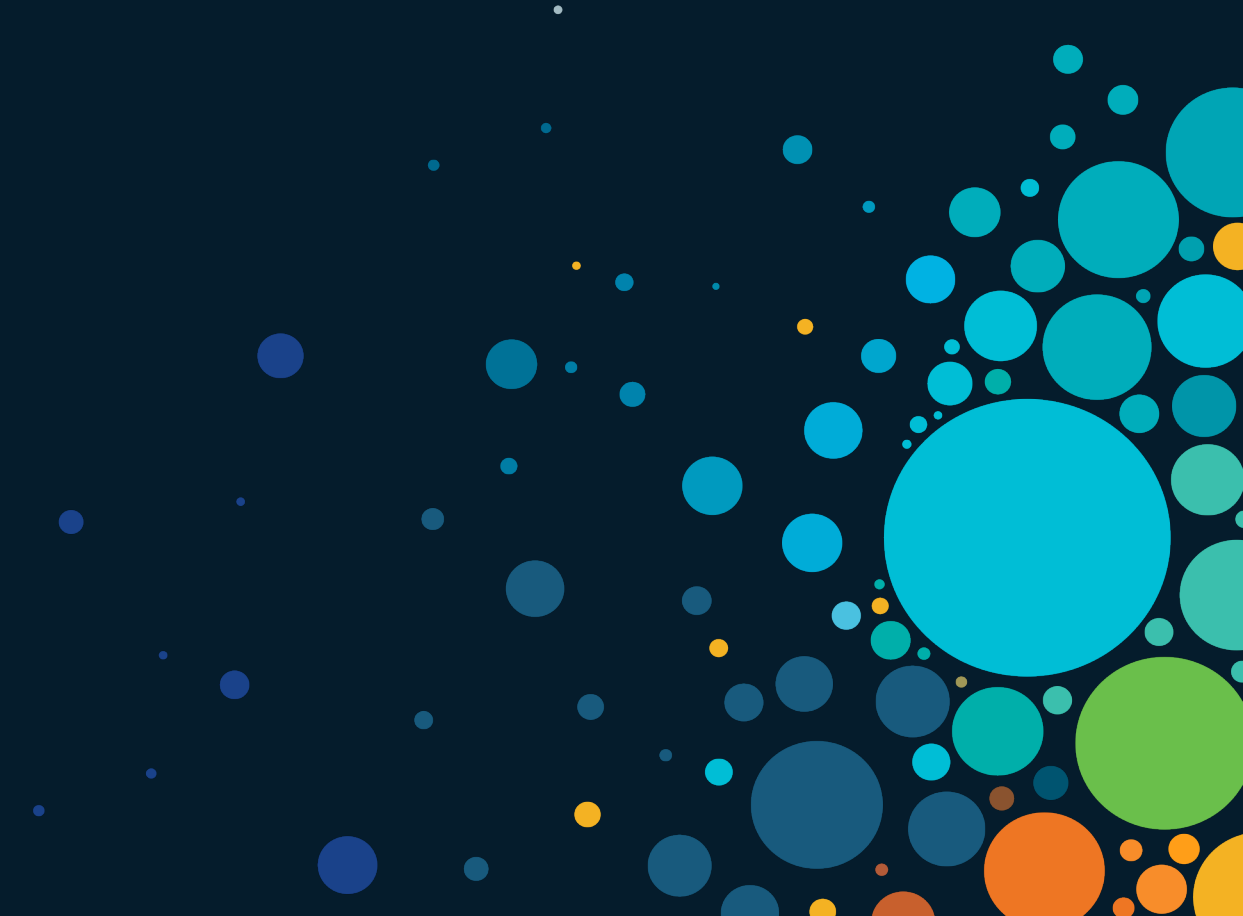
Telemetry Data Transport



Transport Options



Configure Telemetry



STEP-1

The first step is to set the format and destination to which the data is to be sent.

```
telemetry
  destination-profile
    use-vrf management
  destination-group 1
    ip address 10.10.20.50 port 57000 protocol gRPC encoding GPB
```

STEP-2

The second step is to configure the data that is to be collected as part of the sensor group.

```
telemetry
  destination-profile
    use-vrf management
  destination-group 1
    ip address 10.10.20.50 port 57000 protocol gRPC encoding GPB
    certificate /bootflash/telegraf.crt telegraf
  sensor-group 1
    data-source DME
    path sys/ch depth unbounded
  sensor-group 2
    data-source DME
    path sys/intf depth unbounded
```

STEP-3

The third and final step is set the subscription between the sensor-group and the destination, along with the cadence at which to send the data (in msecs).

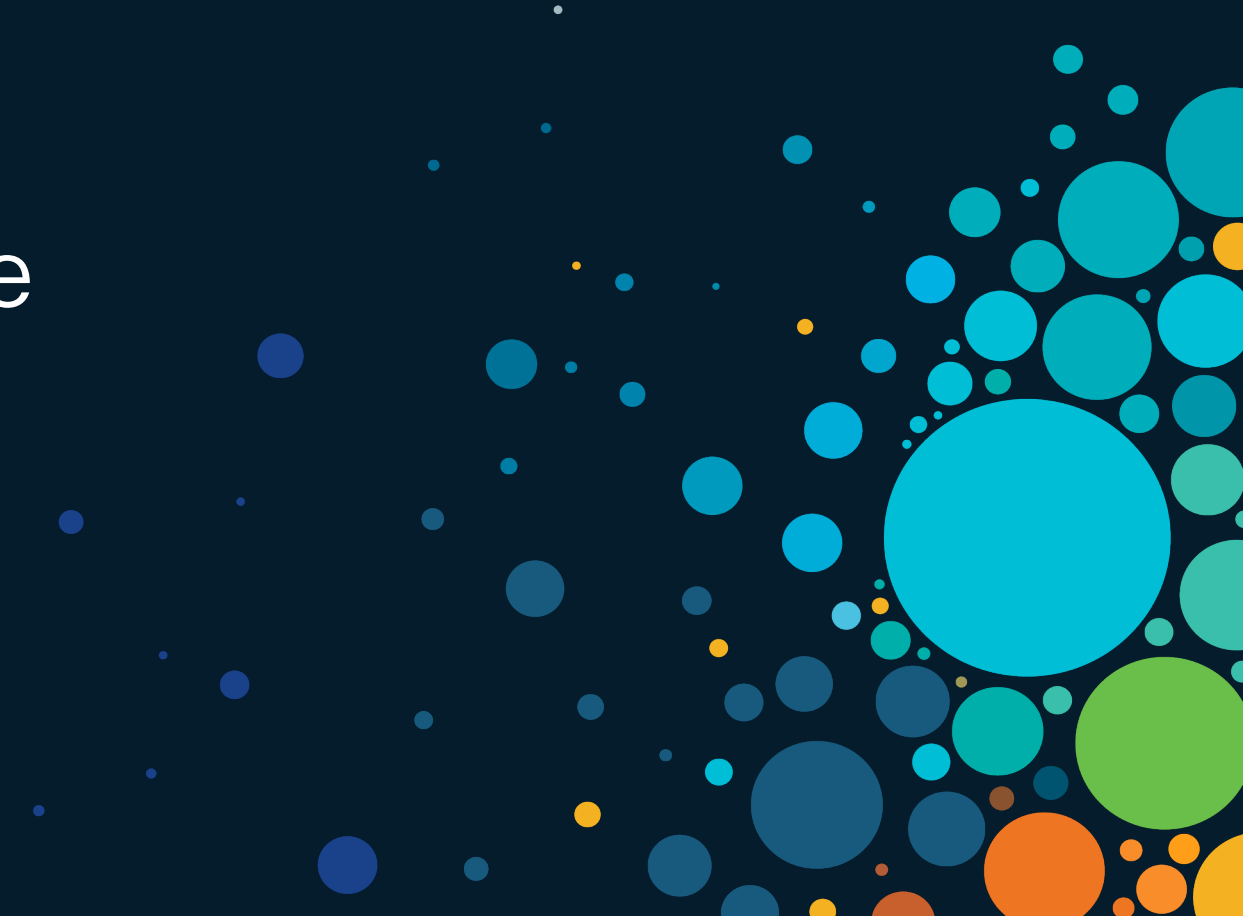
```
subscription 1
  dst-grp 1
  snsr-grp 1 sample-interval 10000
subscription 2
  dst-grp 1
  snsr-grp 2 sample-interval 10000
```

Testing

Use the following show commands to determine the transport status with the receiver. If the connection with the receiver is not successful or the connection times out, the status changes to Disconnected.

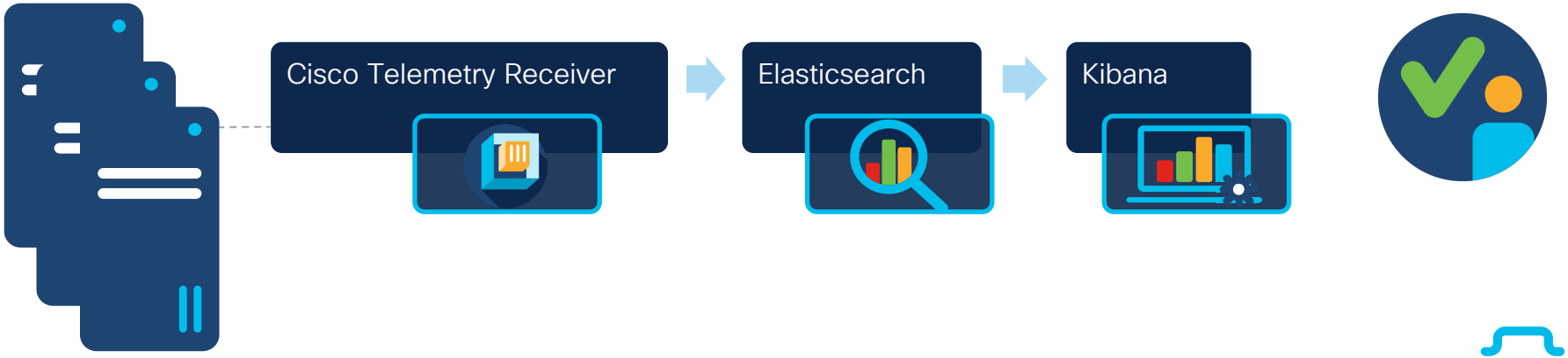
```
switch# show running-config | section telemetry
switch# sh telemetry transport
switch# sh telemetry transport 0 stats
switch# show telemetry control database
switch# show telemetry control stats
switch# show telemetry data collector brief
switch# show telemetry data collector details
switch# show telemetry event collector errors
switch# show telemetry pipeline stats
```

Open-Source Tools for Telemetry



Cisco Telemetry Collector on Dockerhub

docker pull dockercisco/telemetryreceiver

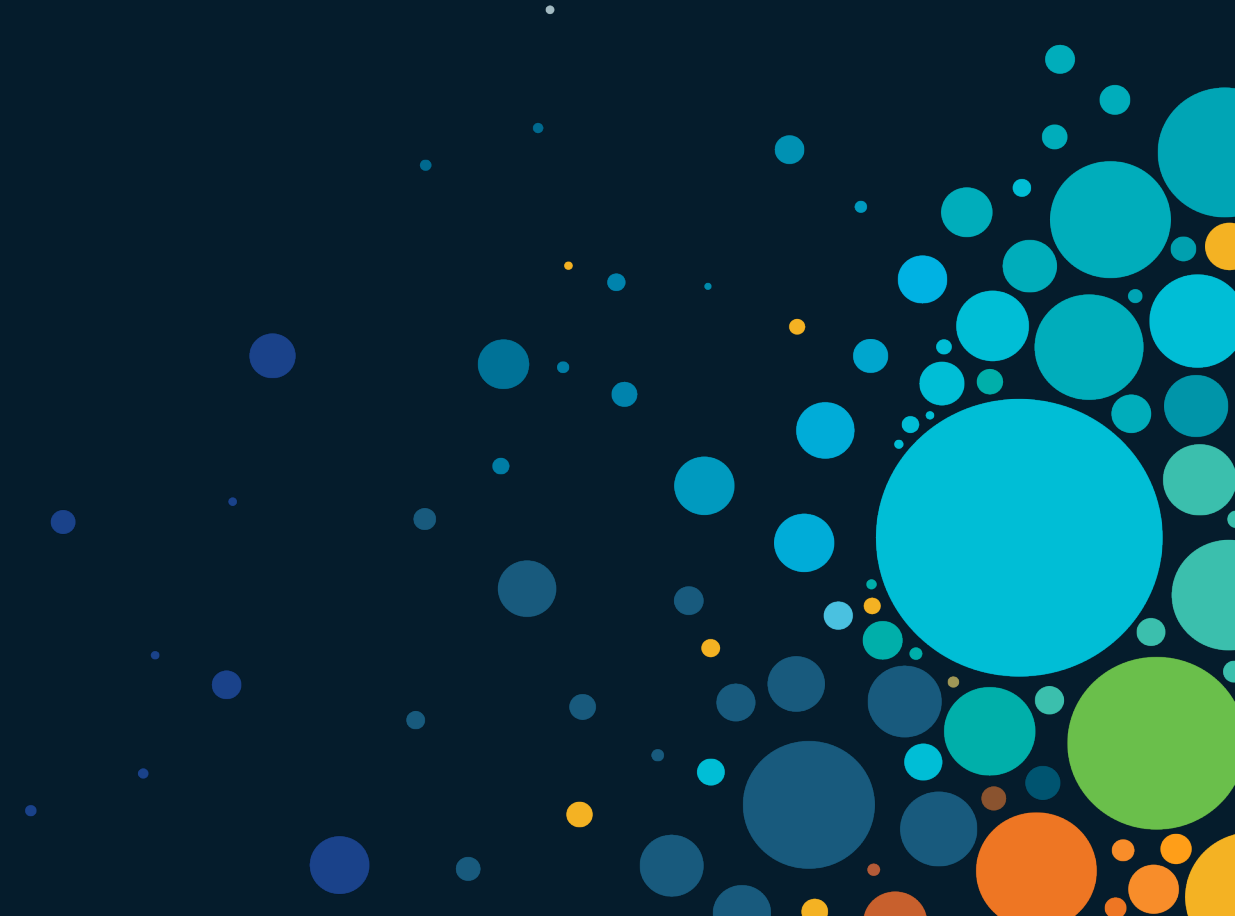


Elasticsearch and Kibana

docker pull dockercisco/elklat

- Log Analysis and Visualization

Demo



Telemetry Guidelines

- Telemetry has the following configuration guidelines and limitations:
- Telemetry is supported in Cisco NX-OS releases starting from 7.0(3)I5(1) for releases that support the data management engine (DME) Native Model.
- Release 7.0(3)I6(1) supports DME data collection, NX-API data sources, Google protocol buffer (GPB) encoding over Google Remote Procedure Call (gRPC) transport, and JSON encoding over HTTP.
- The smallest sending interval (cadence) supported is five seconds for a depth of 0. The minimum cadence values for depth values greater than 0 depends on the size of the data being streamed out. Configuring cadences below the minimum value may result in undesirable system behavior.
- Up to five remote management receivers (destinations) are supported. Configuring more than five remote receivers may result in undesirable system behavior.
- In the event that a telemetry receiver goes down, other receivers will see data flow interrupted. The failed receiver must be restarted. Then start a new connection with the switch by unconfiguring then reconfiguring the failer receiver's IP address under the destination group.
- Telemetry can consume up to 20% of the CPU resource.
- To configure SSL certificate based authentication and the encryption of streamed data, you can provide a self signed SSL certificate with **certificate** *ssl/cert path* **hostname "CN"** command. (NX-OS 7.0(3)I7(1) and later).
- QoS Explicit Congestion Notification (ECN) statistics are supported only on Cisco Nexus 9364C, 9336C-FX, and 93240YC-FX switches.

References

- https://developer.cisco.com/docs/cisco-nexus-3000-and-9000-series-nx-api-rest-sdk-user-guide-and-api-reference-release-9-3x/#!configuring-streaming-statistics-export-ssx/example-configuring-ssx
- https://developer.cisco.com/site/nxapi-dme-model-reference-api/
- https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/7-x/programmability/guide/b_Cisco_Nexus_9000_Series_NX-OS_Programmability_Guide_7x/b_Cisco_Nexus_9000_Series_NX-OS_Programmability_Guide_7x_chapter_011000.html
- https://developer.cisco.com/docs/nx-os/#!telemetry/streaming-telemetry
- https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/93x/progammability/guide/b-cisco-nexus-9000-series-nx-os-programmability-guide-93x/b-cisco-nexus-9000-series-nx-os-programmability-guide-93x_chapter_0101001.html
- https://pubhub.devnetcloud.com/media/dme-docs-10-2-2/docs/appendix/
- https://developer.cisco.com/site/nxapi-dme-model-reference-api/
- https://www.cisco.com/c/en/us/td/docs/dcn/nx-os/nexus9000/102x/programmability/cisco-nexus-9000-series-nx-os-programmability-guide-release-102x/m-n9k-hardware-telemetry-93x.html#Cisco_Concept.dita_46deccca-a83a-4091-8ec2-73aa0246ffb7
- https://www.cisco.com/c/en/us/td/docs/dcn/nx-os/nexus9000/101x/programmability/cisco-nexus-9000-series-nx-os-programmability-guide-release-101x/m-n9k-hardware-telemetry-93x.html

Technical Session Surveys

- Attendees who fill out a minimum of four session surveys and the overall event survey will get Cisco Live branded socks!
- Attendees will also earn 100 points in the Cisco Live Game for every survey completed.
- These points help you get on the leaderboard and increase your chances of winning daily and grand prizes.

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer 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

cisco Live!

Thank you

#CiscoLive