



I M A G I N E

A horizontal sequence of seven letters, each composed of a different set of colored dots and shapes. 'I' is blue and teal. 'M' is dark blue with a dotted pattern. 'A' is green with a triangular shape. 'G' is orange with a hexagonal pattern. 'I' is red with a dotted pattern. 'N' is dark green with a diagonal white bar. 'E' is blue with a dotted pattern.

INTUITIVE

Cisco *live!*
June 10-14, 2018 • Orlando, FL

#CLUS



Leveraging NX-API for Customized Operational Analytics

Dr Tim Miller, Virtual Systems Engineer
DEVNET-2594

Cisco*live!*

#CLUS



INTUITIVE

Cisco Webex Teams



Questions?

Use Cisco Webex Teams (formerly Cisco Spark) 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 Webex Teams or go directly to the team space
- 4 Enter messages/questions in the team space

Webex Teams will be moderated by the speaker until June 18, 2018.

ITM-2016: ITM...
IT took away the complexity so that employees can get on with their core job – allowing more time for engineers to build, salespeople to sell and executives to lead. Join us as we share how digitizing the client experience helped us achieve

Show more ▾

SPEAKER 1
Neil Bamberger
Director - IT
1 event

SPEAKER 2
Julian Wiffen
Manager, IT
1 event

SPARK

Join the Discussion

DOCUMENTS
Session Presentation

cs.co/ciscolivebot#DEVNET-2594

Agenda

- Introduction
- NX-API Overview
- Metrics and Monitoring
- Hands-on
- Conclusion

The Remainder of our 45 Minutes

- Who You Are
 - Command Line Fighter Pilot
 - Know your network
 - Have your favorite set of metrics
- What You'll Learn
 - Translate CLI to Python programming
 - Plethora of monitoring tools
 - Parsing and graphing the metrics

Automation via SSH can be Challenging

- SECURITY!!!!
- Sending standard CLI commands
- Parsing output
- Usually, TCL-based Expect scripts are used
- Complex regular expressions to parse the output
- Entire process is fragile to subtle changes in output structure

Parsing CLI Output

Vintage style versus Hip style

```
Software
BIOS: version 07.59
NXOS: version 7.0(3)I7(3)
BIOS compile time: 08/26/2016
NXOS image file is: bootflash:///nxos.7.0.3.I7.3.bin
NXOS compile time: 2/12/2018 13:00:00 [02/12/2018 19:13:48]

Hardware
cisco Nexus9000 C9372PX chassis
Intel(R) Core(TM) i3- CPU @ 2.50GHz with 16400992 kB of memory.
Processor Board ID SAL18516SA8

Device name: spine-1
bootflash: 51496280 kB
Kernel uptime is 0 day(s), 0 hour(s), 5 minute(s), 17 second(s)
```

```
{
  "bios_ver_str": "07.59",
  "kickstart_ver_str": "7.0(3)I7(3)",
  "bios_cmpl_time": "08/26/2016",
  "kick_file_name": "bootflash:///nxos.7.0.3.I7.3.bin",
  "kick_cmpl_time": "2/12/2018 13:00:00",
  "kick_tmstmp": "02/12/2018 19:13:48",
  "chassis_id": "Nexus9000 C9372PX chassis",
  "cpu_name": "Intel(R) Core(TM) i3- CPU @ 2.50GHz",
  "memory": "16400992",
  "mem_type": "kB",
  "proc_board_id": "SAL18516SA8",
  "host_name": "spine-1",
  "bootflash_size": "51496280",
  "kern_uptm_days": "0",
  "kern_uptm_hrs": "0",
  "kern_uptm_mins": "5",
  "kern_uptm_secs": "31",
}
```

Parsing CLI Output

Vintage style versus Hip style

```
(server) $ ssh admin@switch "show version" > output.txt  
Password:  
(server) $ awk '/BIOS:/ { print $3; }' output.txt  
07.59
```

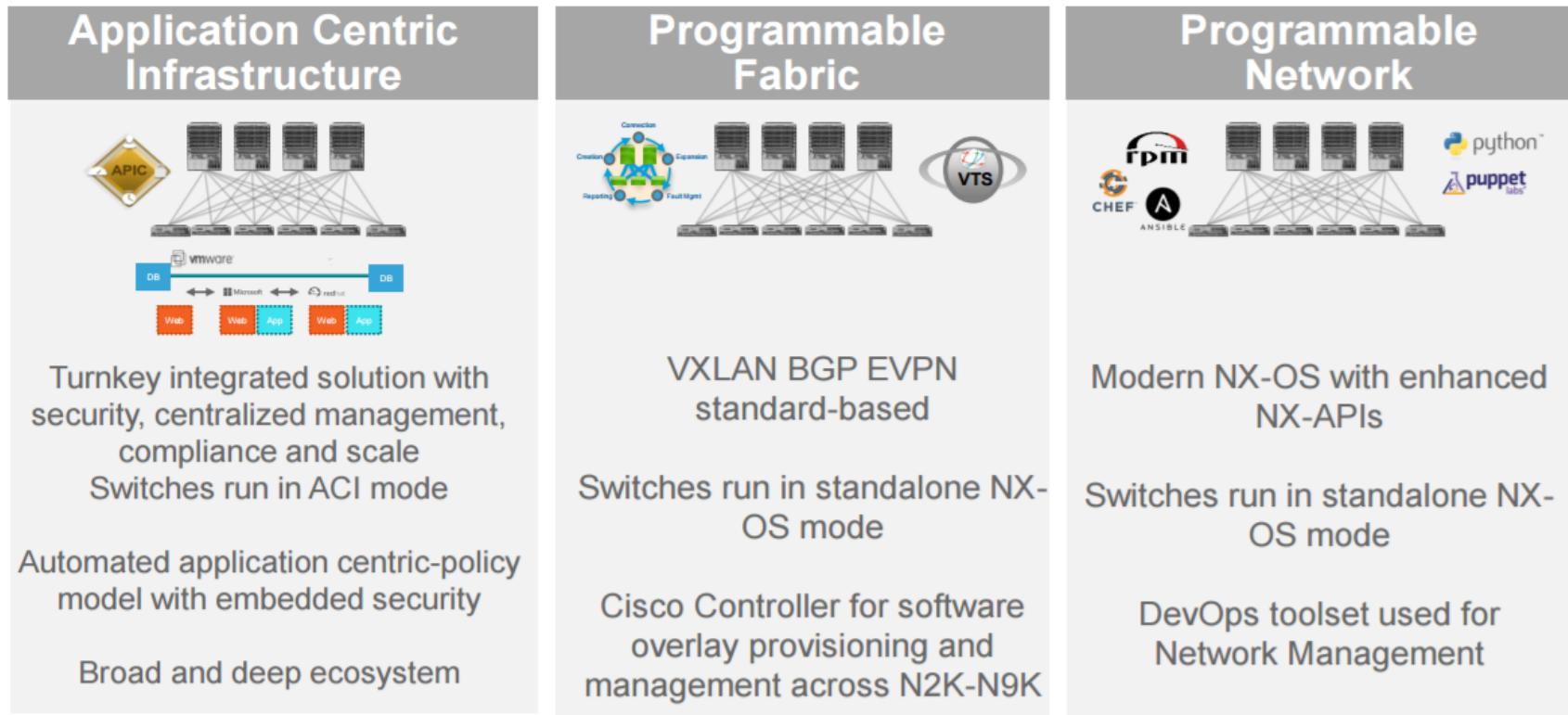
Vintage

Hip

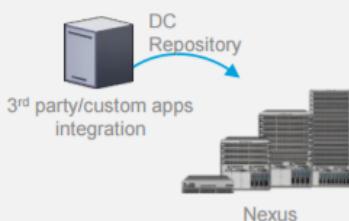
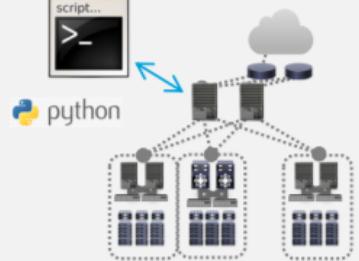
```
print(output["bios_ver_str"])
```

Cisco Data Center Networks:

Providing Choice in Automation and Programmability



Open NX-OS Provides

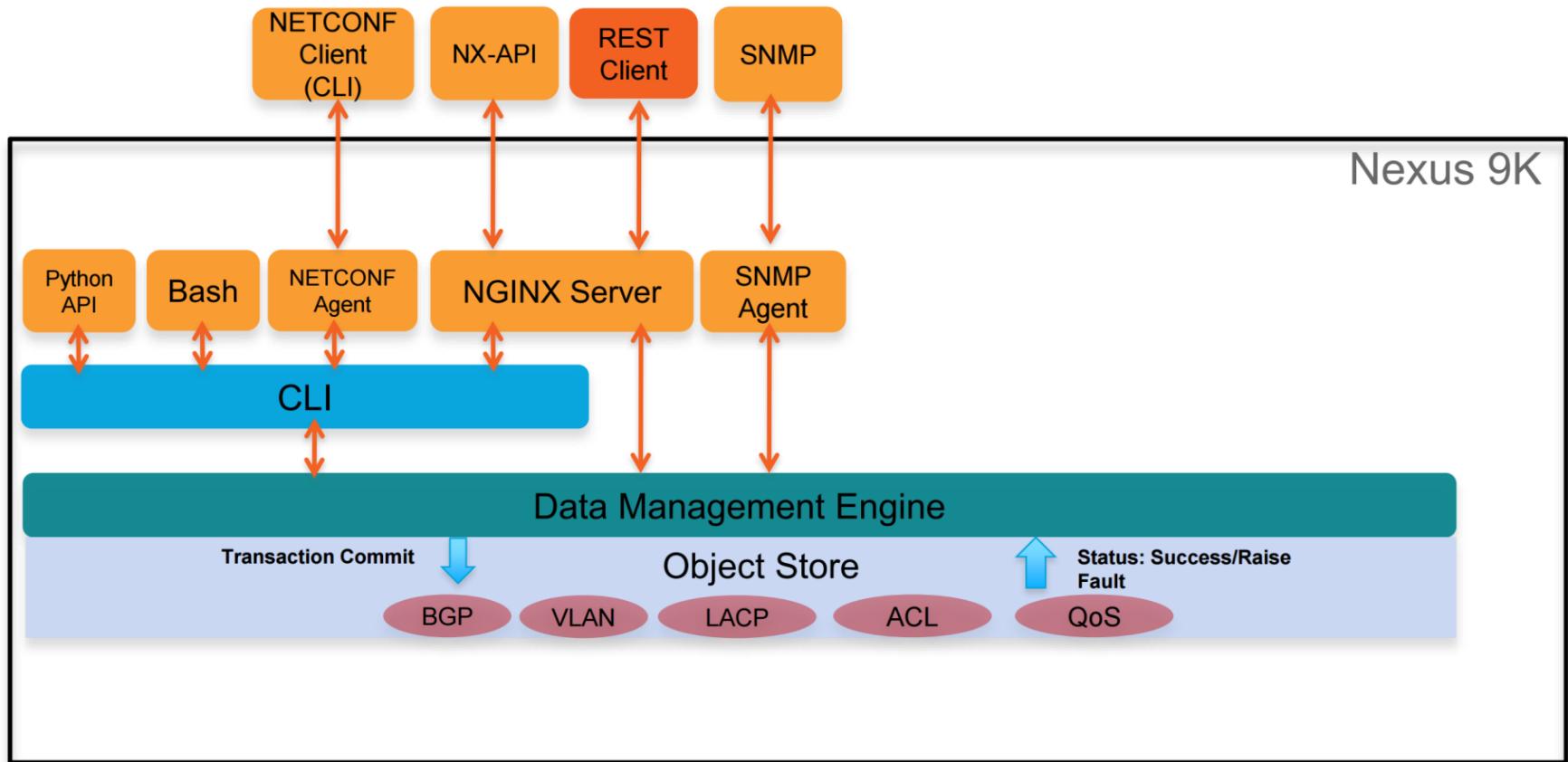
3 rd Party Linux Applications	Linux Tools	Programmable Open APIs	3 rd Party DevOps Automation Tools
 <p>Use 3rd party applications using secure LXC containers</p>		 <p>Object-based, model driven APIs (RESTful XML/JSON)</p>	 <p>Leverage same software tools and expertise across different IT departments</p>

NX-API Overview

Cisco *live!*



NX-API CLI vs NX-API REST



SSH CLI Example



“show version”

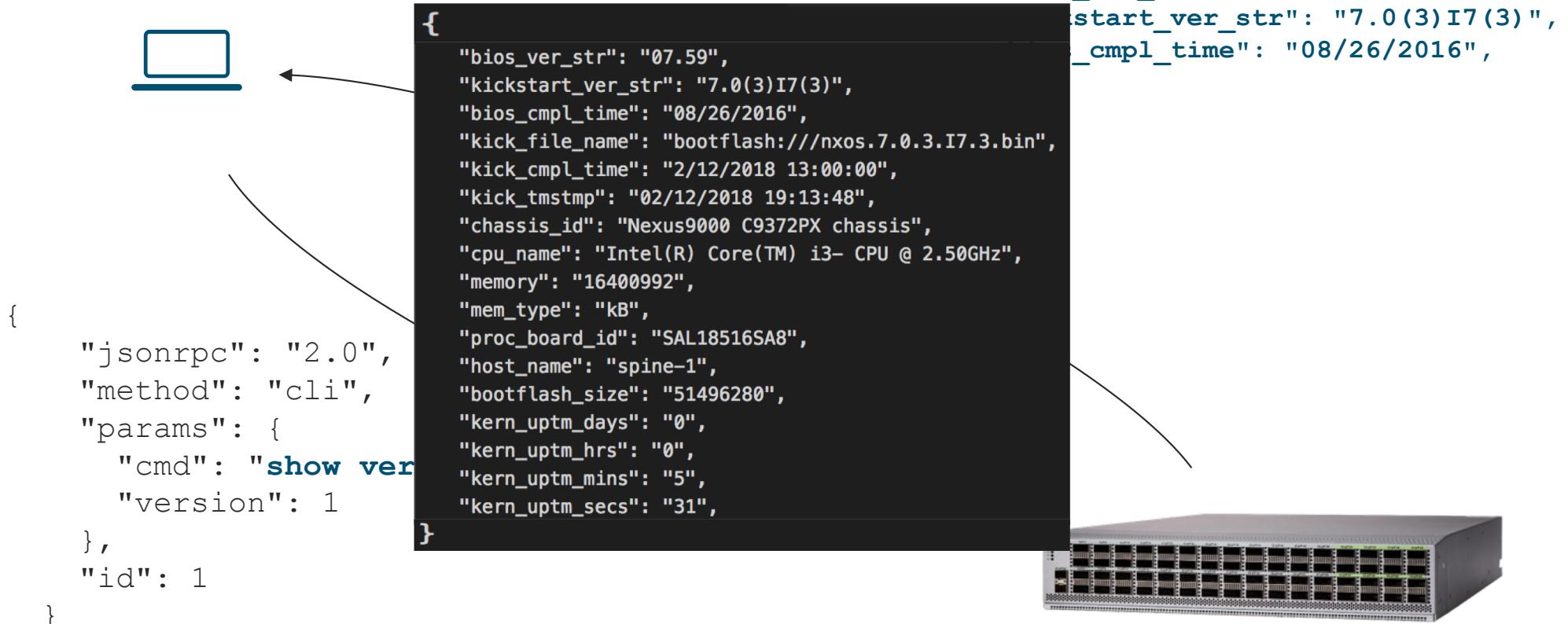
```
Software
  BIOS: version 07.59
  NXOS: version 7.0(3)I7(3)
  BIOS compile time: 08/26/2016
  NXOS image file is: bootflash:///nxos.7.0.3.I7.3.bin
  NXOS compile time: 2/12/2018 13:00:00 [02/12/2018 19:13:48]

Hardware
cisco Nexus9000 C9372PX chassis
Intel(R) Core(TM) i3- CPU @ 2.50GHz with 16400992 kB of memory.
Processor Board ID SAL18516SA8

Device name: spine-1
bootflash: 51496280 kB
Kernel uptime is 0 day(s), 0 hour(s), 5 minute(s), 17 second(s)
```



NX-API CLI Example



Cisco live!

#CLUS

DEVNET-2594

© 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public

14

← → C ⓘ 10.10.10.52 ⭐

NX-API Developer Sandbox

Quick Start Logout ↗

show version

Message format: ⓘ

json-rpc **xml** json nx-api rest
nx yang

Command type: ⓘ

cli **cli_ascii**

POST Reset

REQUEST:

```
[
  {
    "jsonrpc": "2.0",
    "method": "cli",
    "params": {
      "cmd": "show version",
      "version": 1
    },
    "id": 1
  }
]
```

Copy Python

RESPONSE:

```
{
  "jsonrpc": "2.0",
  "result": {
    "body": {
      "header_str": "Cisco Nexus Operating System (NX-OS) Software\nnTAC supp",
      "bios_ver_str": "",
      "kickstart_ver_str": "7.0(3)I5(1)",
      "bios_cmpl_time": "",
      "kick_file_name": "bootflash:///nxos.7.0.3.I5.1.bin",
      "kick_cmpl_time": " 10/29/2016 6:00:00",
      "kick_tmstmp": "10/29/2016 13:46:41",
      "chassis_id": "NX-0Sv Chassis",
      "cpu_name": "Intel(R) Xeon(R) CPU E5-4640 v2 @ 2.20GHz",
      "memory": 8165348,
      "mem_type": "kB",
      "proc_board_id": "99FJDMFVPY5",
      "mem_size": 8165348
    }
  }
}
```

Copy

The screenshot shows the NX-API Developer Sandbox interface. At the top, there's a header with the Cisco logo and the title 'NX-API Developer Sandbox'. Below the header, there's a search bar with the IP address '10.10.10.52' and a star icon. To the right of the search bar are 'Quick Start' and 'Logout' buttons. The main area has a large input field containing the command 'show version'. To the right of this input field are two sections: 'Message format:' with options for json-rpc (selected), xml, json, nx-api rest, and nx yang; and 'Command type:' with options for cli (selected) and cli_ascii. Below the input field are 'POST' and 'Reset' buttons. At the bottom, there are two side-by-side boxes. The left box is labeled 'REQUEST:' and contains a JSON-RPC message. The right box is labeled 'RESPONSE:' and contains the resulting JSON object. Both boxes have 'Copy' and 'Python' buttons.

Cisco live!

#CLUS

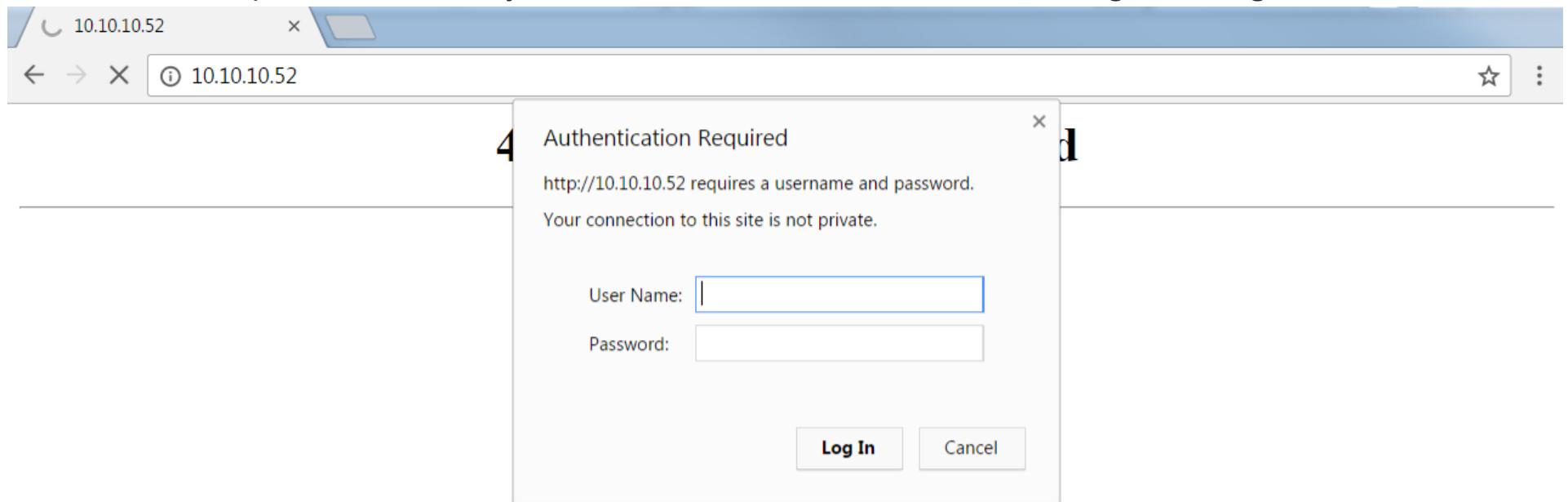
DEVNET-2594

© 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public

15

NX-API CLI is Secure

- Users must have the correct device role to use NX-API CLI.
- For example, a read-only role will not be able to make changes using NX-API CLI.

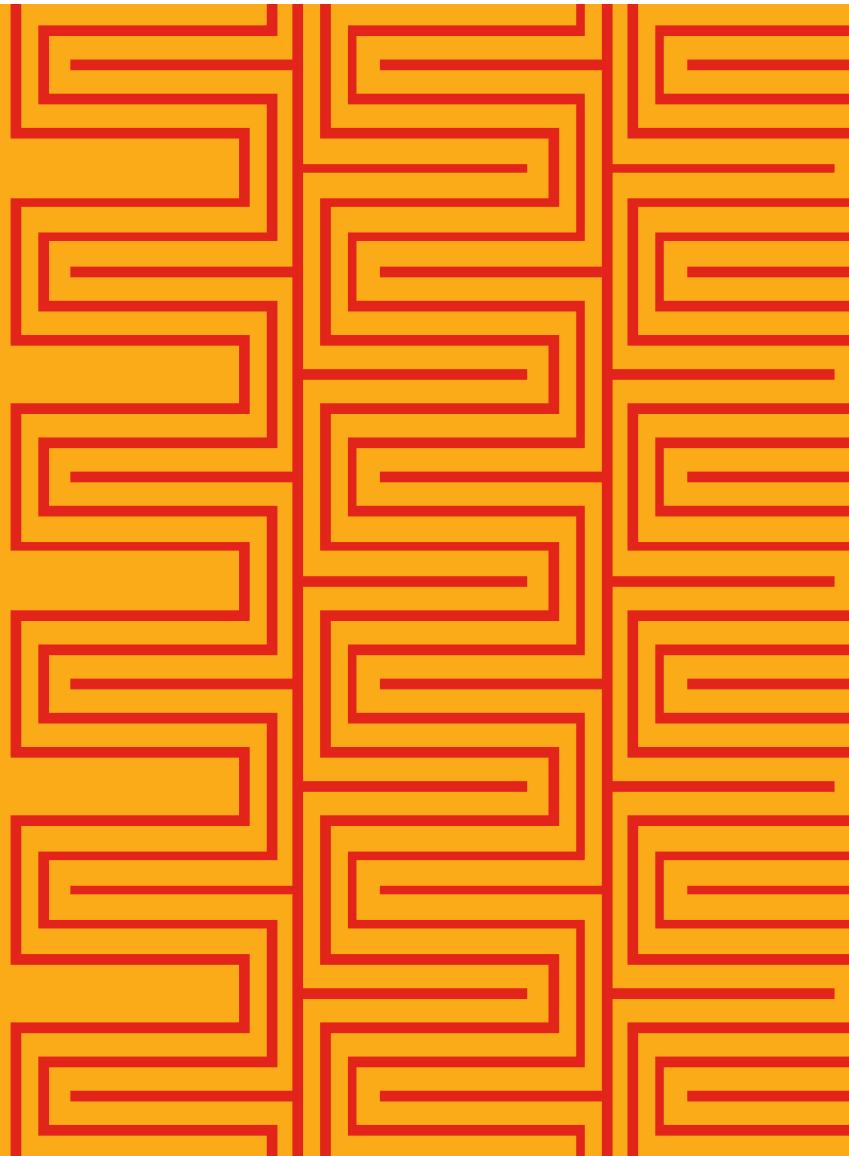


NX-API CLI Has Many Use Cases

- Check versions of multiple switches in one command
- VLAN provisioning
- Poll routing table to watch for flapping routes
- Poll MAC address table for end point tracking
- Collect LLDP/CDP data to build wiring maps
- Couple collection of structured output with database backend for more advanced applications

NX-API CLI Demo!

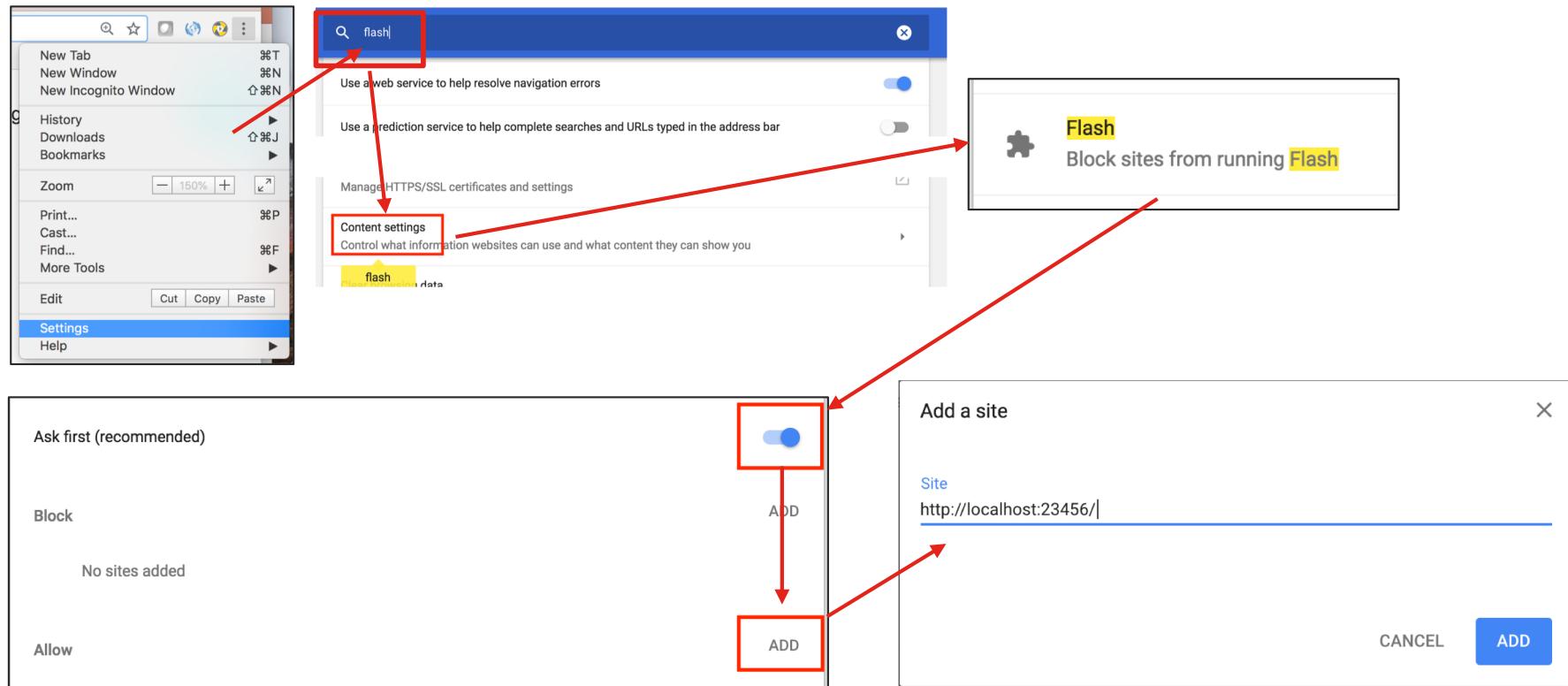
Cisco*live!*



Connecting to Local N9KV Developer Sandbox

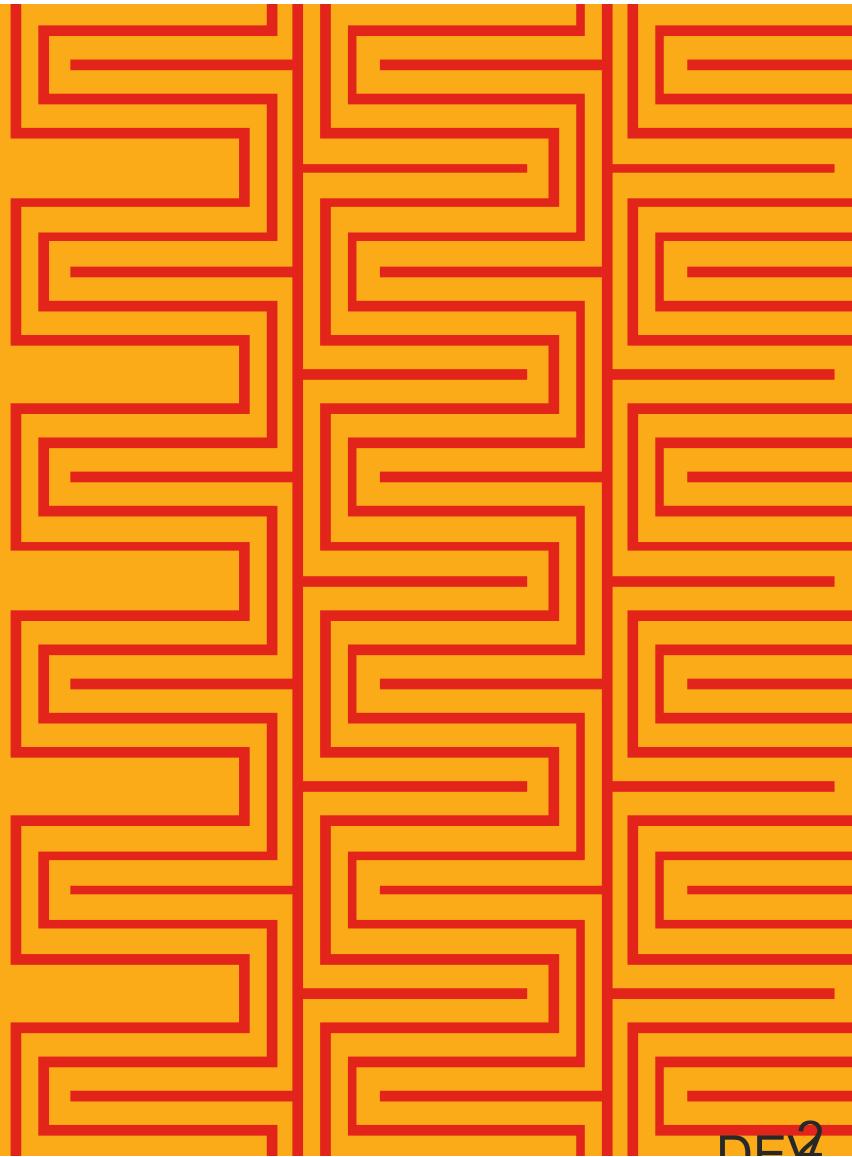
- Developer Sandbox is running on `http://localhost:23456/`
 - Username/Password is `admin/admin`
- May have to permit Flash in Chrome (next slide)
 - When connecting to the Developer Sandbox URL above, you'll get a warning that Flash is needed
 - OR... the “Python” button in the lower left request box does not produce Python text

Allowing Chrome to Use Flash for URL



Fire up NX9000V

Cisco *live!*



DEV2

Start Your Vagrant Box

```
$ mkdir -p ${HOME}/workspace/DEVNET-2594-CLUS18/n9kv  
$ cp nxosv-final.7.0.3.I7.3.box ${HOME}/workspace/DEVNET-2594-CLUS18/n9kv  
$ cd ${HOME}/workspace/DEVNET-2594-CLUS18/n9kv  
$ vagrant box add base nxosv-final.7.0.3.I7.3.box  
# Use your favorite editor to edit Vagrantfile (see below)  
$ vagrant up
```

If Vagrantfile does not exist, run **vagrant init** to create one in the **n9kv** directory.

Make sure to uncomment the line below and change port to **23456**:

```
config.vm.network "forwarded_port", guest: 80, host: 8080
```

https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/7-x/nx-osv/configuration/guide/b_NX-OSv_9000/b_NX-OSv_chapter_01.html#task_jhy_dvw_qy

Verify Working Environment

- Bootstrap NXAPI and BOOT setup

```
$ vagrant ssh
```

Command in Mac terminal
REPO_ROOT/n9kv directory

```
Nexus9000v# config terminal
Nexus9000v# feature nxapi
Nexus9000v# boot nxos bootflash:nxos.7.0.3.I7.3.bin
Nexus9000v# end
Nexus9000v# copy run start
```

Commands in NX-OS virtual switch running in Vagrant box

- Developer Sandbox is running on localhost:23456
 - Username/Password is admin/admin

Metrics

Cisco *live!*



Anatomy of Metrics Analytics

- Service to monitor
- Metric Generation
- Metric Collection
- Metric Storage
- Metric Visualization
- Browser to view it all

Metrics Generation

- Time Series Data
 - We are collecting measurements at regularly points in time
 - Name of Metric, Time Stamp, Metric Value, Metric Labels (units, source)
- Different Types of Metrics
 - Gauges
 - Counters
 - Timers (StatsD)
 - Histogram (Prometheus)
- Generators can be kernel level data, real time measurements, or calculated values

Graphite/Carbon/Whisper

Components

- Graphite – web-frontend for dynamic graph generation
- Carbon – daemon to receive time-series data
- Whisper – database format for storing time-series data (RRD-like)

Operation

- Simply feed it 3 values : **metric_path value timestamp**
- Definition of metric_path before use not required
- Set of Graphite functions used to transform/combine data for rendering
- <http://graphite.readthedocs.io/en/latest/overview.html>

Telegraf/InfluxDB

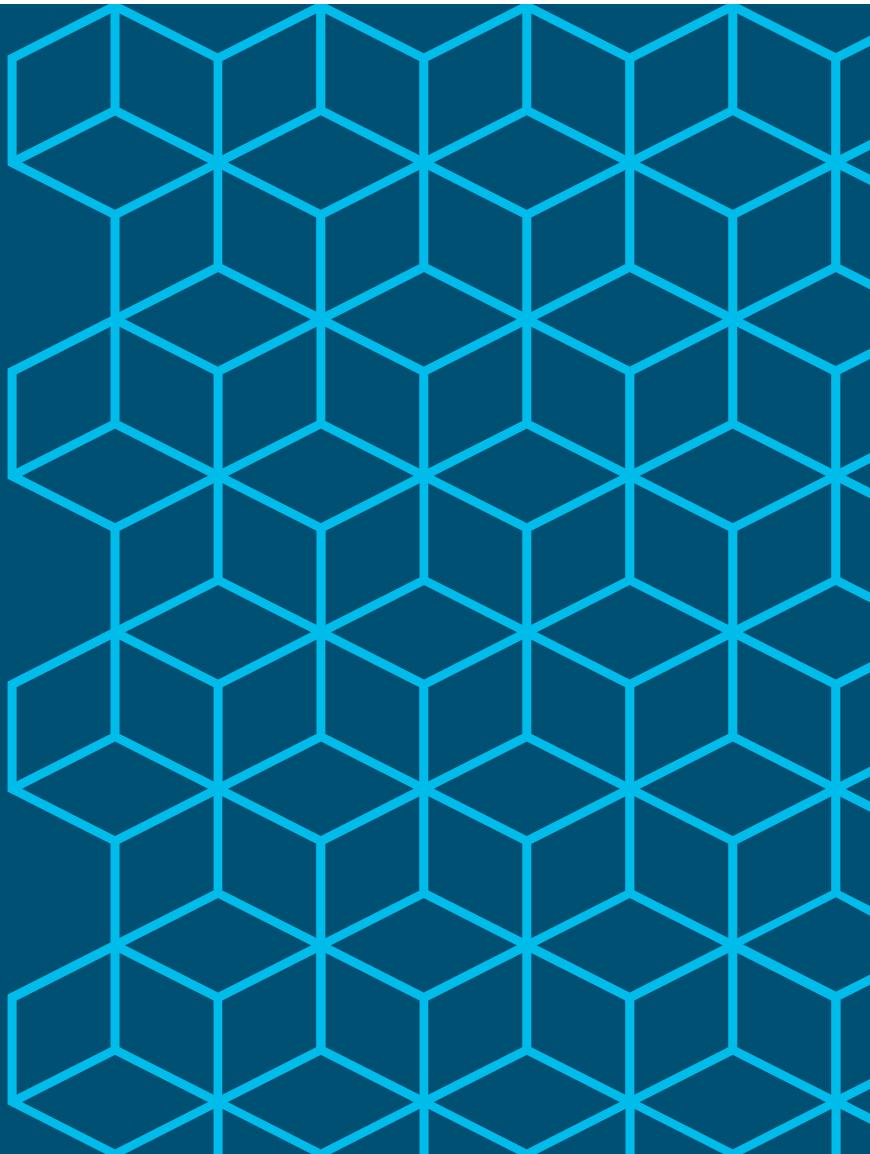
- Telegraf – Metric collection
 - Runs commands that generate values
 - Sends values to various formats/destinations (JSON, Influx, Graphite)
- InfluxDB – Metric storage
 - Optimized for large datastores
 - SQL-like language
 - Retention policies
 - Tags for indexing metrics for fast, efficient queries
- <https://www.influxdata.com/time-series-platform/telegraf/>

Prometheus

- Collection occurs via pull model over HTTP
 - Pushgateway exists for short-lived services or batch jobs
- Supported for service discovery (DNS, K8s, etc.)
- Data model for multi-dimension storage of time series data
 - Metric name, key/value pairs
 - Designed with microservices in mind
- <https://prometheus.io/docs/introduction/overview/>

One Python Script to Rule Them All

Cisco*live!*

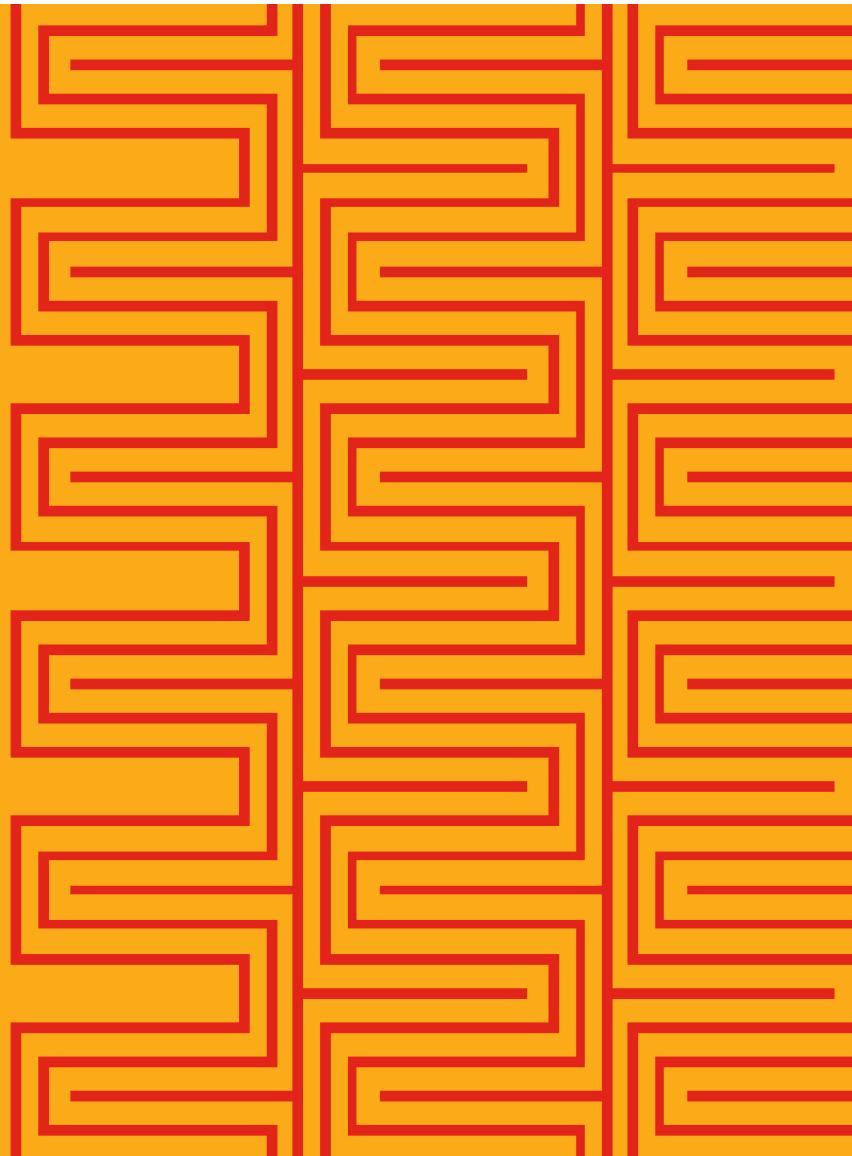


Core Python Script Architecture

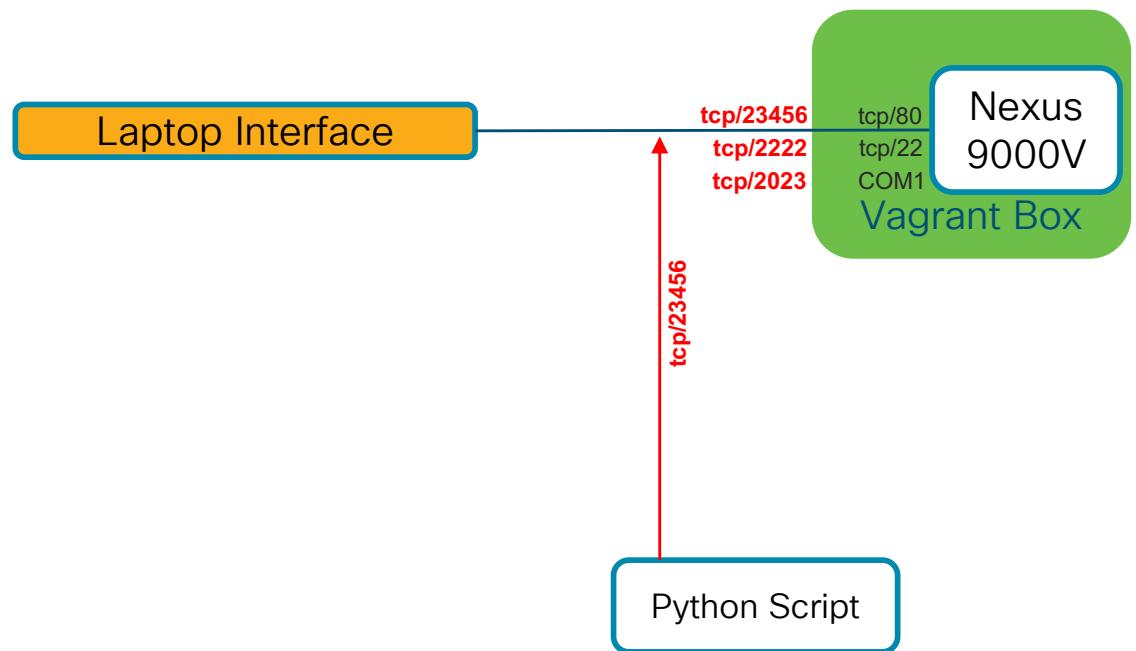
- Form NX-API CLI request
- Connect via HTTP/HTTPS to switch
- Post NX-API CLI data
- Parse NX-API CLI response
- Identify/Calculate Metric
- Transfer that metric to Collector

Stop! Python Time...

Cisco*live!*



Metric Collection Service Connectivity Diagram

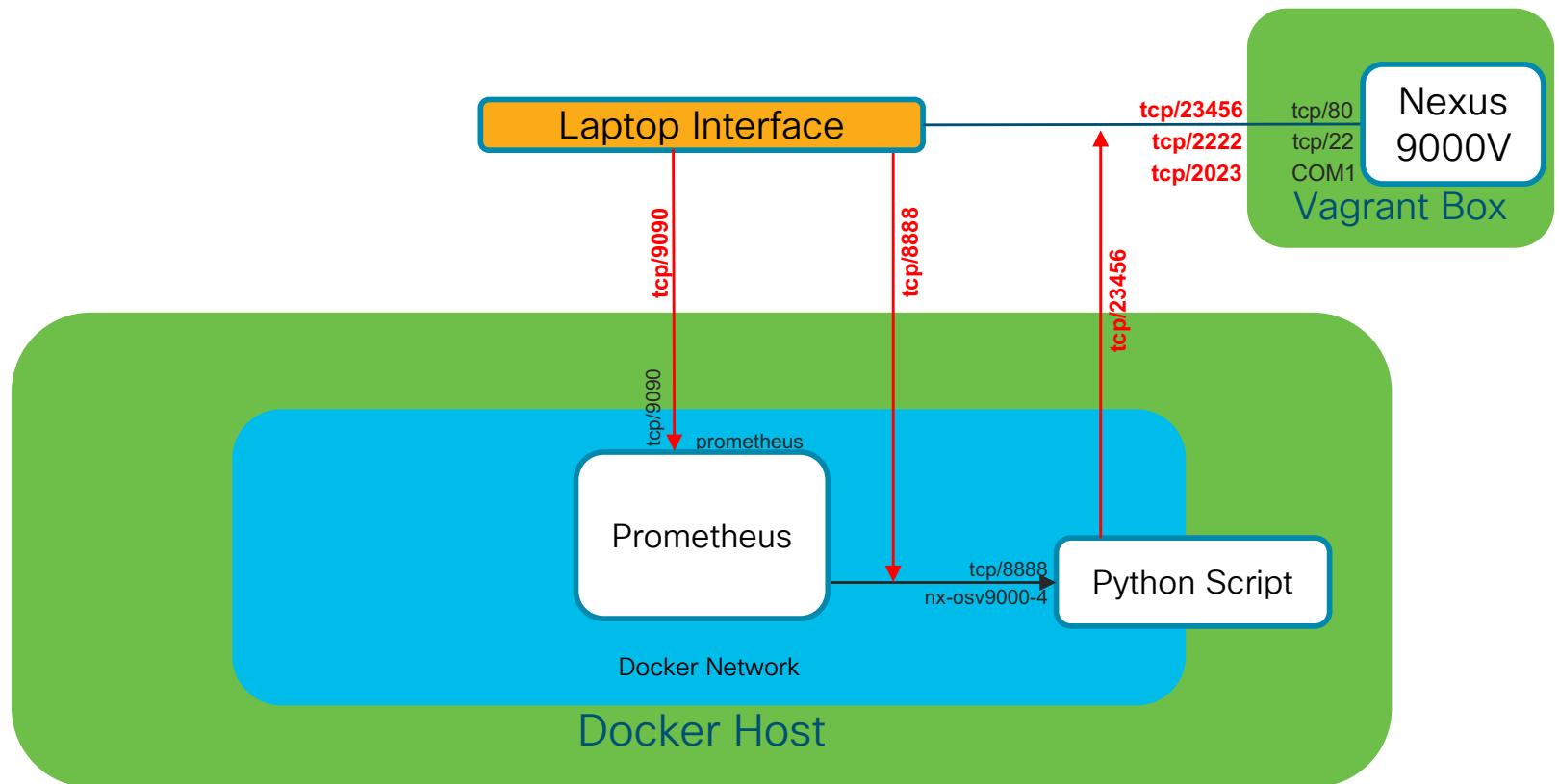


Commands found in README.md

Step-01 – Verify and Test Metric Generation

- Ensure latest copy of code is on your laptop (exit from NX-OS)
 - `(cd ${HOME}/workspace/DEVNET-2594-CLUS; git pull)`
- Enable iCAM features on switch
 - `cd ${HOME}/workspace/DEVNET-2594-CLUS/n9kv`
 - `python setup_nxos.py`
- Run script
 - `cd ../nxapi_cli/step-01; python generate_l2table.py`

Metric Collection Service Connectivity Diagram



Step-02 – Collect metric in Prometheus

- Build Docker image of Collector script

```
docker build -t devnet-2594/publish_l2table:latest -t devnet-2594/publish_l2table:1 .
```

- Create Docker network

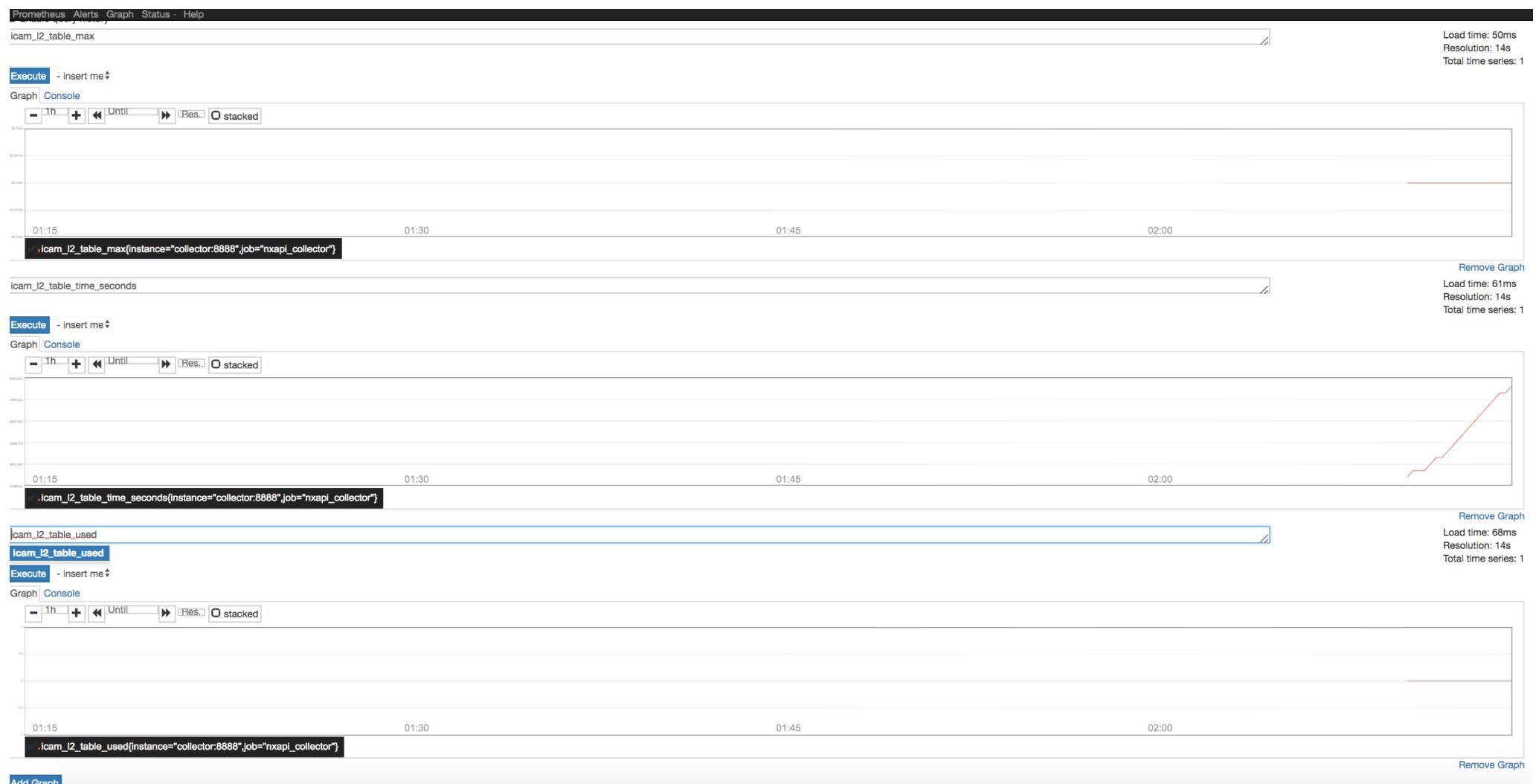
```
docker network create --driver=bridge --subnet=192.168.254.0/24 \
                      --gateway=192.168.254.254 --attachable demo0
```

- Deploy Prometheus container

```
docker run --name prometheus -d --network demo0 -p 127.0.0.1:9090:9090 \
            -v ${PWD}/prometheus.yml:/etc/prometheus/prometheus.yml \
            quay.io/prometheus/prometheus
```

- Deploy Collector container

```
docker run --name collector -d --network demo0 -p 127.0.0.1:8888:8888 \
            devnet-2594/publish_l2table
```



Cisco live!

#CLUS

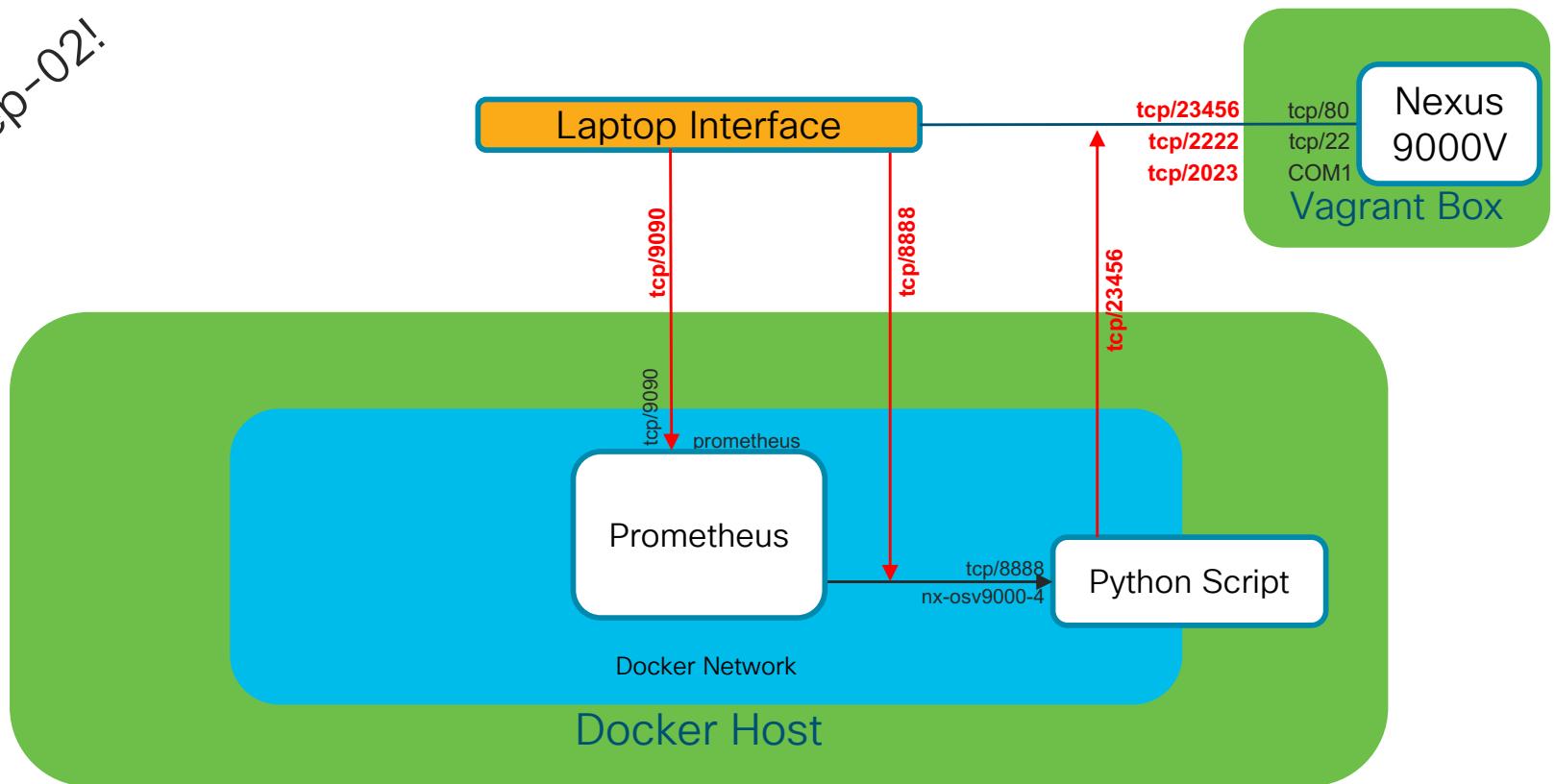
DEVNET-2594

© 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public

37

Metric Collection Service Connectivity Diagram

Same as Step-02!



Step-03 – Expand metrics collected

Refactor Python Code

- Clean up from step-02 (cleanup_step02.sh)
- Build Docker image of Collector script

```
docker build -t devnet-2594/step-03:latest -t devnet-2594/step-03:1 .
```

- Deploy Prometheus container

```
docker run --name prometheus -d --network demo0 -p 127.0.0.1:9090:9090 \
-v ${PWD}/prometheus.yml:/etc/prometheus/prometheus.yml \
quay.io/prometheus/prometheus
```

- Deploy Collector container

```
docker run --name icam -d --network demo0 -p 127.0.0.1:8888:8888 \
devnet-2594/step-03
```

Enable query history

icam_l2_table_max

Load time: 15ms
 Resolution: 14s
 Total time series: 1

Execute - insert me ↴

Graph Console

- 1h + Until Res. ⚡ stacked

[Remove Graph](#)

icam_l2_table_time_seconds

Load time: 10ms
 Resolution: 14s
 Total time series: 1

Execute - insert me ↴

Graph Console

- 1h + Until Res. ⚡ stacked

[Remove Graph](#)

icam_l2_table_used

Load time: 13ms
 Resolution: 14s
 Total time series: 1

Execute - insert me ↴

Graph Console

- 1h + Until Res. ⚡ stacked

[Remove Graph](#)**Cisco live!**

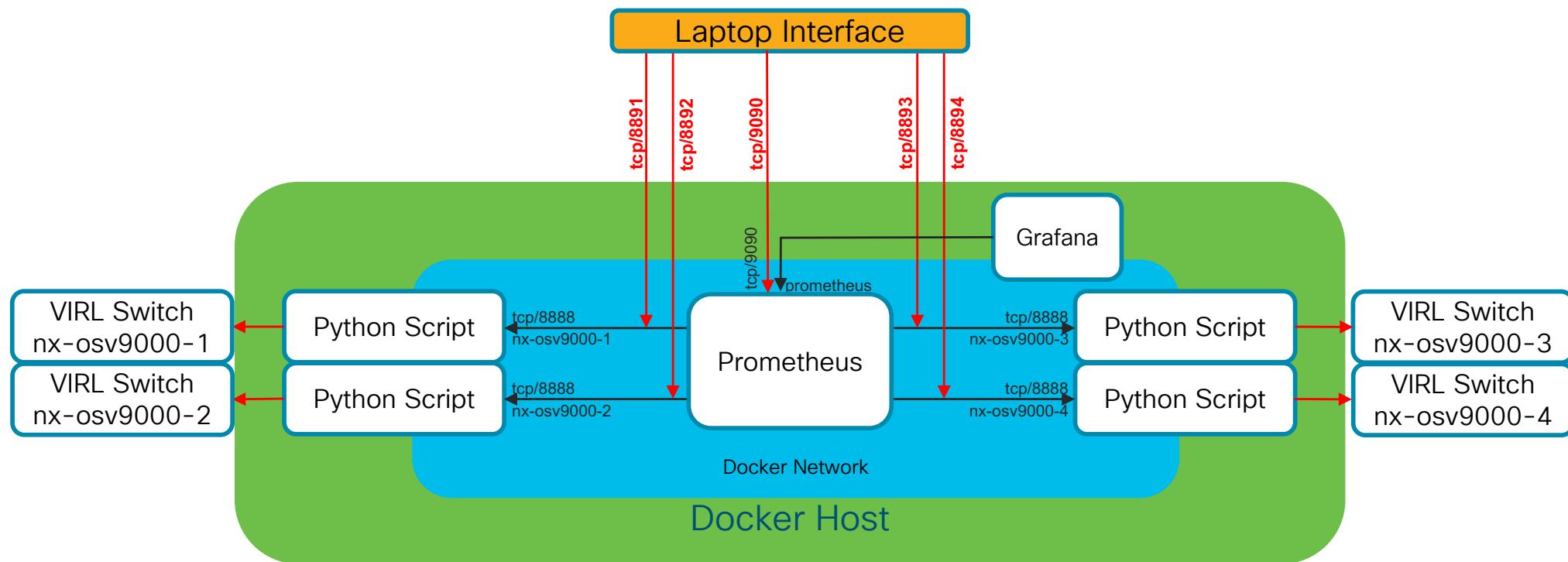
#CLUS

DEVNET-2594

© 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public

40

Metric Service Connectivity Diagram



Step-04 – Routing Metrics and Sandbox

- Clean up from step-03 (cleanup_step03.sh)
- VPN to the DEVNET Sandbox using instructions

- Build Docker image of Collector script

```
docker build -t devnet-2594/step-04:latest -t devnet-2594/step-04:1 .
```

- Deploy Prometheus container

```
docker run --name prometheus -d --network demo0 -p 127.0.0.1:9090:9090 \
-v ${PWD}/prometheus.yml:/etc/prometheus/prometheus.yml \
quay.io/prometheus/prometheus
```

Step-04 – Routing Metrics and Sandbox

- Deploy 4 collector container instances

```
docker run --name nx-osv9000-1 -d --network demo0 -p 127.0.0.1:8891:8888 \
-e "NXAPI_HOST=172.16.30.101" -e "NXAPI_PORT=80" \
-e "NXAPI_USER=cisco" -e "NXAPI_PASS=cisco" \
devnet-2594/step-04
```

```
docker run --name nx-osv9000-2 -d --network demo0 -p 127.0.0.1:8892:8888 \
-e "NXAPI_HOST=172.16.30.102" -e "NXAPI_PORT=80" \
-e "NXAPI_USER=cisco" -e "NXAPI_PASS=cisco" \
devnet-2594/step-04
```

Step-04 – Routing Metrics and Sandbox

- Deploy 4 collector container instances

```
docker run --name nx-osv9000-3 -d --network demo0 -p 127.0.0.1:8893:8888 \
-e "NXAPI_HOST=172.16.30.103" -e "NXAPI_PORT=80" \
-e "NXAPI_USER=cisco" -e "NXAPI_PASS=cisco" \
devnet-2594/step-04
```

```
docker run --name nx-osv9000-4 -d --network demo0 -p 127.0.0.1:8894:8888 \
-e "NXAPI_HOST=172.16.30.104" -e "NXAPI_PORT=80" \
-e "NXAPI_USER=cisco" -e "NXAPI_PASS=cisco" \
devnet-2594/step-04
```

Step-04 – Routing Metrics and Sandbox

- Deploy Grafana instance

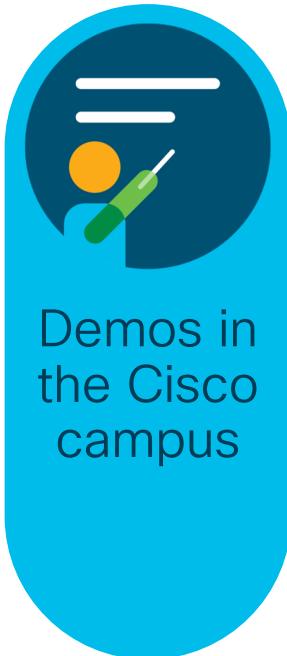
```
docker run --name grafana -d --network demo0 \
    -p 127.0.0.1:3000:3000 \
    grafana/grafana
```

Continue your education

- DEVNET-1897 : Coding 1001 - Intro to APIs and REST
- DEVNET-1725 : How to be a Network Engineer in a Programmable Age
- BRKDCN-2025 : Maximizing Network Programmability and Automation with Open NX-OS
- BRKDCN-2712 : DC Network Telemetry with Nexus and NX-OS



Continue your education



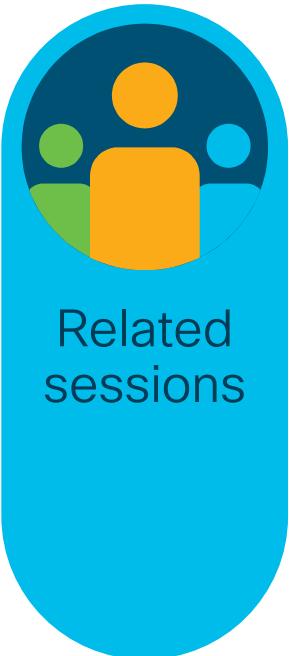
Demos in
the Cisco
campus



Walk-in
self-paced
labs



Meet the
engineer
1:1
meetings



Related
sessions

Cisco Webex Teams



Questions?

Use Cisco Webex Teams (formerly Cisco Spark) 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 Webex Teams or go directly to the team space
- 4 Enter messages/questions in the team space

Webex Teams will be moderated by the speaker until June 18, 2018.

cs.co/ciscolivebot#DEVNET-2594

Complete your online session evaluation

Give us your feedback to be entered into a Daily Survey Drawing.

Complete your session surveys through the Cisco Live mobile app or on www.CiscoLive.com/us.

Don't forget: Cisco Live sessions will be available for viewing on demand after the event at www.CiscoLive.com/Online.





Thank you

Cisco *live!*

#CLUS



INTUITIVE



INTUITIVE

#CLUS