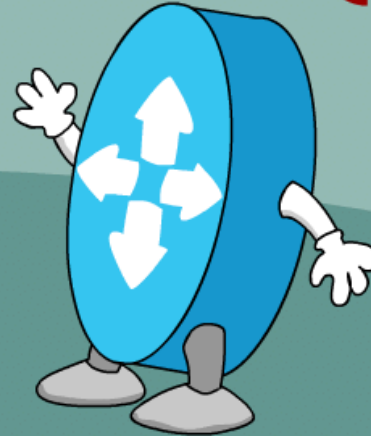




NETDEVOPS {LIVE!}



DEVNET

Stop Waiting for that Network Feature, Build it Yourself

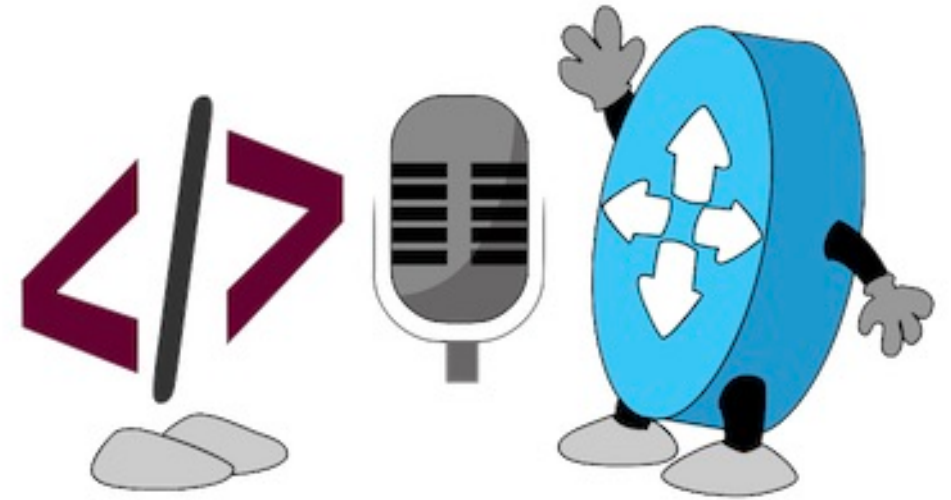
Adrian ILIESIU
DevNet Engineer
@aidevnet

Season 2 – Talk 12

<https://developer.cisco.com/netdevops/live>

What are we going to talk about?

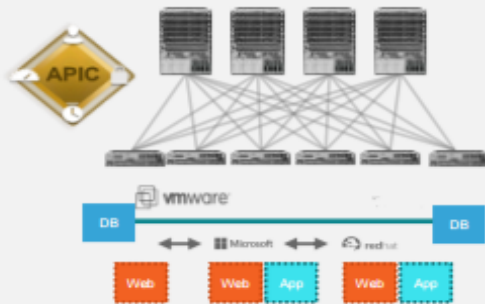
- Cisco Data Center Networks
- Extending NX-OS
 - GuestShell
 - Docker
 - NX-SDK



Cisco Data Center Networks

- Providing Choice in Automation and Programmability

Application Centric Infrastructure

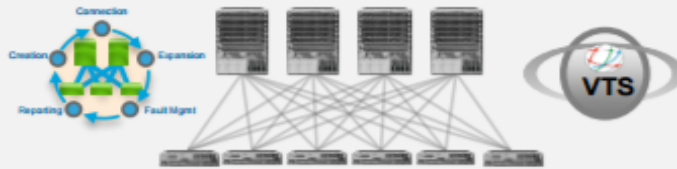


Turnkey integrated solution with security, centralized management, compliance and scale
Switches run in ACI mode

Automated application centric-policy model with embedded security

Broad and deep ecosystem

Programmable Fabric



VXLAN BGP EVPN
standard-based

Switches run in standalone NX-OS mode

Cisco Controller for software overlay provisioning and management across N2K-N9K

Programmable Network



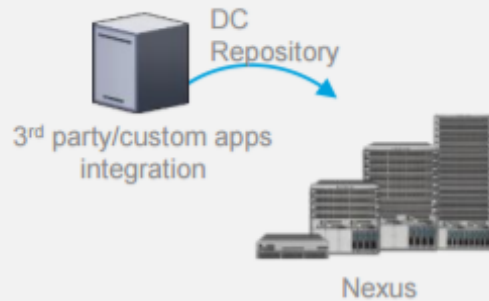
Modern NX-OS with enhanced NX-APIs

Switches run in standalone NX-OS mode

DevOps toolset used for Network Management

Open NX-OS Provides

3rd Party Linux Applications



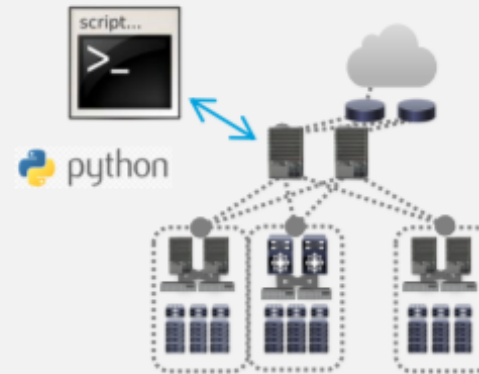
Use 3rd party applications using secure LXC containers

Linux Tools



Leverage Linux commands for config and troubleshooting

Programmable Open APIs



Object-based, model driven APIs (RESTful XML/JSON)

3rd Party DevOps Automation Tools



Leverage same software tools and expertise across different IT departments

Extending NX-OS

GuestShell

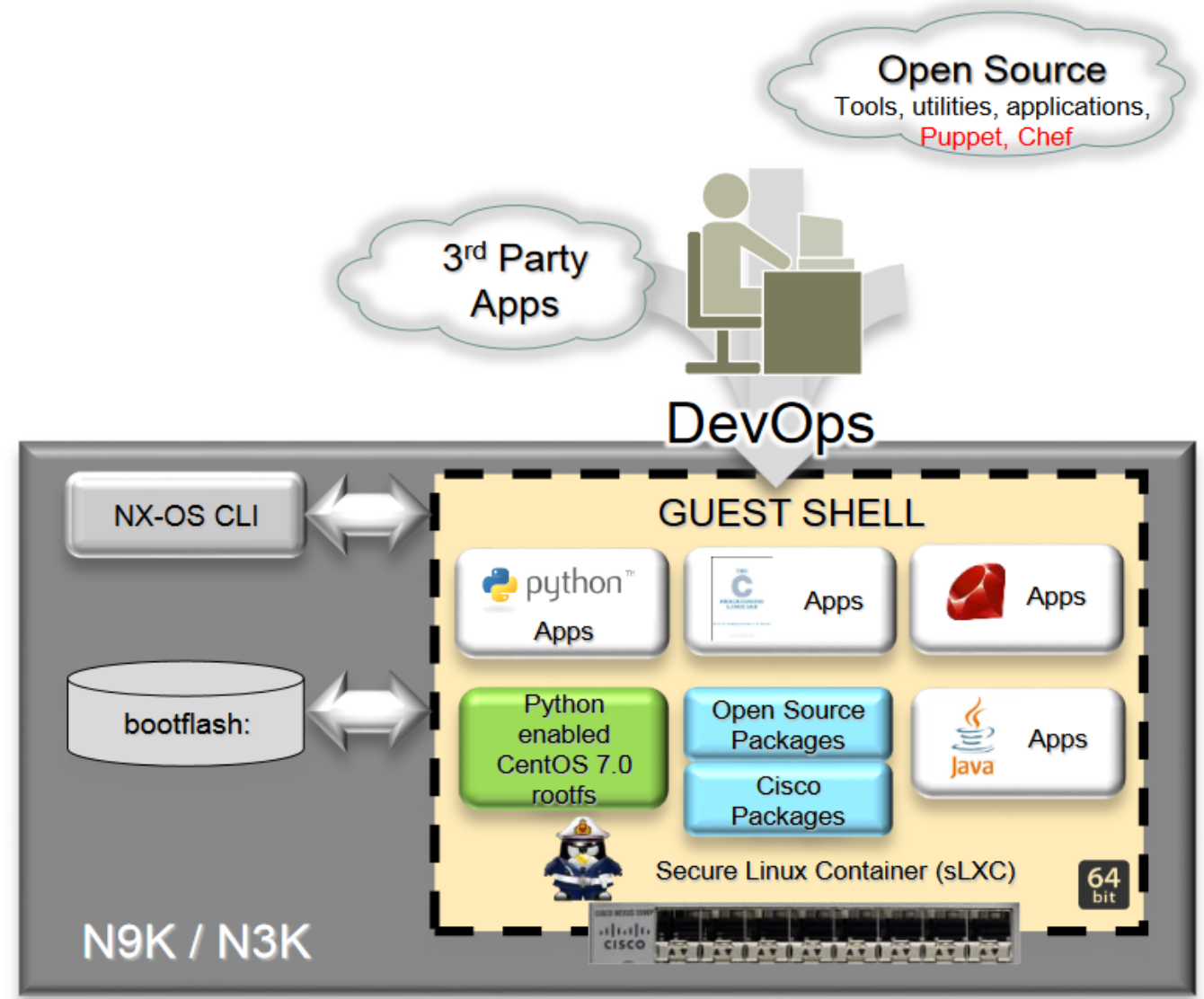
Guest Shell 2.0

It's an open Linux environment, decoupled from NX-OS.

It allows to run applications that monitor, control and extend the switch.

Supported on Nexus 3K and 9K today.

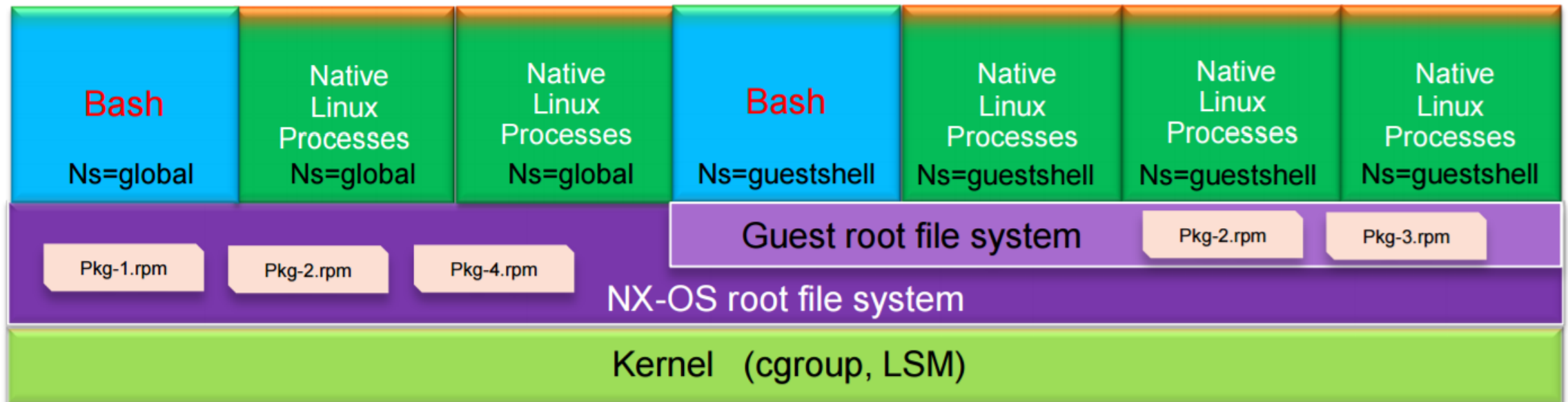
64-bit CentOS 7 application environment.



Open NX-OS: Third Party Application Integration

Secure Guest Shell

Native Shell, RPM +
Containers

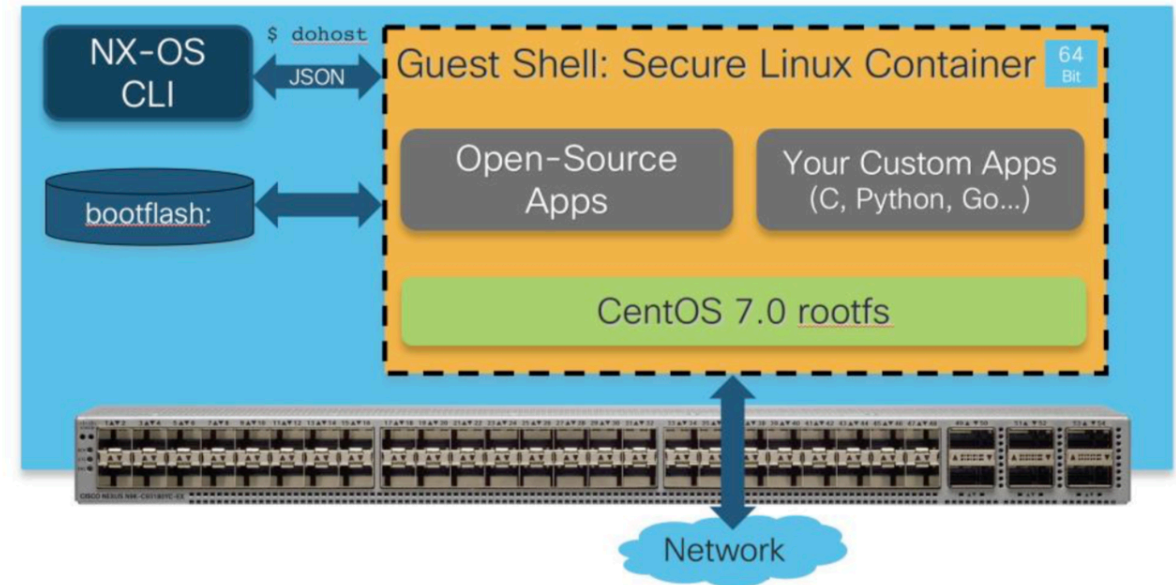


- Secure common distribution CentOS7 environment in which customer may install their own custom applications
- Use “guestshell resize” command to restrict CPU/memory/rootfs resources available to Guest Shell

Docker

Securely Run 3rd Party Apps Prior to NX-OS 9.2(1)

- Prior to NX-OS 9.2(1), the only solution is the [Guest Shell](#).
- It works fine but...
- Only [one Guest Shell instance](#) can be defined.
- The container Linux distribution is always [CentOS](#).
- The setup of apps in the Guest Shell must be done from a [Nexus 9K](#).
- Re-using and distributing Guest Shell images can be done with the [import/export](#) feature, but is fairly limited.
- [No container orchestration](#) support.

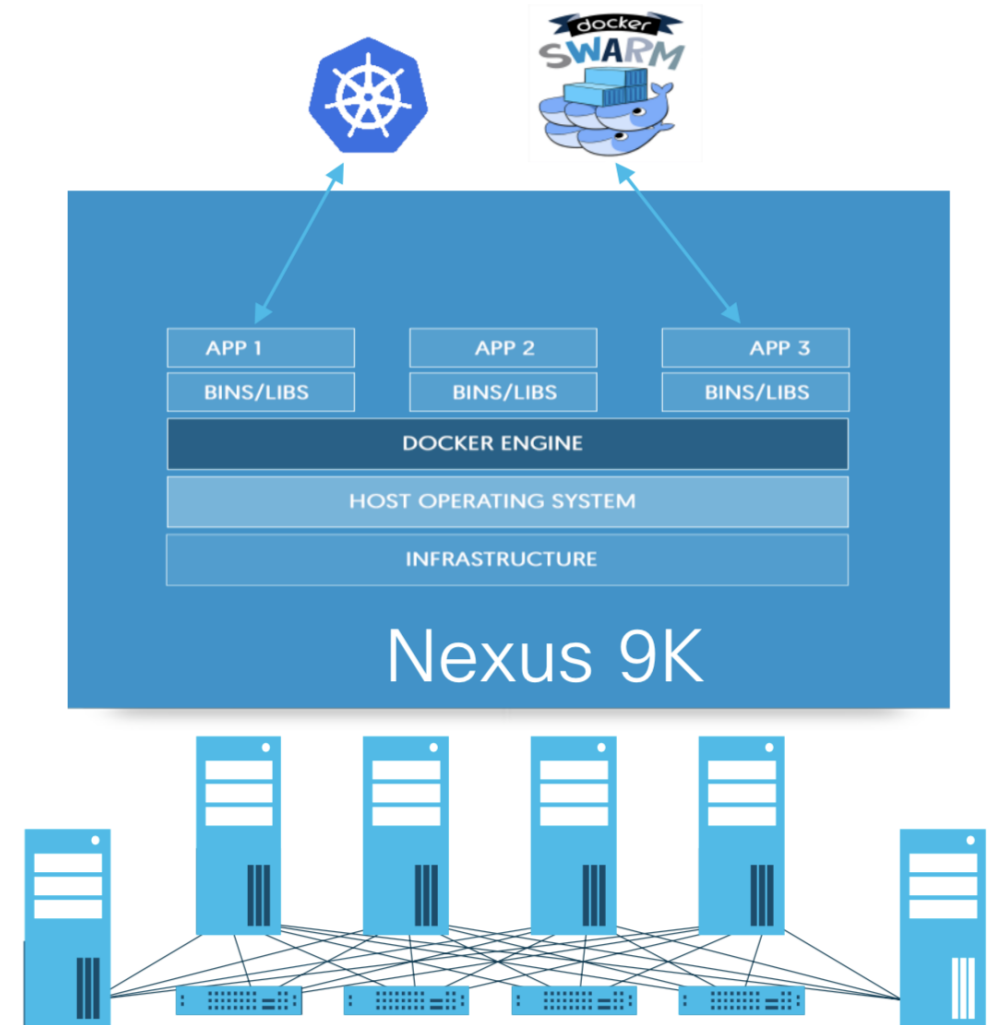


Docker Engine


NX-OS 9.2(1) - July 2018



- Available on **all Nexus 9K models**, and on Nexus 3K models equipped with 8G+ of memory.
- Standard Docker engine with all the commands supported:
docker run/pull/push/kill/info etc.



Standardization, Flexibility, and Efficiency

	Guest Shell	Docker Engine 
Number of container instances	One	Many
Access to storage and network	Yes	Yes
Linux distribution type	CentOS	Any
Container manipulation primitives	NX-OS CLI (# guestshell *)	Standard Linux docker tool
Definition of the container image content	Must be done on a Nexus 9K	Can be done from any computer supporting Docker
Repository of existing container images	None	Docker Hub
Container orchestration	None	Docker Swarm or Kubernetes

What Apps are Interesting to Host on N9K with Docker?

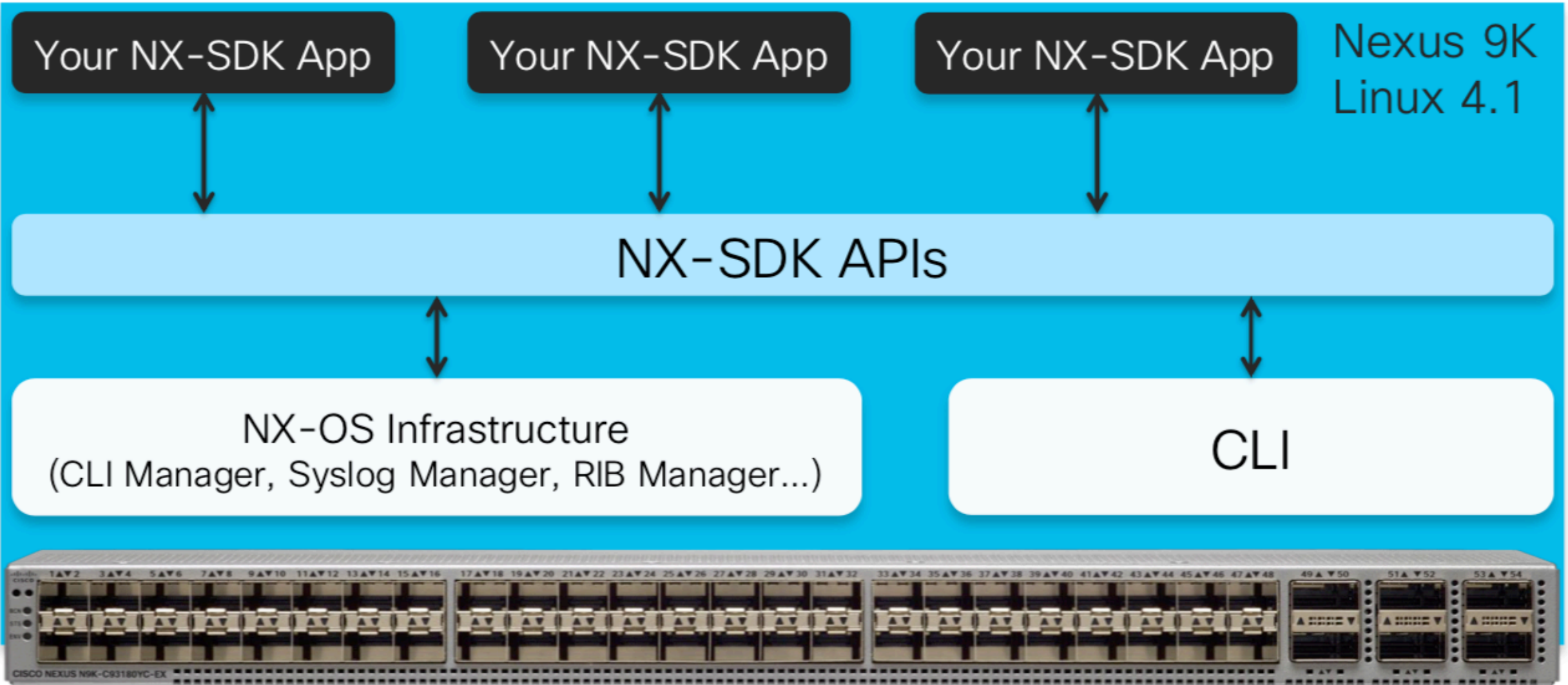
- **Monitoring agents**
 - **Open-source** agents: OpenTSDB, Ganglia, Nagios, etc.
Monitor both **standard Linux** components (CPU, memory, interface counters), and **NX-OS** (routes, buffers,...)
 - **Custom** agents: ECMP load balancing, PTP accuracy...
- **Automation agents** (Chef, Puppet, SaltStack...)
- **Intrusion Detection**
 - DNSFlow agent to detect phishing activity
 - Custom Intrusion Detection agents
- **Automatic configuration backup** to a private Git repository

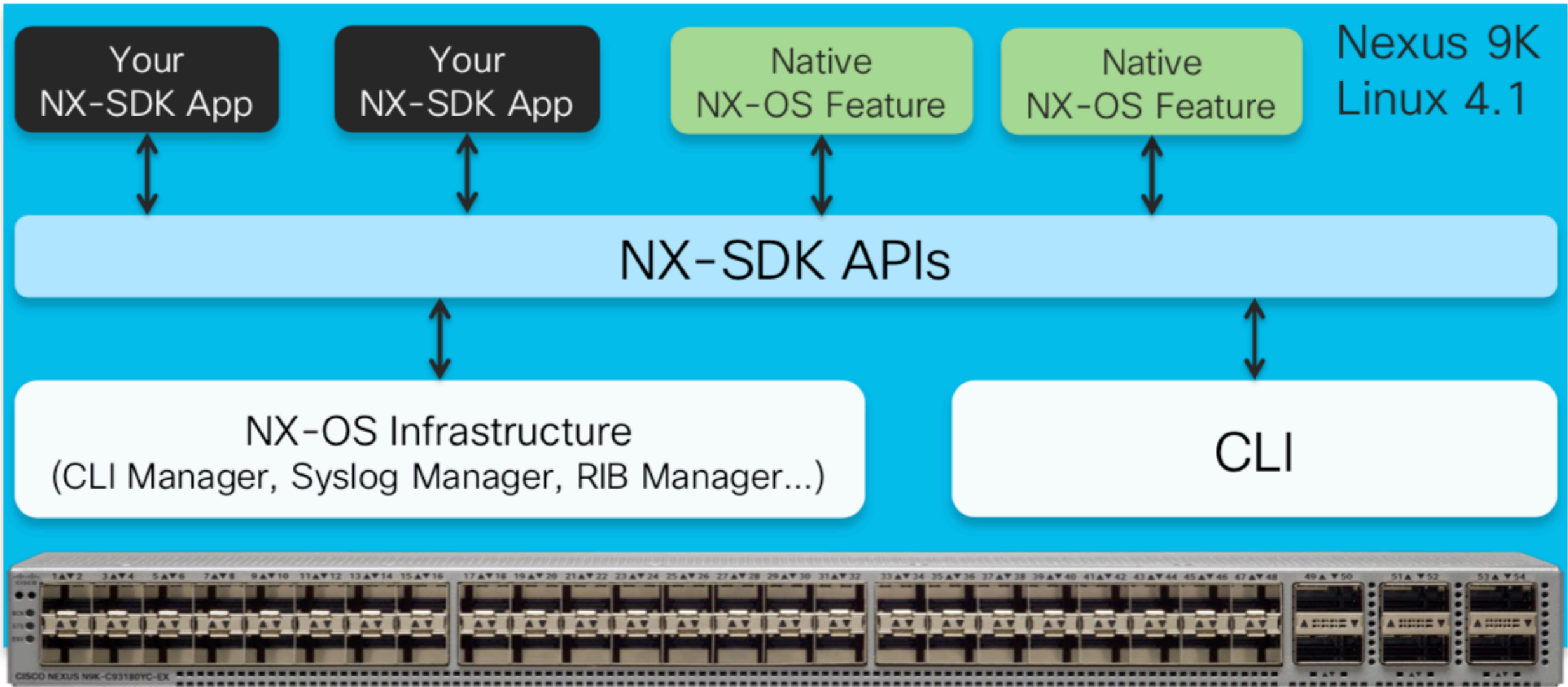
NX-SDK

Tighter Integration of Custom Linux Apps with NX-OS



- **NX-SDK** is a simple, flexible and powerful set of APIs for custom on-box applications to gain access to NX-OS infra functionalities.
- Apps are still regular Linux programs. They just use the NX-SDK APIs.
- Apps run natively as NX-OS features just like the Cisco-developed features.
- Startup and management of the apps is handled by NX-OS.







NX-SDK Features

NX-OS 7.0(3)I6(1) – May 2017

- Python and C++ support.
- Definition of custom CLIs:
 - Config and show commands.
 - Callback handler with your code gets invoked when the CLI gets executed.
- Generation of custom syslogs.

- Gain access to NX-OS CLI:

```
cli = sdk.getCliParser()
```
- Create a new custom CLI config command:

```
cli.newConfigCmd("threshold_cmd",  
                 "threshold <value>")  
cli.updateParam("<value>",  
               "Threshold value in Mbps",  
               nx_sdk_py.P_INTEGER)
```
- Add our command handler callback for custom CLI:

```
myCmd = pyCmdHandler()  
cliP.setCmdHandler(myCmd)
```

Cisco *live!*

NX-SDK Features

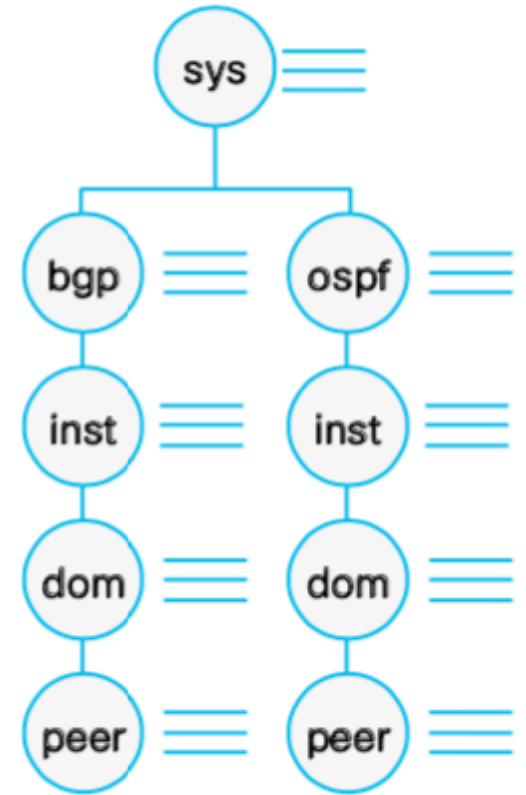
NX-OS 7.0(3)I7(3) – February 2018



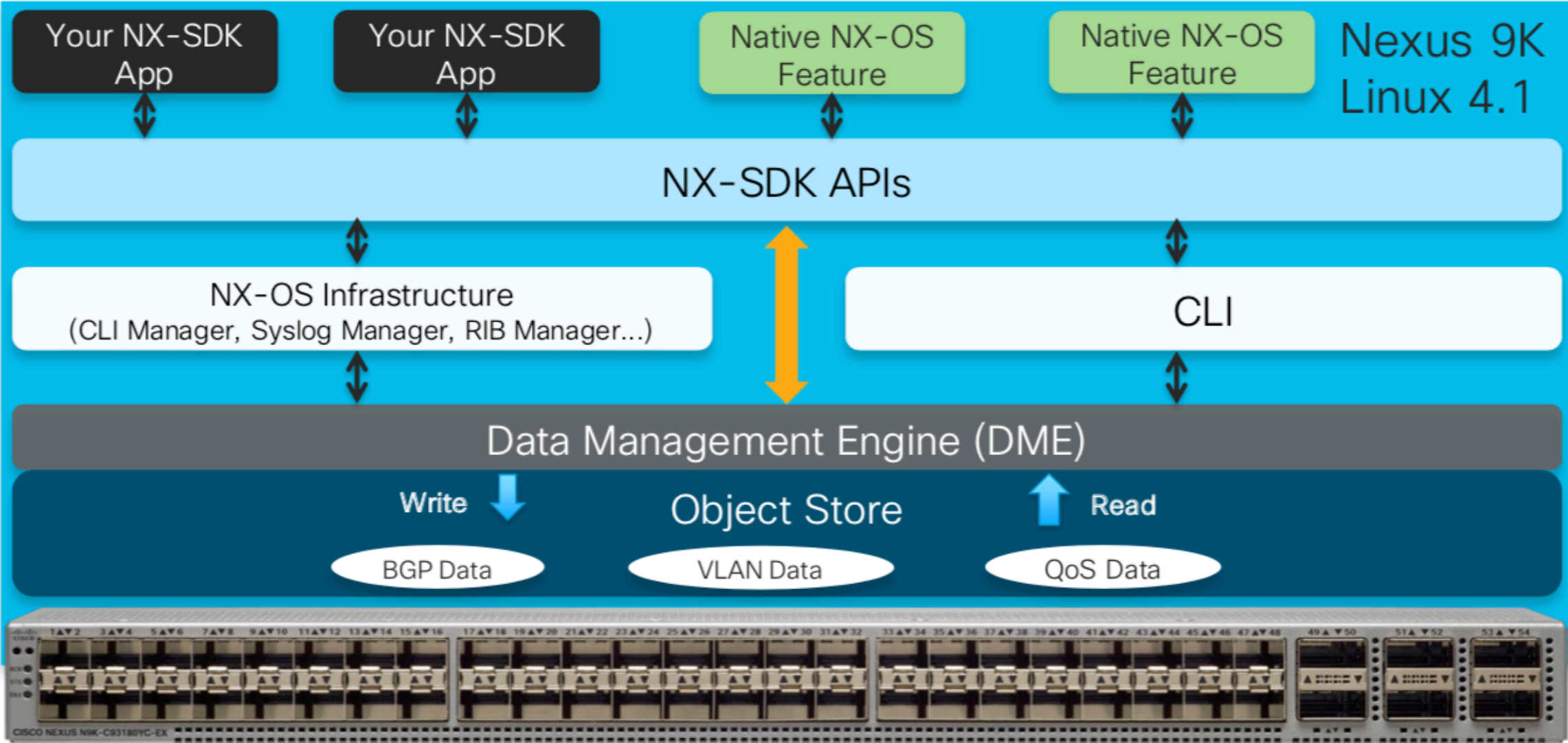
- **RIB APIs:**
 - Route **lookup**.
 - Route **events**. Get notified upon updates: route add, route remove, next-hop change.
 - Granularity: **prefix, protocol, address family, VRF**.
- **Streaming Telemetry** support.
- Gain access to NX-OS **RIB**:
`ribMgr = sdk.getRibMgr()`
- Register a RIB **callback** handler, that will be invoked when a **route event** happens:
`myRibCb = pyRibHandler()`
`ribMgr.setRibMgrHandler(myRibCb)`
- **Subscribe** for route events:
`ribMgr.watchL3Route("direct")`
`ribMgr.watchL3Route("bgp")`

NX-OS Structured Data Model

- **Object Store:**
 - Hierarchical **tree**-structure representing switch **configuration** and **state**.
 - Comprised of **Managed Objects (MOs)** representing an element of configuration or operational state.
- **Data Management Engine (DME):** read/write **interface** to the object store.
- Object store documentation and examples on cisco.com: <https://developer.cisco.com/site/nxapi-dme-model-reference-api/#>.
- Browse the object store on the switch itself: <https://<switch management IP>/visore.html>.



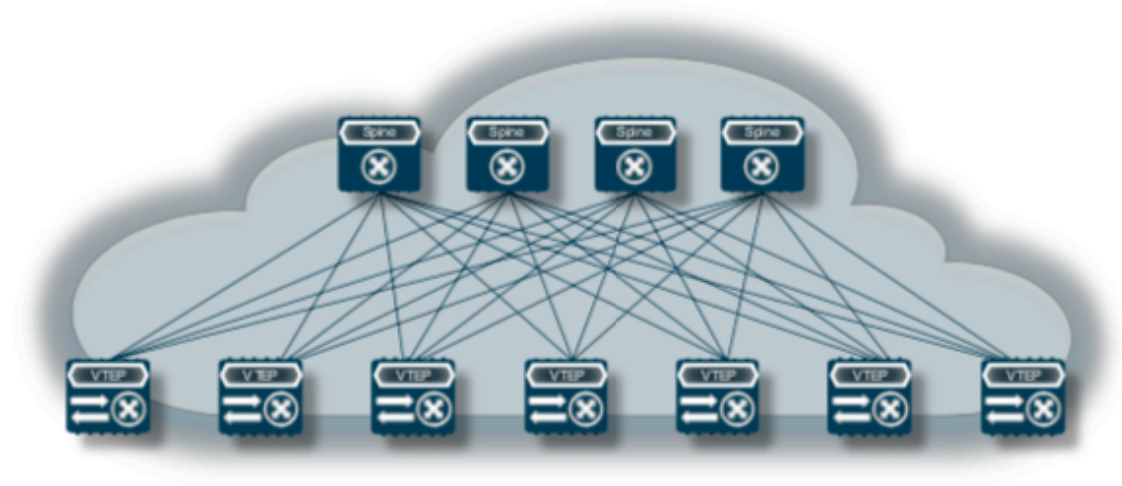
↔ Direct DME access



NX-SDK Use Cases

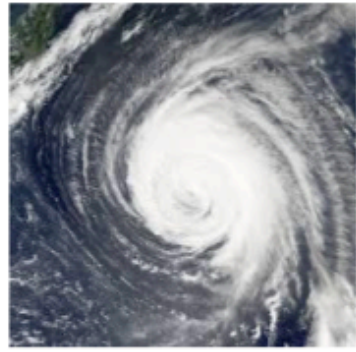
ECMP Imbalance Monitoring

- Automatically detect ECMP bundles.
- Define a load threshold via a custom CLI.
- Monitor the load of ECMP member links to detect hashing imbalance.
- When high load i.e., hashing imbalance is detected:
 - Display a custom syslog.
 - Trigger a streaming telemetry event.



NX-SDK Use Cases

Storm Control Persistence Protection



- Storm Control: shut down an interface victim of storm traffic.
- If enabled, the shutdown happens immediately in the current NX-OS implementation.
- NX-SDK enhancements:
 - Shut it down only if storm traffic is **persisting** after a given interval of time.
 - Interfaces can be **automatically brought back up** after an interval of time.
- Both time intervals are defined by custom CLIs:

```
switch(config)# storm.py ?
  shut-interval      Interval in seconds after which the interfaces on which storm
                    traffic persists are shut down
  unshut-interval    Interval in seconds after which the interfaces that got shut
                    down by the app are brought back up
```

NX-SDK Use Cases

Precision Timing Protocol (PTP) Monitoring

- Display a custom syslog upon:
 - Change of interface PTP state: master / slave / passive / disabled.
 - Change of PTP Grandmaster.
 - High clock correction, with threshold defined via a custom CLI.
- Rogue Grandmaster enforcement:
 - Define interfaces which must only be connected to PTP slaves.
 - When detecting a connection to a master, automatically shut them down.



```
switch# show ptp clock
PTP Device Type: Boundary clock
Clock Identity :
00:fe:c8:ff:fe:09:7f:f3
Clock Domain: 0
Number of PTP ports: 3
Priority1 : 255
Priority2 : 255
Clock Quality:
  Class : 248
  Accuracy : 254
  Offset (log variance) : 65535
```


NX-SDK Demo

Environment

- Cisco Nexus9000v version 9.2(3)
- NX-SDK docker image v1.7.5
- Visual Studio Code

Connect to NX-OSv

- Connect to the device and enable features

```
ssh admin@X.X.X.X  
  
feature scp-server  
feature bash-shell  
feature nxsdk
```

Review the new feature

```
git clone https://github.com/CiscoDevNet/NX-SDK.git  
cd /NX-SDK/examples/python/  
code pbwMonitor
```

Build the rpm for the new feature

```
docker pull dockercisco/nxsdk:v1.7.5
docker run -it dockercisco/nxsdk:v1.7.5 /bin/bash
python scripts/rpm_gen.py pbwMonitor -s examples/python/ -u
cd /NX-SDK/rpm/RPMS/
scp pbwMonitor-1.0-1.7.5.x86_64.rpm admin@128.107.70.6x:
```

Install the new feature

```
ssh admin@128.107.70.6x
dir bootflash:
config t
install add bootflash:pbwMonitor-1.0-1.7.5.x86_64.rpm
show install inactive
install activate pbwMonitor-1.0-1.7.5.x86_64
show install active
nxsdk service-name pbwMonitor
```

Run the new feature

```
show pbwMonitor port bw utilization  
config t  
pbwMonitor port bw threshold x  
show nxsdk internal service  
sh run nxsdk
```

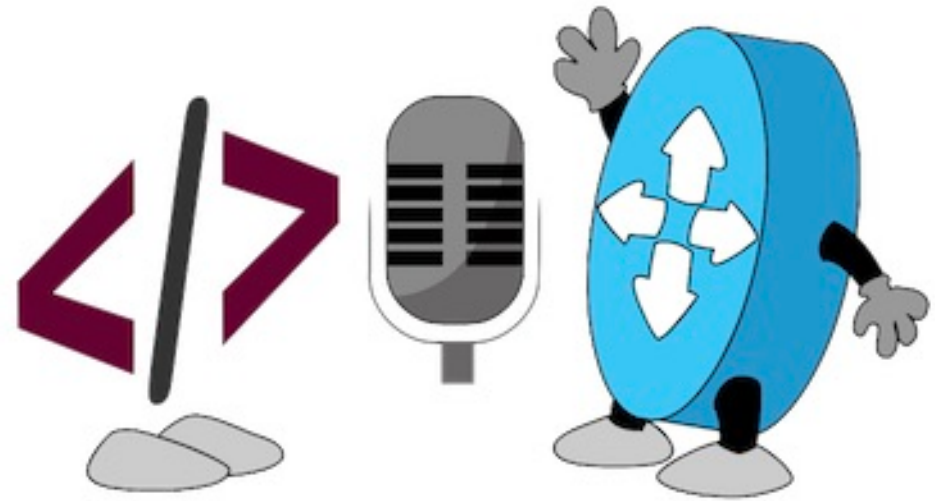
Remove the feature from the switch

```
config t
no nxsdk service-name pbwMonitor
install deactivate pbwMonitor-1.0-1.7.5.x86_64
show install inactive
install remove pbwMonitor-1.0-1.7.5.x86_64.rpm
```

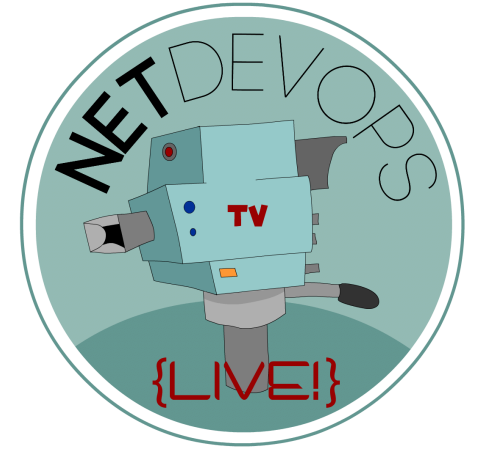

Summing up

What did we talk about?

- Cisco Data Center Networks
- Extending NX-OS
 - GuestShell
 - Docker
 - NX-SDK



Webinar Resource List



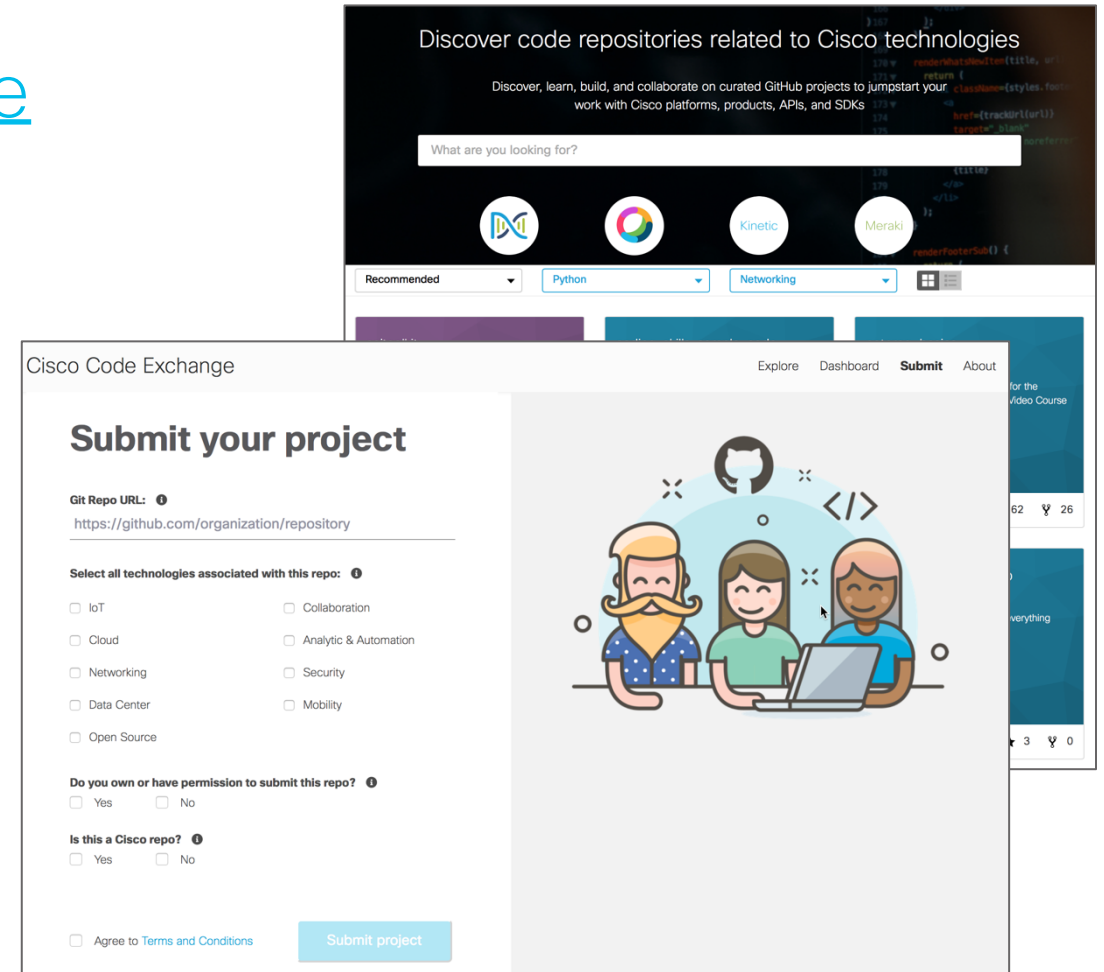
- Open NX-OS on DevNet
 - <https://developer.cisco.com/nx-os>
- NX-SDK Documentation
 - https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/92x/programmability/guide/b-cisco-nexus-9000-series-nx-os-programmability-guide-92x/b-cisco-nexus-9000-series-nx-os-programmability-guide-92x_chapter_010001.html
- Docker on NX-OS Documentation
 - https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/92x/programmability/guide/b-cisco-nexus-9000-series-nx-os-programmability-guide-92x/b-cisco-nexus-9000-series-nx-os-programmability-guide-92x_chapter_010010.html
- NX-OS Programmability Learning Labs
 - <https://developer.cisco.com/learning/tracks/nxos-programmability>

NetDevOps Live! Code Exchange Challenge

developer.cisco.com/codeexchange

Build a custom show command highlighting details that matter to you!

Example: Create a new version of “show interfaces” that displays the specific details you look for when troubleshooting.



The image shows two overlapping screenshots of the Cisco Code Exchange website. The top screenshot displays a search interface with the heading "Discover code repositories related to Cisco technologies" and a search bar. Below the search bar are icons for various technologies: Python, Kinetic, and Meraki. The bottom screenshot shows the "Submit your project" form, which includes a "Git Repo URL" field, a "Select all technologies associated with this repo" section with checkboxes for IoT, Cloud, Networking, Data Center, Open Source, Collaboration, Analytic & Automation, Security, and Mobility, and a "Submit project" button.

Looking for more about NetDevOps?

- NetDevOps on DevNet
developer.cisco.com/netdevops
- NetDevOps Live!
developer.cisco.com/netdevops/live
- NetDevOps Blogs
blogs.cisco.com/tag/netdevops
- Network Programmability Basics Video Course
developer.cisco.com/video/net-prog-basics/



Got more questions? Stay in touch!



developer.cisco.com

 @aidevnet

 <http://github.com/aidevnet>

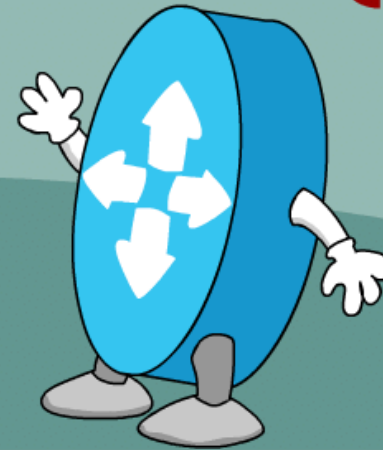
 @CiscoDevNet

 facebook.com/ciscocodevnet/

 <http://github.com/CiscoDevNet>



NETDEVOPS {LIVE!}



DEVNET

<https://developer.cisco.com/netdevops/live>

@netdevopslive 