

Examining Nexus OS Automation and Scripting Tools



Sean Douglas

DATA CENTER ENGINEER

@ocdlearning



Overview



Automate the day-to-day management, monitoring, and configuration to increase efficiency and help eliminate errors

- NX-OS
- Cisco Embedded Event Manager
- Bash and Guest Shell for NX-OS



Cisco NX-OS Programmability



Traditional Network Management



Traditional network management process uses CLI to send commands

- Text editor, copy/paste
- Tcl scripts
- CLI can lead to errors

The challenge with traditional network management process

- Managing scalable infrastructure
- The network lags behind industry automation capabilities



Programmability Benefits



Programmability benefits

- Saves resources
- Enables fast and flexible service delivery
- Minimizes human error
- Allows customization and innovation



Cisco Nexus Software

Modular

Highly programmatic

Secure

Flexible

Scalable

Easy to use



Standard Network Management Interfaces



CLI

- Designed as a person readable interface
- Returns unstructured data

SNMP

- Used for monitoring of network devices
- NX-OS supports SNMP v1, v2, v3

NETCONF

- Used for monitoring of network devices

Advanced Automation Features



Cisco Nexus platform includes the following advanced automation features:

- POAP support
- XXMP support
- Chef and Puppet integration
- OpenStack integration



Nexus 9000 Programmability Support



Nexus 9000 also support:

- Python scripting
- Bash
- Bash shell access and Linux container support
- Guest Shell



Nexus IOS Scheduler and EEM



Scheduler Components

Job:

- A routine task or tasks defined as a command list

Schedule:

- The timetable for completing a job
- **Periodic mode:** a recurring interval (daily, weekly, monthly)
- **One-time mode:** a job completed only once at a specific time



Scheduler Configuration



Define a job:

- Configure the scheduler job name
- Add a sequence of commands separated with a space and a semicolon

Define a schedule:

- Configure the schedule name
- Associate a job with a schedule
- Configure a periodic or one-time execution for a scheduler

Scheduler Job

```
switch# configure terminal
```

```
switch(config)# scheduler job name MY_CFG_BACKUP
```

```
switch(config-job)# cli var name timestamp $(TIMESTAMP) ;copy running-  
config bootflash:/$(SWITCHNAME)-cfg.$(timestamp) ;copy  
bootflash:/$(SWITCHNAME)-cfg.$(timestamp) tftp://10.1.1.1/ vrf management
```

```
switch(config-job)# end
```

Scheduler Configuration

```
switch(config)# scheduler schedule name MYDAILY
switch(config-schedule)# job name MY_CFG_BACKUP
switch(config-schedule)# time daily 4:00
switch(config-schedule)# end
```

Verify the Job Schedule

```
switch# show scheduler schedule
```

```
Schedule Name : MYDAILY
```

```
-----
```

```
User Name : admin
```

```
Schedule Type : Run every day at 4 Hrs 00 Mins
```

```
Last Execution Time : Sat Oct 21 4:00:00 2019
```

```
Last Completion Time: Sat Oct 21 4:00:01 2019
```

```
Execution count : 2
```

`show scheduler config`: displays the scheduler configuration

`show scheduler job`: displays the jobs configured

`show scheduler logfile`: displays the contents of the scheduler log file

`show scheduler schedule`: displays the schedules configured

Verify the Scheduler

To verify the scheduler, use these commands



Cisco Embedded Event Manager



EEM Policy

EEM consists of three major components:

- Event statements
- Action statements
- Policies

You can configure EEM policies using the CLI or a virtual shell (VSH) script



EEM Event Statements



Some examples of events:

- **CLI:** with a command that matches the regular expression
- **Counter:** Add a sequence of commands separated with a space and a semicolon
- **Module:** if the specified module enters the selected status
- **Memory:** if a memory threshold is crossed
- **None:** manually runs the policy event without any events specified

EEM Action Statements

EEM supports the following actions in action statements:

- Execute any CLI commands
- Update a counter
- Log an exception
- Force the shutdown
- Reload the device
- Shut down specified modules because the power is over budget
- Generate a syslog
- Generate a Call Home event
- Generate an SNMP notification



Configure Cisco IOS EEM

Register the applet with EEM and enters applet configuration mode

```
event manager applet applet-name
```

Register the applet with EEM and enters applet configuration mode

```
event event-statement
```

Configure an action statement for the policy

```
action number[.number2] action-statement
```



Example IOS EEM Policy

```
event manager applet COPY_RUN
event cli match "copy run* start*"
action 1.0 syslog priority alerts msg "Running configuration is saved"
action 2.0 cli local command "show version | grep uptime >> bootflash:/uptime"
action 3.0 event-default
```

```
switch# copy running-config startup-config
```

```
[#####] 100%
```

```
Copy complete
```

Copy Run Start

Use the `copy run start` command to verify the EEM policy



```
switch# show logging | grep "Running configuration is saved"
```

```
2019 Oct 20 14:17:00 switch %EEM_ACTION-1-ALERT: Running configuration is saved
```

Verify Logs



Verify the EEM History

switch# show event manager history events

Event ID	Time of Event	Event Type	Slot	Policies
89	10/20/2019 14:31:12	cli	active(1)	COPY_RUN
88	10/20/2019 14:33:11	cli	active(1)	COPY_RUN

Bash Shell and Guest Shell for Nexus



NX-OS Supports Two Linux Environments



Bash Shell

- Allows access to the underlying Linux system
- Disabled by default

Guest Shell

- Secure Linux container environment
- Separate from the host software
- Allows us to add software packages and update libraries as needed
- Enabled by default

```
switch# configure terminal  
switch(config)# feature bash-shell  
switch(config)# end  
switch# run bash
```

Enable Bash

After feature is enabled, you can access the Bash



Bash Commands

```
bash-4.2$ whoami
```

```
admin
```

```
bash-4.2$ pwd
```

```
/bootflash/home/admin
```

```
username sdouglas shelltype bash
```

Bash Commands

```
#!/bin/bash
```

```
i=0
```

```
while [ $i -lt 120 ]
```

```
do
```

```
    echo "`date`: `vsh -c "show ip route" | grep ubest | wc -l`" >> route_count
```

```
    sleep 30
```

```
    i=$((i+1))
```

```
done
```

```
switch# run bash /bin/bash /bootflash/home/admin/script.sh
```

Run Script

We can run the script from the NX-OS CLI



```
switch# show file bootflash:home/admin/route_count
```

```
Fri Oct 20 10:27:09 UTC 2019: 16
```

```
Fri Oct 20 10:27:39 UTC 2019: 16
```

```
Fri Oct 20 10:28:10 UTC 2019: 16
```

Verify

Verify the file with outputs



Bash Commands

```
bash-4.2$ yum list installed | grep n9000
```

base-files.n9000	3.0.14-r74.2	installed
bfd.lib32_n9000	1.0.0-r0	installed
core.lib32_n9000	1.0.0-r0	installed
eigrp.lib32_n9000	1.0.0-r0	installed
eth.lib32_n9000	1.0.0-r0	installed
isis.lib32_n9000	1.0.0-r0	installed

Guest Shell

Guest Shell is a decoupled execution space running within a Linux Container

From within the Guest Shell the network-admin has the following capabilities:

- Access to the network
- Access to Cisco Nexus bootflash
- The ability to install and run python scripts
- The ability to install and run 64-bit Linux applications



`run guestshell` or `guestshell` commands to access the Guest Shell

`run guestshell command` command to execute the command in Guest Shell

`dohost command` command to run NX-OS command from Guest Shell

Guest Shell Commands



```
switch# guestshell
```

Guest Shell Commands



`guestshell enable` installs and activates the Guest Shell

`guestshell disable` shuts down and disables the Guest Shell

`guestshell upgrade` deactivates and upgrades the Guest Shell

`guestshell reboot` deactivates the Guest Shell and then reactivates it

`guestshell destroy` deactivates and uninstalls the Guest Shell

`guestshell resize` changes the allotted resources available for the Guest Shell

`show guestshell detail` displays details about the Guest Shell

Guest Shell

```
switch# show guestshell detail
```

```
Virtual service guestshell+ detail
```

```
State : Activated
```

Package information

```
Name : rootfs_puppet
```

```
Path : usb2:/rootfs_puppet
```

Application

```
Name : GuestShell
```

```
Installed version : 2.3(0.0)
```

```
Description : Exported GuestShell: 20191024T1722697
```

Signing

```
Key type : Unsigned
```

```
Method : Unknown
```

Licensing

```
Name : None
```

```
Version : None
```

Guest Shell ifconfig output

```
[admin@guestshell ~]$ ifconfig Eth1-47
```

```
Eth1-47: flags=4098<BROADCAST,MULTICAST>  mtu 1500
```

```
    ether ea:80:f4:3d:2c:7d  txqueuelen 100  (Ethernet)
```

```
    RX packets 0   bytes 0 (0.0 B)
```

```
    RX errors 0   dropped 0  overruns 0  frame 0
```

```
    TX packets 0   bytes 0 (0.0 B)
```

```
    TX errors 0   dropped 0 overruns 0  carrier 0  collisions 0
```

```
[admin@guestshell ~]$ ls -al /var/run/netns
```

```
total 2
```

```
drwxrwxrwx 2 root root 80 Mar 11 19:11 .
```

```
drwxr-xr-x 9 root root 1024 Oct 17 11:34 ..
```

```
-r----- 1 root root 0 Mar 11 19:11 default
```

```
-r----- 1 root root 0 Mar 11 19:11 management
```



```
[admin@guestshell ~]$ ifconfig | grep Eth1
```

```
Eth1-22: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
```

```
Eth1-23: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
```

```
Eth1-24: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
```

```
Eth1-25: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
```

```
[admin@guestshell ~]$ chvrf management
```

```
[admin@guestshell ~]$ ifconfig | grep Eth1
```

```
[admin@guestshell ~]$ ifconfig
```

```
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
```

```
    inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
```

```
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 16436
```

```
    inet 127.0.0.1  netmask 255.255.0.0
```

Summary



Programmability tools can be deployed to increase efficiency, improve management, and scalability

- NX-OS
- Cisco Embedded Event Manager
- Bash and Guest Shell

Guest Shell Summary



Guest Shell is a virtualized Linux-based environment, designed to run custom Linux applications, for automated control and management of Cisco devices



Using the Guest Shell, you can also install, update, and operate third-party Linux applications



A container shell that provides a secure environment, in which users can install scripts or software packages and run them



Guest Shell Summary

Guest Shell is a virtualized Linux-based environment, designed to run custom Linux applications, for automated control and management of Cisco devices

Using the Guest Shell, you can also install, update, and operate third-party Linux applications

A container shell that provides a secure environment, in which users can install scripts or software packages and run them

