



# Bash

---

- [About Bash, on page 1](#)
- [Accessing Bash, on page 1](#)
- [Escalate Privileges to Root, on page 2](#)
- [Examples of Bash Commands, on page 3](#)

## About Bash

In addition to the NX-OS CLI, Cisco Nexus 3500 Series devices support access to the Bourne-Again SHell (Bash). Bash interprets commands that you enter or commands that are read from a shell script. Using Bash enables access to the underlying Linux system on the device and to manage the system.

## Accessing Bash

In Cisco NX-OS, Bash is accessible from user accounts that are associated with the Cisco NX-OS dev-ops role or the Cisco NX-OS network-admin role.

The following example shows the authority of the dev-ops role and the network-admin role:

```
switch# show role name dev-ops

Role: dev-ops
  Description: Predefined system role for devops access. This role
  cannot be modified.
  Vlan policy: permit (default)
  Interface policy: permit (default)
  Vrf policy: permit (default)
-----
Rule   Perm   Type   Scope   Entity
-----
4      permit command  conf t ; username *
3      permit command  bcm module *
2      permit command  run bash *
1      permit command  python *

switch# show role name network-admin

Role: network-admin
  Description: Predefined network admin role has access to all commands
  on the switch
-----
```

```

Rule      Perm      Type      Scope      Entity
-----
1         permit   read-write
switch#

```

Bash is enabled by running the **feature bash-shell** command.

The **run bash** command loads Bash and begins at the home directory for the user.

The following examples show how to enable the Bash shell feature and how to run Bash.

```

switch# configure terminal
switch(config)# feature bash-shell

switch# run bash
Linux# whoami
admin
Linux# pwd
/bootflash/home/admin
Linux#

```



**Note** You can also execute Bash commands with the **run bash <command>** command.

The following is an example of the **run bash <command>** command.

```
run bash whoami
```

## Escalate Privileges to Root

The privileges of an admin user can escalate their privileges for root access.

The following are guidelines for escalating privileges:

- Only an admin user can escalate privileges to root.
- Bash must be enabled before escalating privileges.
- Escalation to root is password protected.
- SSH to the switch using `root` username through a non-management interface will default to Linux Bash shell-type access for the root user. Type `vsh` to return to NX-OS shell access.

The following example shows how to escalate privileges to root and how to verify the escalation:

```

switch# run bash
Linux# sudo su root

```

```

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

```

- ```

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.

```

```
Password:
```

```
Linux# whoami
```

```
root
Linux# exit
exit
```

## Examples of Bash Commands

This section contains examples of Bash commands and output.

### Displaying System Statistics

The following example shows how to display system statistics:

```
switch# run bash
Linux# cat /proc/meminfo
MemTotal:      3795100 kB
MemFree:       1472680 kB
Buffers:       136 kB
Cached:        1100116 kB
ShmFS:         1100116 kB
Allowed:       948775 Pages
Free:          368170 Pages
Available:     371677 Pages
SwapCached:    0 kB
Active:        1198872 kB
Inactive:      789764 kB
SwapTotal:     0 kB
SwapFree:      0 kB
Dirty:         0 kB
Writeback:     0 kB
AnonPages:     888272 kB
Mapped:        144044 kB
Slab:          148836 kB
SReclaimable: 13892 kB
SUnreclaim:   134944 kB
PageTables:    28724 kB
NFS_Unstable: 0 kB
Bounce:        0 kB
WritebackTmp: 0 kB
CommitLimit:  1897548 kB
Committed_AS: 19984932 kB
VmallocTotal: 34359738367 kB
VmallocUsed:   215620 kB
VmallocChunk: 34359522555 kB
HugePages_Total: 0
HugePages_Free: 0
HugePages_Rsvd: 0
HugePages_Surp: 0
Hugepagesize: 2048 kB
DirectMap4k:  40960 kB
DirectMap2M:  4190208 kB
Linux#
```

### Running Bash from CLI

The following example shows how to run a bash command from the CLI with the `run bash <command>` command:

```
switch# run bash ps -el
F S  UID  PID  PPID  C  PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0    1    0  0  80   0  -   497 select ?      00:00:08 init
5 S   0    2    0  0  75  -5  -    0 kthrea ?      00:00:00 kthreadd
1 S   0    3    2  0 -40   -  -    0 migrat ?      00:00:00 migration/0
1 S   0    4    2  0  75  -5  -    0 ksofti ?      00:00:01 ksoftirqd/0
5 S   0    5    2  0  58   -  -    0 watchd ?      00:00:00 watchdog/0
1 S   0    6    2  0 -40   -  -    0 migrat ?      00:00:00 migration/1
1 S   0    7    2  0  75  -5  -    0 ksofti ?      00:00:00 ksoftirqd/1
5 S   0    8    2  0  58   -  -    0 watchd ?      00:00:00 watchdog/1
1 S   0    9    2  0 -40   -  -    0 migrat ?      00:00:00 migration/2
1 S   0   10    2  0  75  -5  -    0 ksofti ?      00:00:00 ksoftirqd/2
5 S   0   11    2  0  58   -  -    0 watchd ?      00:00:00 watchdog/2
1 S   0   12    2  0 -40   -  -    0 migrat ?      00:00:00 migration/3
1 S   0   13    2  0  75  -5  -    0 ksofti ?      00:00:00 ksoftirqd/3
5 S   0   14    2  0  58   -  -    0 watchd ?      00:00:00 watchdog/3

...

4 S   0  8864    1  0  80   0  -   2249 wait  ttyS0    00:00:00 login
4 S  2002 28073  8864  0  80   0  -   69158 select ttyS0    00:00:00 vsh
4 R   0 28264  3782  0  80   0  -   54790 select ?      00:00:00 in.dcos-telnet
4 S   0 28265 28264  0  80   0  -   2247 wait  pts/0    00:00:00 login
4 S  2002 28266 28265  0  80   0  -   69175 wait  pts/0    00:00:00 vsh
1 S  2002 28413 28266  0  80   0  -   69175 wait  pts/0    00:00:00 vsh
0 R  2002 28414 28413  0  80   0  -    887  -     pts/0    00:00:00 ps

switch#
```

## Running Python from Bash

The following example shows how to load Python and configure a switch using Python objects:

```
switch# run bash
Linux# python
Python 2.7.5 (default, May 16 2014, 10:58:01)
[GCC 4.3.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Loaded cisco NxOS lib!
>>>
>>> from cisco import *
>>> from cisco.vrf import *
>>> from cisco.interface import *
>>> vrfobj=VRF('myvrf')
>>> vrfobj.get_name()
'myvrf'
>>> vrfobj.add_interface('Ethernet1/3')
True
>>> intf=Interface('Ethernet1/3')
>>> print intf.config()

!Command: show running-config interface Ethernet1/3
!Time: Thu Aug 21 23:32:25 2014

version 6.0(2)U4(1)

interface Ethernet1/3
 no switchport
 vrf member myvrf

>>>
```