Bash

This chapter contains the following topics:

About Bash

In addition to the NX-OS CLI, Cisco Nexus 3400-S platform switches 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.

Guidelines and Limitations

The Bash shell has the following guidelines and limitations:

- The binaries that are located in the `/isan` folder are meant to be run in an environment which is set up differently from the environment of the shell that is entered from the **run bash** command. It is advisable not to use these binaries from the Bash shell as the behavior within this environment is not predictable.
- When importing Cisco Python modules, do not use Python from the Bash shell. Instead use the more recent Python in NX-OS VSH.
- Some processes and **show** commands can cause a large amount of output. If you are running scripts, and need to terminate long-running output, use Ctrl+C (not Ctrl+Z) to terminate the command output. If you use Ctrl+Z, a SIGCONT (signal continuation) message can be generated, which can cause the script to halt. Scripts that are halted through SIGCONT messages require user intervention to resume operation.

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?
  run         Execute/run program
  run-script  Run shell scripts

switch# run bash?
  bash  Linux-bash

switch# run bash
bash-4.2$ whoami
admin
bash-4.2$ pwd
/bootflash/home/admin
bash-4.2$
```

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

For instance, you can run **whoami** using **run bash** *command*:

```
run bash whoami
```

You can also run Bash by configuring the user **shelltype**:

```
username foo shelltype bash
```

This command puts you directly into the Bash shell upon login. This does not require **feature bash-shell** to be enabled.

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.

NX-OS network administrator users must escalate to root to pass configuration commands to the NX-OS VSH if:

- The NX-OS user has a shell-type Bash and logs into the switch with a shell-type Bash.
- The NX-OS user that logged into the switch in Bash continues to use Bash on the switch.

Run **sudo su 'vsh -c "<configuration commands>"'** or **sudo bash -c 'vsh -c "<configuration commands>"'**.

The following example demonstrates the network-administrator user MyUser with a default shell type Bash using **sudo** to pass configuration commands to the NX-OS.

```
ssh -l MyUser 1.2.3.4
-bash-4.2$ sudo vsh -c "configure terminal ; interface eth1/2 ; shutdown ; sleep 2 ; show
interface eth1/2 brief"

--------------------------------------------------------------------------------
Ethernet      VLAN    Type Mode   Status  Reason                   Speed     Port
Interface                                                                    Ch #
--------------------------------------------------------------------------------
Eth1/2        --      eth  routed down    Administratively down      auto(D) --
```

The following example demonstrates the network-administrator user MyUser with default shell type Bash entering the NX-OS and then running Bash on the NX-OS.

```
ssh -l MyUser 1.2.3.4
-bash-4.2$ vsh -h
Cisco Nexus Operating System (NX-OS) Software
TAC support: http://www.cisco.com/tac
```

```
Copyright (C) 2002-2019, Cisco and/or its affiliates.
All rights reserved.
The copyrights to certain works contained in this software are
owned by other third parties and used and distributed under their own
licenses, such as open source.  This software is provided "as is," and unless
otherwise stated, there is no warranty, express or implied, including but not
limited to warranties of merchantability and fitness for a particular purpose.
Certain components of this software are licensed under
the GNU General Public License (GPL) version 2.0 or
GNU General Public License (GPL) version 3.0  or the GNU
Lesser General Public License (LGPL) Version 2.1 or
Lesser General Public License (LGPL) Version 2.0.
A copy of each such license is available at
http://www.opensource.org/licenses/gpl-2.0.php and
http://opensource.org/licenses/gpl-3.0.html and
http://www.opensource.org/licenses/lgpl-2.1.php and
http://www.gnu.org/licenses/old-licenses/library.txt.

switch# run bash
bash-4.2$ vsh -c "configure terminal ; interface eth1/2 ; shutdown ; sleep 2 ; show interface
 eth1/2 brief"

--------------------------------------------------------------------------------
Ethernet      VLAN    Type Mode   Status  Reason                   Speed     Port
Interface                                                                    Ch #
--------------------------------------------------------------------------------
Eth1/2        --      eth  routed down    Administratively down      auto(D) --
```

Note Do not use **sudo su -** or the system hangs.

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

```
switch# run bash
bash-4.2$ sudo su root
bash-4.2# whoami
root
bash-4.2# exit
exit
```

Examples of Bash Commands

This section contains examples of Bash commands and output.

Displaying System Statistics

The following example displays system statistics:

```
switch# run bash
bash-4.2$ cat /proc/meminfo
MemTotal:       32827712 kB
MemFree:        27429772 kB
MemAvailable:   28004236 kB
Buffers:           54296 kB
Cached:          2863648 kB
SwapCached:            0 kB
```

```
Active:          1993452 kB
Inactive:        2616472 kB
Active(anon):    1812124 kB
Inactive(anon):  2192904 kB
Active(file):     181328 kB
Inactive(file):   423568 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:             0 kB
SwapFree:              0 kB
Dirty:                24 kB
Writeback:             0 kB
AnonPages:       1691732 kB
Mapped:           578756 kB
Shmem:           2313336 kB
Slab:             248788 kB
SReclaimable:      53660 kB
SUnreclaim:       195128 kB
KernelStack:       11520 kB
PageTables:        58812 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:    16413856 kB
Committed_AS:   23471740 kB
VmallocTotal:   34359738367 kB
VmallocUsed:      579308 kB
VmallocChunk:   34358945788 kB
HardwareCorrupted:     0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
DirectMap4k:       26588 kB
DirectMap2M:     1998848 kB
DirectMap1G:    33554432 kB
bash-4.3#
```

Running Bash from CLI

The following example runs **ps** from Bash using **run bash** *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 -   528 poll_s ?        00:00:03 init
1 S     0     2     0  0  80   0 -     0 kthrea ?        00:00:00 kthreadd
1 S     0     3     2  0  80   0 -     0 run_ks ?        00:00:56 ksoftirqd/0
1 S     0     6     2  0 -40   - -     0 cpu_st ?        00:00:00 migration/0
1 S     0     7     2  0 -40   - -     0 watchd ?        00:00:00 watchdog/0
1 S     0     8     2  0 -40   - -     0 cpu_st ?        00:00:00 migration/1
1 S     0     9     2  0  80   0 -     0 worker ?        00:00:00 kworker/1:0
1 S     0    10     2  0  80   0 -     0 run_ks ?        00:00:00 ksoftirqd/1
```

Managing Feature RPMs

RPM Installation Prerequisites

Use these procedures to verify that the system is ready before installing or adding an RPM.

Procedure

	Command or Action	Purpose
Step 1	switch# **show logging logfile | grep -i "System ready"**	Before running Bash, this step verifies that the system is ready before installing or adding an RPM. Proceed if you see output similar to the following: **2019 Apr 18 17:24:22 switch %ASCII-CFG-2-CONF_CONTROL: System ready**

Step 2	switch# **run bash sudo su**
	Example:
	`switch# run bash sudo su`
	`bash-4.2#`
		Loads Bash.

Installing Feature RPMs from Bash

Procedure

	Command or Action	Purpose
Step 1	**sudo yum installed | grep** *platform*	Displays a list of the NX-OS feature RPMs installed on the switch.
Step 2	**yum list available**	Displays a list of the available RPMs.
Step 3	**sudo yum -y install** *rpm*	Installs an available RPM.

Example

The following is an example of installing the **bfd** RPM:

```
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
lacp.lib32_n9000                        1.0.0-r0                        installed
linecard.lib32_n9000                    1.0.0-r0                        installed
lldp.lib32_n9000                        1.0.0-r0                        installed
ntp.lib32_n9000                         1.0.0-r0                        installed
nxos-ssh.lib32_n9000                    1.0.0-r0                        installed
ospf.lib32_n9000                        1.0.0-r0                        installed
perf-cisco.n9000_gdb                    3.12-r0                         installed
platform.lib32_n9000                    1.0.0-r0                        installed
shadow-securetty.n9000_gdb              4.1.4.3-r1                      installed
snmp.lib32_n9000                        1.0.0-r0                        installed
svi.lib32_n9000                         1.0.0-r0                        installed
sysvinit-inittab.n9000_gdb              2.88dsf-r14                     installed
tacacs.lib32_n9000                      1.0.0-r0                        installed
task-nxos-base.n9000_gdb                1.0-r0                          installed
tor.lib32_n9000                         1.0.0-r0                        installed
vtp.lib32_n9000                         1.0.0-r0                        installed
bash-4.2$ yum list available
bgp.lib32_n9000                         1.0.0-r0
bash-4.2$ sudo yum -y install bfd
```

Note Upon switch reload during boot up, use the **rpm** command instead of **yum** for persistent RPMs. Otherwise, RPMs initially installed using **yum bash** or **install cli** shows `reponame` or `filename` instead of `installed`.

Upgrading Feature RPMs

Before you begin

There must be a higher version of the RPM in the Yum repository.

Procedure

	Command or Action	Purpose
Step 1	**sudo yum -y upgrade** *rpm*	Upgrades an installed RPM.

Example

The following is an example of upgrading the **bfd** RPM:

```
bash-4.2$ sudo yum -y upgrade bfd
```

Downgrading a Feature RPM

Procedure

	Command or Action	Purpose
Step 1	**sudo yum -y downgrade** *rpm*	Downgrades the RPM if any of the Yum repositories has a lower version of the RPM.

Example

The following example shows how to downgrade the **bfd** RPM:

```
bash-4.2$ sudo yum -y downgrade bfd
```

Erasing a Feature RPM

Note The SNMP RPM and the NTP RPM are protected and cannot be erased.

You can upgrade or downgrade these RPMs. It requires a system reload for the upgrade or downgrade to take effect.

For the list of protected RPMs, see `/etc/yum/protected.d/protected_pkgs.conf`.

Procedure

	Command or Action	Purpose
Step 1	**sudo yum -y erase** *rpm*	Erases the RPM.

Example

The following example shows how to erase the **bfd** RPM:

```
bash-4.2$ sudo yum -y erase bfd
```

Managing Patch RPMs

RPM Installation Prerequisites

Use these procedures to verify that the system is ready before installing or adding an RPM.

Procedure

	Command or Action	Purpose
Step 1	switch# **show logging logfile | grep -i "System ready"**	Before running Bash, this step verifies that the system is ready before installing or adding an RPM. Proceed if you see output similar to the following: **2019 Apr 18 17:24:22 switch %ASCII-CFG-2-CONF_CONTROL: System ready**

Step 2	switch# **run bash sudo su**
	Example:
	`switch# run bash sudo su`
	`bash-4.2#`
		Loads Bash.

Adding Patch RPMs from Bash

Procedure

	Command or Action	Purpose
Step 1	**yum list --patch-only**	Displays a list of the patch RPMs present on the switch.
Step 2	**sudo yum install --add** *URL_of_patch*	Adds the patch to the repository, where *URL_of_patch* is a well-defined format, such as **bootflash:/*patch***, not in standard Linux format, such as **/bootflash/*patch***.
Step 3	**yum list --patch-only available**	Displays a list of the patches that are added to the repository but are in an inactive state.

Example

The following is an example of installing the **nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000** RPM:

```
bash-4.2# yum list --patch-only
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                              | 1.1 kB     00:00 ...
localdb                                                  |  951 B     00:00 ...
patching                                                 |  951 B     00:00 ...
thirdparty                                               |  951 B     00:00 ...
bash-4.2#
bash-4.2# sudo yum install --add bootflash:/nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000.rpm
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
```

```
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
[##############     ] 70%Install operation 135 completed successfully at Tue Mar 26 17:45:34
 2019.

[####################] 100%
bash-4.2#
```

Once the patch RPM is installed, verify that it was installed properly. The following command lists the patches that are added to the repository and are in the inactive state:

```
bash-4.2# yum list --patch-only available
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
nxos.CSCab00001-n9k_ALL.lib32_n9000    1.0.0        patching
bash-4.2#
```

You can also add patches to a repository from a tar file, where the RPMs are bundled in the tar file. The following example shows how to add two RPMs that are part of the nxos.CSCab00002_CSCab00003-n9k_ALL-1.0.0.lib32_n9000 tar file to the patch repository:

```
bash-4.2# sudo yum install --add
bootflash:/nxos.CSCab00002_CSCab00003-n9k_ALL-1.0.0.lib32_n9000.tar
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
[##############     ] 70%Install operation 146 completed successfully at Tue Mar 26 21:17:39
 2019.

[####################] 100%
bash-4.2#
bash-4.2# yum list --patch-only
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
patching/primary                                           |  942 B     00:00 ...
patching                                                                    2/2
thirdparty                                                 |  951 B     00:00 ...
nxos.CSCab00003-n9k_ALL.lib32_n9000    1.0.0              patching
nxos.CSCab00002-n9k_ALL.lib32_n9000    1.0.0              patching
bash-4.2#
```

Activating a Patch RPM

Before you begin

Verify that you have added the necessary patch RPM to the repository using the instructions in Adding Patch RPMs from Bash, on page 9.

Procedure

	Command or Action	Purpose
Step 1	**sudo yum install** *patch_RPM* **--nocommit**	Activates the patch RPM, where *patch_RPM* is a patch that is located in the repository. Do not provide a location for the patch in this step. **Note** Adding the --nocommit flag to the command means that the patch RPM is activated in this step, but not committed. See Committing a Patch RPM, on page 12 for instructions on committing the patch RPM after you have activated it.

Example

The following example shows how to activate the **nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000** patch RPM:

```
bash-4.2# sudo yum install nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000 --nocommit
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package nxos.CSCab00001-n9k_ALL.lib32_n9000 0:1.0.0 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                    Arch           Version               Repository   Size
================================================================================
Installing:
 nxos.CSCab00001-n9k_ALL    lib32_n9000    1.0.0                 patching     28 k

Transaction Summary
================================================================================
Install       1 Package

Total download size: 28 k
Installed size: 82 k
Is this ok [y/N]: y
```

```
Downloading Packages:
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000            1/1
[##################  ] 90%error: reading
/var/sysmgr/tmp/patches/CSCab00001-n9k_ALL/isan/bin/sysinfo manifest, non-printable characters
 found

Installed:
  nxos.CSCab00001-n9k_ALL.lib32_n9000 0:1.0.0

Complete!
Install operation 140 completed successfully at Tue Mar 27 18:07:40 2018.

[####################] 100%
bash-4.2#
```

Enter the following command to verify that the patch RPM was activated successfully:

```
bash-4.2# yum list --patch-only
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
nxos.CSCab00001-n9k_ALL.lib32_n9000    1.0.0              installed
bash-4.2#
```

Committing a Patch RPM

Procedure

	Command or Action	Purpose
Step 1	**sudo yum install** *patch_RPM* **--commit**	Commits the patch RPM. The patch RPM must be committed to keep it active after reloads.

Example

The following example shows how to commit the **nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000** patch RPM:

```
bash-4.2# sudo yum install nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000 --commit
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
Install operation 142 completed successfully at Tue Mar 27 18:13:16 2018.

[####################] 100%
bash-4.2#
```

Enter the following command to verify that the patch RPM was committed successfully:

```
bash-4.2# yum list --patch-only committed
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
nxos.CSCab00001-n9k_ALL.lib32_n9000    1.0.0              installed
bash-4.2#
```

Deactivating a Patch RPM

Procedure

	Command or Action	Purpose
Step 1	**sudo yum erase** *patch_RPM* **--nocommit**	Deactivates the patch RPM. **Note** Adding the --nocommit flag to the command means that the patch RPM is only deactivated in this step.
Step 2	**sudo yum install** *patch_RPM* **--commit**	Commits the patch RPM. You will get an error message if you try to remove the patch RPM without first committing it.

Example

The following example shows how to deactivate the **nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000** patch RPM:

```
bash-4.2# sudo yum erase nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000 --nocommit
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package nxos.CSCab00001-n9k_ALL.lib32_n9000 0:1.0.0 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                     Arch           Version              Repository    Size
================================================================================
Removing:
 nxos.CSCab00001-n9k_ALL    lib32_n9000    1.0.0                @patching     82 k

Transaction Summary
================================================================================
Remove        1 Package

Installed size: 82 k
Is this ok [y/N]: y
Downloading Packages:
```

```
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
[######              ] 30%error: reading
/var/sysmgr/tmp/patches/CSCab00001-n9k_ALL/isan/bin/sysinfo manifest, non-printable characters
 found
  Erasing    : nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000                        1/1
[##################  ] 90%
Removed:
  nxos.CSCab00001-n9k_ALL.lib32_n9000 0:1.0.0

Complete!
Install operation 143 completed successfully at Tue Mar 27 21:03:47 2018.

[####################] 100%
bash-4.2#
```

You must commit the patch RPM after deactivating it. If you do not commit the patch RPM after deactivating it, you will get an error message if you try to remove the patch RPM using the instructions in Removing a Patch RPM, on page 14.

```
bash-4.2# sudo yum install nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000 --commit
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
Install operation 144 completed successfully at Tue Mar 27 21:09:28 2018.

[####################] 100%
bash-4.2#
```

Enter the following command to verify that the patch RPM has been committed successfully:

```
bash-4.2# yum list --patch-only
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
nxos.CSCab00001-n9k_ALL.lib32_n9000    1.0.0               patching
bash-4.2#
```

Removing a Patch RPM

Procedure

	Command or Action	Purpose
Step 1	**sudo yum install --remove** *patch_RPM*	Removes an inactive patch RPM.

Example

The following example shows how to remove the **nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000** patch RPM:

```
bash-4.2# sudo yum install --remove nxos.CSCab00001-n9k_ALL-1.0.0.lib32_n9000
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
[##########        ] 50%Install operation 145 completed successfully at Tue Mar 27 21:11:05
 2018.

[####################] 100%
bash-4.2#
```

Note If you see the following error message after attempting to remove the patch RPM:

Install operation 11 "failed because patch was not committed". at Wed Mar 28 22:14:05 2018

Then you did not commit the patch RPM before attempting to remove it. See Deactivating a Patch RPM, on page 13 for instructions on committing the patch RPM before attempting to remove it.

Enter the following command to verify that the inactive patch RPM was removed successfully:

```
bash-4.2# yum list --patch-only
Loaded plugins: downloadonly, importpubkey, localrpmDB, patchaction, patching,
              : protect-packages
groups-repo                                                | 1.1 kB     00:00 ...
localdb                                                    |  951 B     00:00 ...
patching                                                   |  951 B     00:00 ...
patching/primary                                           |  197 B     00:00 ...
thirdparty                                                 |  951 B     00:00 ...
bash-4.2#
```

Persistently Daemonizing an SDK- or ISO-Built Third-Party Process

Your application should have a startup Bash script that gets installed in `/etc/init.d/`*`application_name`*. This startup Bash script should have the following general format. For more information about this format, see http://linux.die.net/man/8/chkconfig.

```
#!/bin/bash
#
# <application_name> Short description of your application
#
# chkconfig: 2345 15 85
# description: Short description of your application
#
### BEGIN INIT INFO
# Provides: <application_name>
```

```
# Required-Start: $local_fs $remote_fs $network $named
# Required-Stop: $local_fs $remote_fs $network
# Description: Short description of your application
### END INIT INFO
# See how we were called.
case "$1" in
start)
# Put your startup commands here
# Set RETVAL to 0 for success, non-0 for failure
;;
stop)
# Put your stop commands here
# Set RETVAL to 0 for success, non-0 for failure
;;
status)
# Put your status commands here
# Set RETVAL to 0 for success, non-0 for failure
;;
restart|force-reload|reload)
# Put your restart commands here
# Set RETVAL to 0 for success, non-0 for failure
;;
*)
echo $"Usage: $prog {start|stop|status|restart|force-reload}"
RETVAL=2
esac

exit $RETVAL
```

Persistently Starting Your Application from the Native Bash Shell

Procedure

Step 1 Install your application startup Bash script that you created into `/etc/init.d/`*application_name*

Step 2 Start your application with `/etc/init.d/`*application_name* start

Step 3 Enter **chkconfig --add** *application_name*

Step 4 Enter **chkconfig --level 3** *application_name* **on**

Run level 3 is the standard multi-user run level, and the level at which the switch normally runs.

Step 5 Verify that your application is scheduled to run on level 3 by running **chkconfig --list** *application_name* and confirm that level 3 is set to on

Step 6 Verify that your application is listed in `/etc/rc3.d`. You should see something like this, where there is an 'S' followed by a number, followed by your application name (`tcollector` in this example), and a link to your Bash startup script in `../init.d/`*application_name*

bash-4.2# ls -l /etc/rc3.d/**tcollector**

lrwxrwxrwx 1 root root 20 Sep 25 22:56 /etc/rc3.d/S15tcollector -> ../init.d/tcollector

bash-4.2#

Synchronize Files from Active Bootflash to Standby Bootflash

Cisco Nexus 3400-S platform switches are generally configured with two supervisor modules to provide high availability (one active supervisor module and one standby supervisor module). Each supervisor module has its own bootflash file system for file storage, and the Active and Standby bootflash file systems are generally independent of each other. If there is a need for specific content on the active bootflash, that same content is probably also needed on the standby bootflash in case there is a switchover at some point.

Certain files and directories on the active supervisor module, or active bootflash (`/bootflash`), can be automatically synchronized to the standby supervisor module, or standby bootflash (`/bootflash_sup-remote`), if the standby supervisor module is up and available. You can select the files and directories to be synchronized by loading Bash on your switch, then adding the files and directories that you want to have synchronized from the active bootflash to the standby bootflash into the editable file `/bootflash/bootflash_sync_list`.

For example:

```
switch# run bash
bash-4.2# echo "/bootflash/home/admin" | sudo tee --append /bootflash/bootflash_sync_list
bash-4.2# echo "/bootflash/nxos.5.bin" | sudo tee --append /bootflash/bootflash_sync_list
bash-4.2# cat /bootflash/bootflash_sync_list
/bootflash/home/admin
/bootflash/nxos.5.bin
```

When changes are made to the files or directories on the active bootflash, these changes are automatically synchronized to standby bootflash, if the standby bootflash is up and available. If the standby bootflash is rebooted, either as a regular boot, switchover or manual standby reload, a catch-up synchronization of changes to the active bootflash is pushed out to the standby bootflash, once the standby supervisor comes online.

Following are the characteristics and restrictions for the editable `/bootflash/bootflash_sync_list` file:

- The `/bootflash/bootflash_sync_list` file is automatically created on the first run and is empty at that initial creation state.
- Entries in the `/bootflash/bootflash_sync_list` file follow these guidelines:
  - One entry per line
  - Entries are given as Linux paths (for example, `/bootflash/img.bin`)
  - Entries must be within the `/bootflash` file system
- The `/bootflash/bootflash_sync_list` file itself is automatically synchronized to the standby bootflash. You can also manually copy the `/bootflash/bootflash_sync_list` file to or from the supervisor module using the **copy** virtual shell (VSH) command.
- You can edit the `/bootflash/bootflash_sync_list` file directly on the supervisor module with the following command:

  ```
  run bash vi /bootflash/bootflash_sync_list
  ```

All output from the synchronization event is redirected to the log file `/var/tmp/bootflash_sync.log`. You can view or tail this log file using either of the following commands:

```
run bash less /var/tmp/bootflash_sync.log
```

```
run bash tail -f /var/tmp/bootflash_sync.log
```

The synchronization script will not delete files from the standby bootflash directories unless it explicitly receives a delete event for the corresponding file on the active bootflash directories. Sometimes, the standby bootflash might have more used space than the active bootflash, which results in the standby bootflash running out of space when the active bootflash is synchronizing to it. To make the standby bootflash an exact mirror of the active bootflash (to delete any extra files on the standby bootflash), enter the following command:

```
run bash sudo rsync -a --delete /bootflash/ /bootflash_sup-remote/
```

The synchronization script should continue to run in the background without crashing or exiting. However, if it does stop running for some reason, you can manually restart it using the following command:

```
run bash sudo /isan/etc/rc.d/rc.isan-start/S98bootflash_sync.sh start
```

An Example Application in the Native Bash Shell

The following example demonstrates an application in the Native Bash Shell:

```
bash-4.2# cat /etc/init.d/hello.sh
#!/bin/bash

PIDFILE=/tmp/hello.pid
OUTPUTFILE=/tmp/hello

echo $$ > $PIDFILE
rm -f $OUTPUTFILE
while true
do
    echo $(date) >> $OUTPUTFILE
    echo 'Hello World' >> $OUTPUTFILE
    sleep 10
done
bash-4.2#
bash-4.2#
bash-4.2# cat /etc/init.d/hello
#!/bin/bash
#
# hello Trivial "hello world" example Third Party App
#
# chkconfig: 2345 15 85
# description: Trivial example Third Party App
#
### BEGIN INIT INFO
# Provides: hello
# Required-Start: $local_fs $remote_fs $network $named
# Required-Stop: $local_fs $remote_fs $network
# Description: Trivial example Third Party App
### END INIT INFO

PIDFILE=/tmp/hello.pid

# See how we were called.
case "$1" in
start)
    /etc/init.d/hello.sh &
```

```
    RETVAL=$?
;;
stop)
    kill -9 `cat $PIDFILE`
    RETVAL=$?
;;
status)
    ps -p `cat $PIDFILE`
    RETVAL=$?
;;
restart|force-reload|reload)
    kill -9 `cat $PIDFILE`
    /etc/init.d/hello.sh &
    RETVAL=$?
;;
*)
echo $"Usage: $prog {start|stop|status|restart|force-reload}"
RETVAL=2
esac

exit $RETVAL
bash-4.2#
bash-4.2# chkconfig --add hello
bash-4.2# chkconfig --level 3 hello on
bash-4.2# chkconfig --list hello
hello           0:off   1:off   2:on    3:on    4:on    5:on    6:off
bash-4.2# ls -al /etc/rc3.d/*hello*
lrwxrwxrwx 1 root root 15 Sep 27 18:00 /etc/rc3.d/S15hello -> ../init.d/hello
bash-4.2#
bash-4.2# reboot
```

After reload

```
bash-4.2# ps -ef | grep hello
root      8790     1  0 18:03 ?        00:00:00 /bin/bash /etc/init.d/hello.sh
root      8973  8775  0 18:04 ttyS0    00:00:00 grep hello
bash-4.2#
bash-4.2# ls -al /tmp/hello*
-rw-rw-rw- 1 root root 205 Sep 27 18:04 /tmp/hello
-rw-rw-rw- 1 root root   5 Sep 27 18:03 /tmp/hello.pid
bash-4.2# cat /tmp/hello.pid
8790
bash-4.2# cat /tmp/hello
Sun Sep 27 18:03:49 UTC 2015
Hello World
Sun Sep 27 18:03:59 UTC 2015
Hello World
Sun Sep 27 18:04:09 UTC 2015
Hello World
Sun Sep 27 18:04:19 UTC 2015
Hello World
Sun Sep 27 18:04:29 UTC 2015
Hello World
Sun Sep 27 18:04:39 UTC 2015
Hello World
bash-4.2#
```