Requirements for Candidate Config/ Rollback config.

In order for us to implement candidate config or rollback, we require space on router. GNS3 emulator images don't have any disk, so here we are going to add PCMCIA disk0 slot to store our candidate config on Cisco 7200 router. We will need to configure few pre-requisites on the router as well. First we will add disk.

GNS3: Configure Cisco Router 7200

➔ Right click on the router and Select Configure as shown in Figure 1.

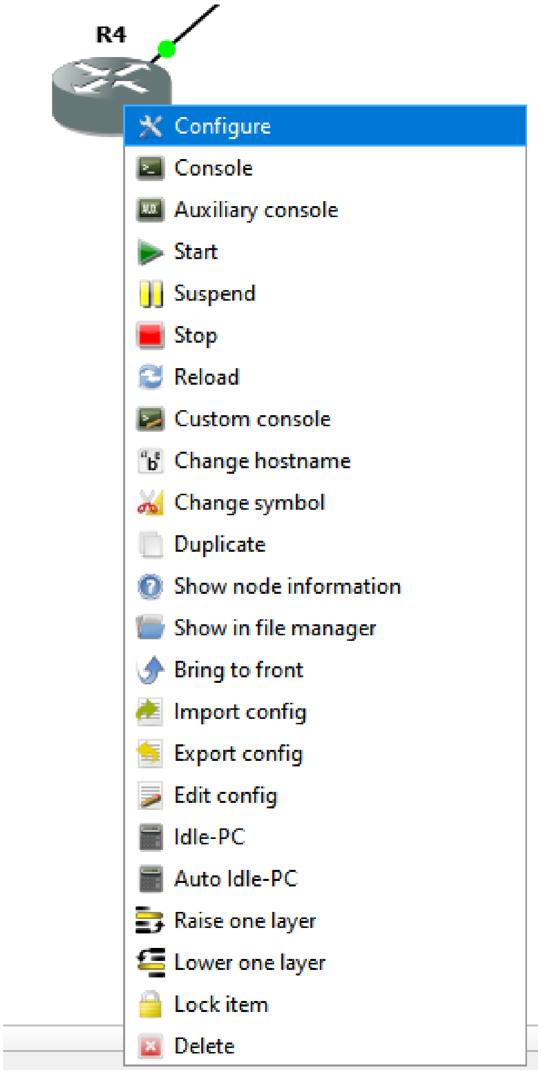
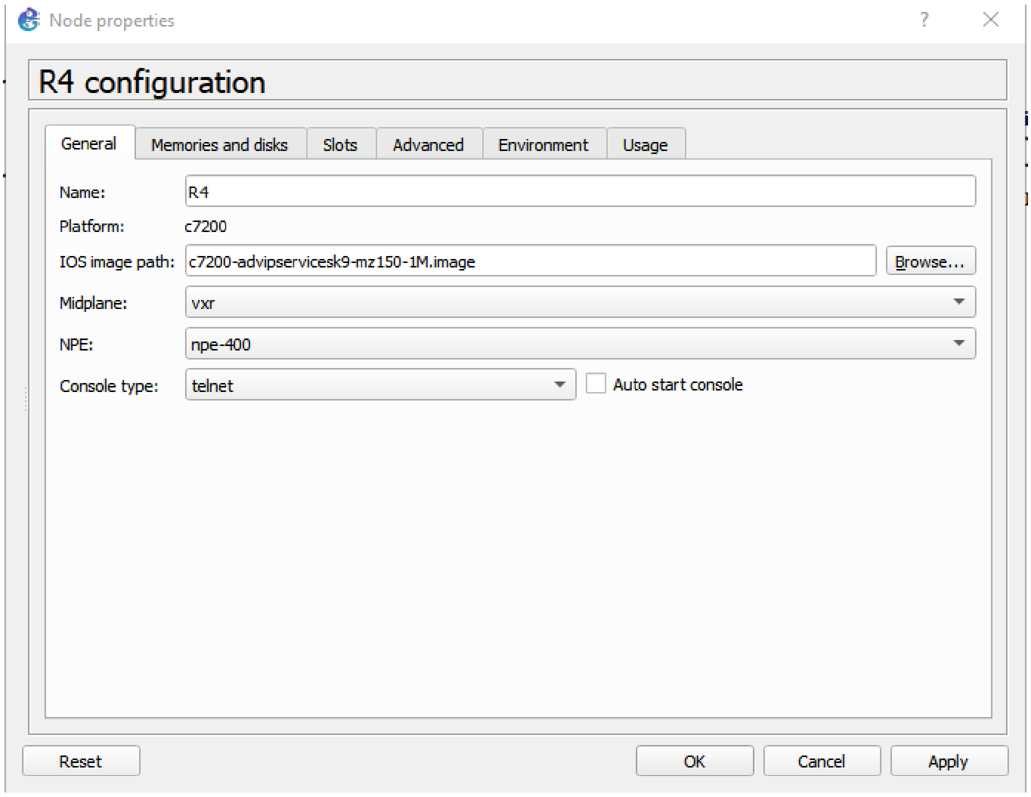


Figure 1 Router Configure

Figure 2 shows Router's General Configuration Dialogue box



R4 configuration

General | Memories and disks | Slots | Advanced | Environment | Usage

Name: R4

Platform: c7200

IOS image path: c7200-advipservicesk9-mz150-1M.image

Midplane: vxr

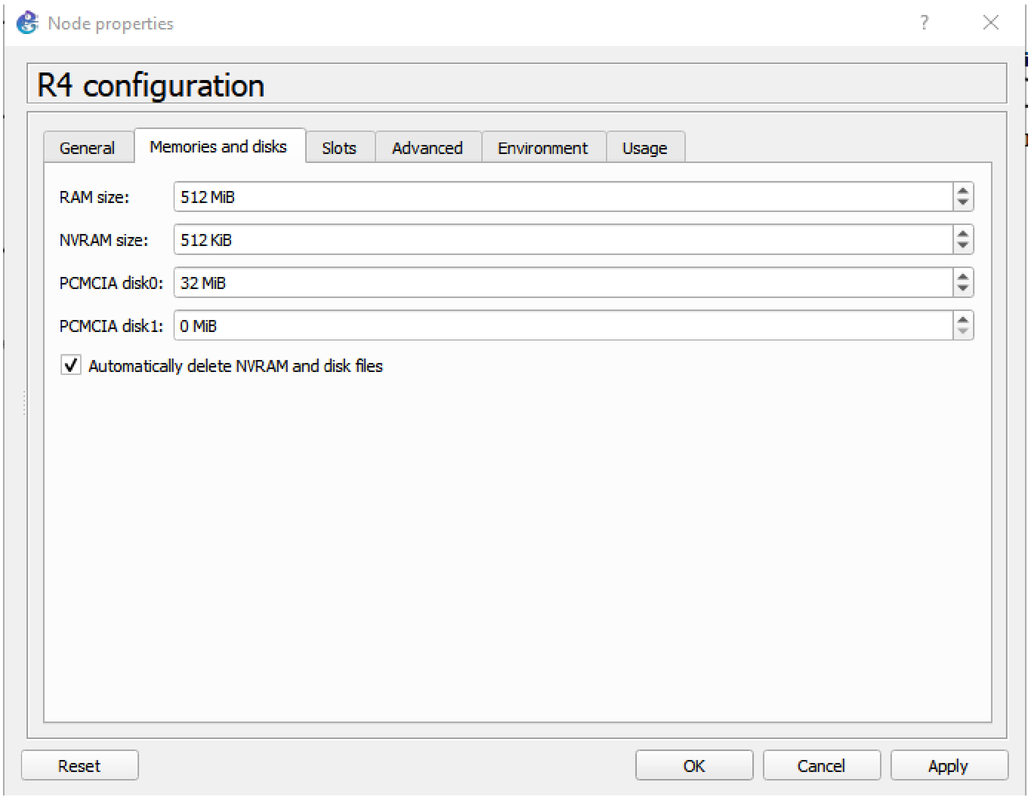
NPE: npe-400

Console type: telnet

□ Auto start console

Figure 2 General Configuration

➔ On Memories and disks, Select PCMCIA disk0, set 32MB



R4 configuration

General | Memories and disks | Slots | Advanced | Environment | Usage

RAM size: 512 MiB

NVRAM size: 512 KiB

PCMCIA disk0: 32 MiB

PCMCIA disk1: 0 MiB

☑ Automatically delete NVRAM and disk files

Figure 3 Configure PCMCIA Disk0

Click ok. Disk0 is now configured on the router to be used.

Prepare Cisco device for SCP/Candidate Config/Rollback Config

We must format the added file system on to the router. All operations would be performed by admin user on the router.

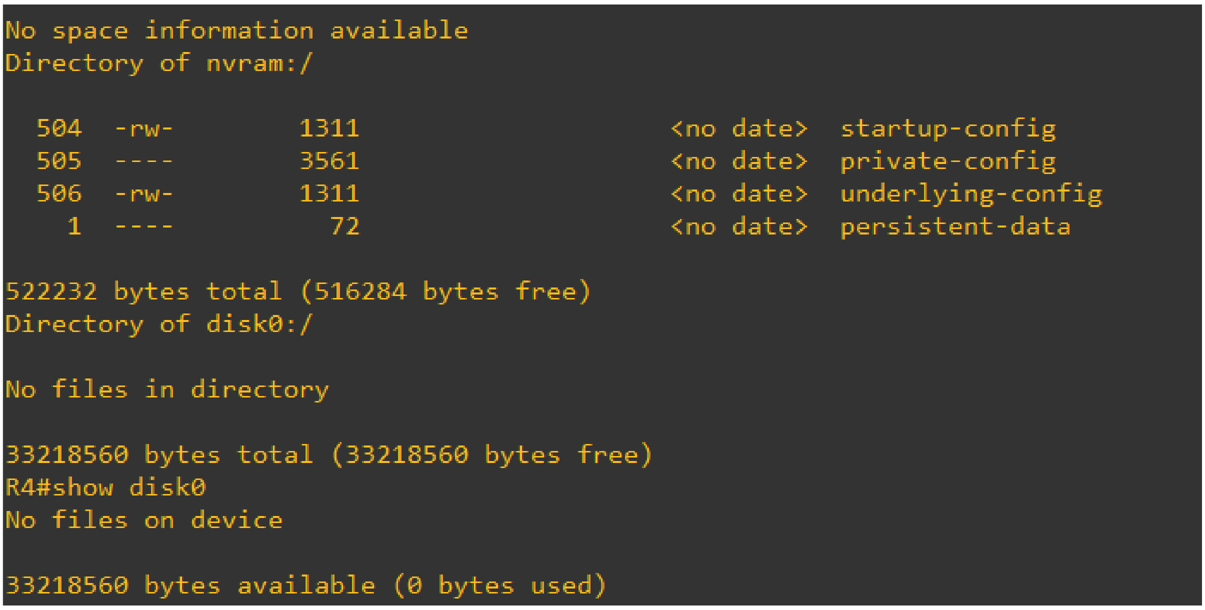
- ➔ User should be configured with privilege level 15
  R4(config)# username netlab privilege 15 secret netlab
- ➔ SSH server must be configured on the device to login via ssh.
- ➔ To enable SCP (Secure Copy) service on the router.
  R4(config)# ip scp server enable
- ➔ Configure AAA
  R4(config)#aaa new-model
  R4(config)#aaa authentication login default local
  R4(config)#aaa authorization exec default local none

To check the disk on the router, it is good practice as the disk could be numbered differently.

R4#show all-filesystem (show file systems – command can be used here too)

R4#show disk0

Output cropped for show all-filesystem command in Figure 4.



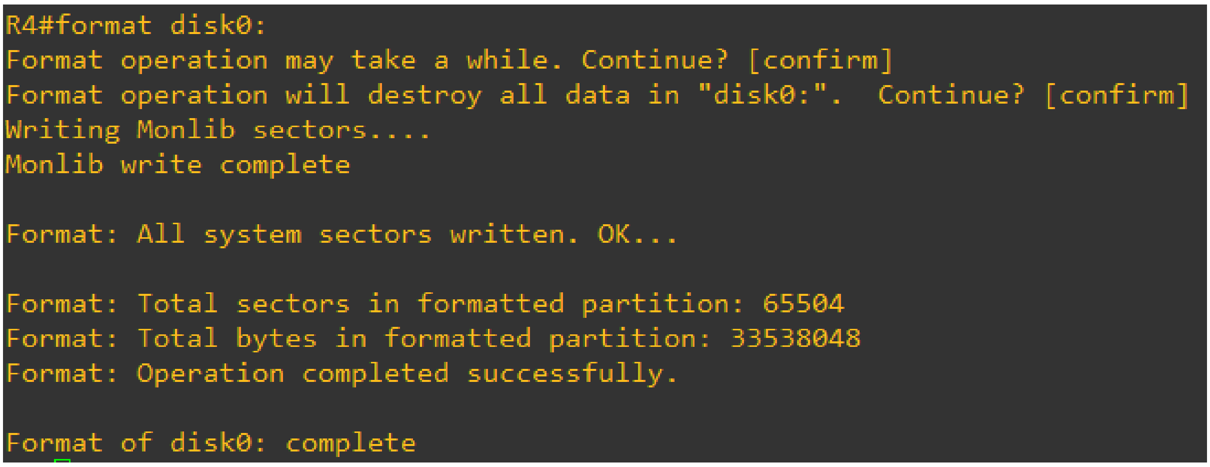
```
No space information available
Directory of nvram:/

  504  -rw-        1311                    <no date>  startup-config
  505  ----        3561                    <no date>  private-config
  506  -rw-        1311                    <no date>  underlying-config
    1  ----          72                    <no date>  persistent-data

522232 bytes total (516284 bytes free)
Directory of disk0:/

No files in directory

33218560 bytes total (33218560 bytes free)
R4#show disk0
No files on device

33218560 bytes available (0 bytes used)
```

Figure 4 Router Disc information

Format the disk0

R4#format disk0:

Figure 5 shows the operation formatting of the disk0 on the router.



```
R4#format disk0:
Format operation may take a while. Continue? [confirm]
Format operation will destroy all data in "disk0:".  Continue? [confirm]
Writing Monlib sectors....
Monlib write complete

Format: All system sectors written. OK...

Format: Total sectors in formatted partition: 65504
Format: Total bytes in formatted partition: 33538048
Format: Operation completed successfully.

Format of disk0: complete
```

Figure 5 Formatting Disk0

Confirm SSH is enabled

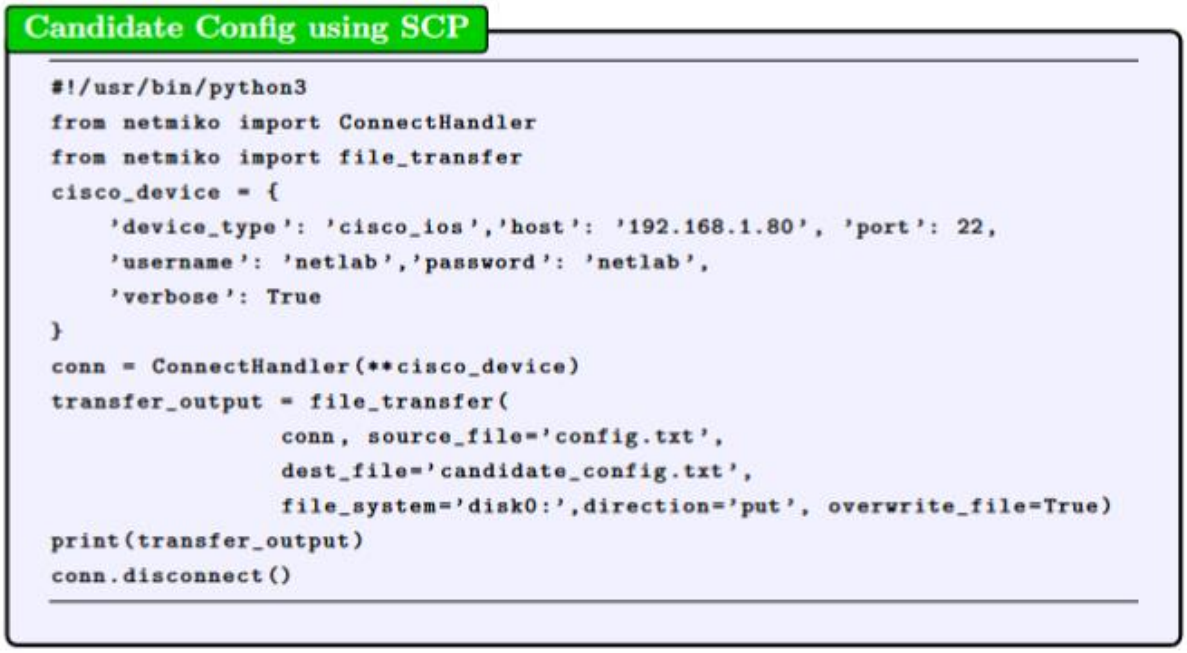
R4(config)#ip domain-name netlab.net
R4(config)#crypto key generate rsa 1024
R4(config)#ip ssh version 2
R4(config)#line vty 0 1869
R4(config)#transport input telnet ssh
R4(config)#login local

Router is now ready to be used for SCP(Secure Copy).

Secure Copy

Figure 6 shows the script to perform SCP from local desktop to router.

NAPALM uses scp for configuration management.

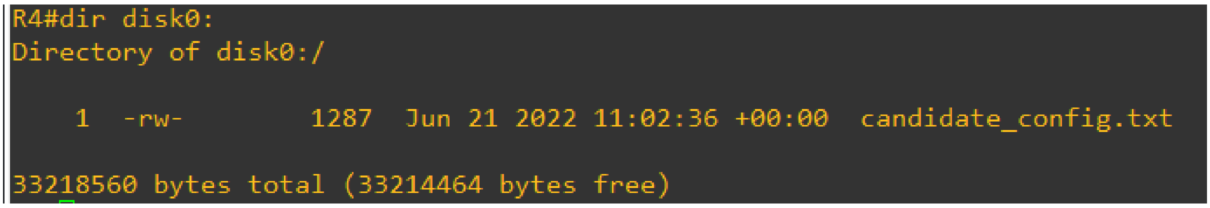


Candidate Config using SCP

```
#!/usr/bin/python3
from netmiko import ConnectHandler
from netmiko import file_transfer
cisco_device = {
    'device_type': 'cisco_ios','host': '192.168.1.80', 'port': 22,
    'username': 'netlab','password': 'netlab',
    'verbose': True
}
conn = ConnectHandler(**cisco_device)
transfer_output = file_transfer(
                conn, source_file='config.txt',
                dest_file='candidate_config.txt',
                file_system='disk0:',direction='put', overwrite_file=True)
print(transfer_output)
conn.disconnect()
```

Figure 6 Netmiko SCP Script

Result after running the script is shown in Figure 7.



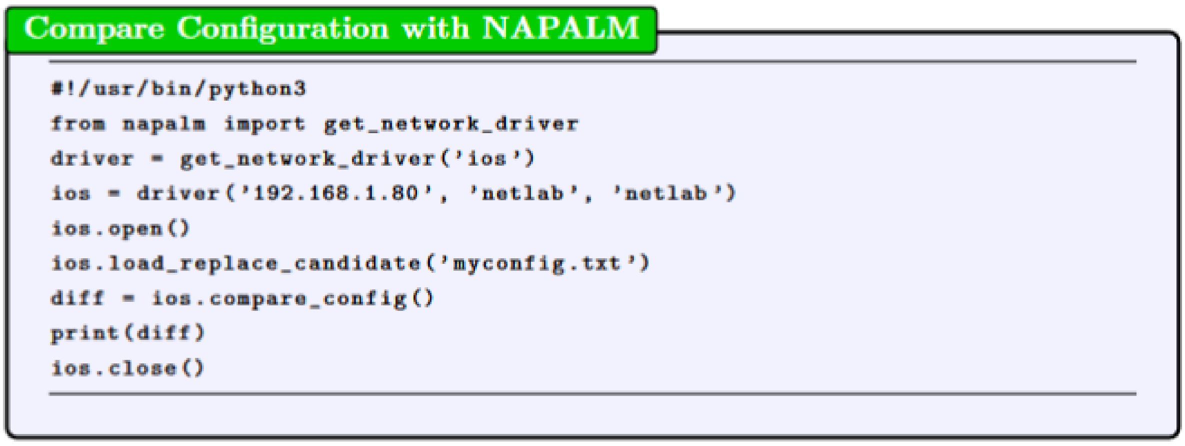
```
R4#dir disk0:
Directory of disk0:/

    1  -rw-        1287  Jun 21 2022 11:02:36 +00:00  candidate_config.txt

33218560 bytes total (33214464 bytes free)
```

Figure 7 SCP Config from Script

One of the most important thing to keep in mind while using NAPALM is, archive and path for the configuration should be given before proceeding.

R4(config)#archive
R4(config-archive)#path disk0:

Figure 8 shows sample script to check if there is any configuration change/difference in current configuration and configuration file(myconfig.txt).

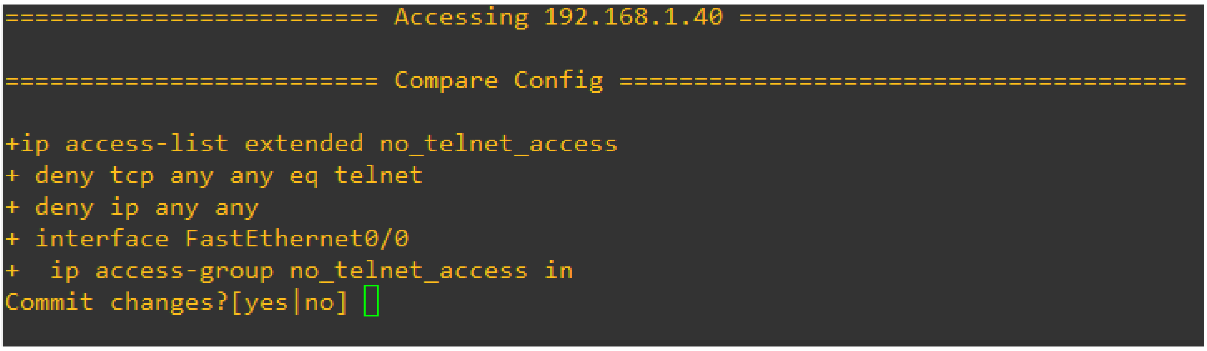


Compare Configuration with NAPALM

```
#!/usr/bin/python3
from napalm import get_network_driver
driver = get_network_driver('ios')
ios = driver('192.168.1.80', 'netlab', 'netlab')
ios.open()
ios.load_replace_candidate('myconfig.txt')
diff = ios.compare_config()
print(diff)
ios.close()
```

Figure 8 NAPALM Compare Configuration

If there are any configuration differences, It will display, shown in Figure 10.

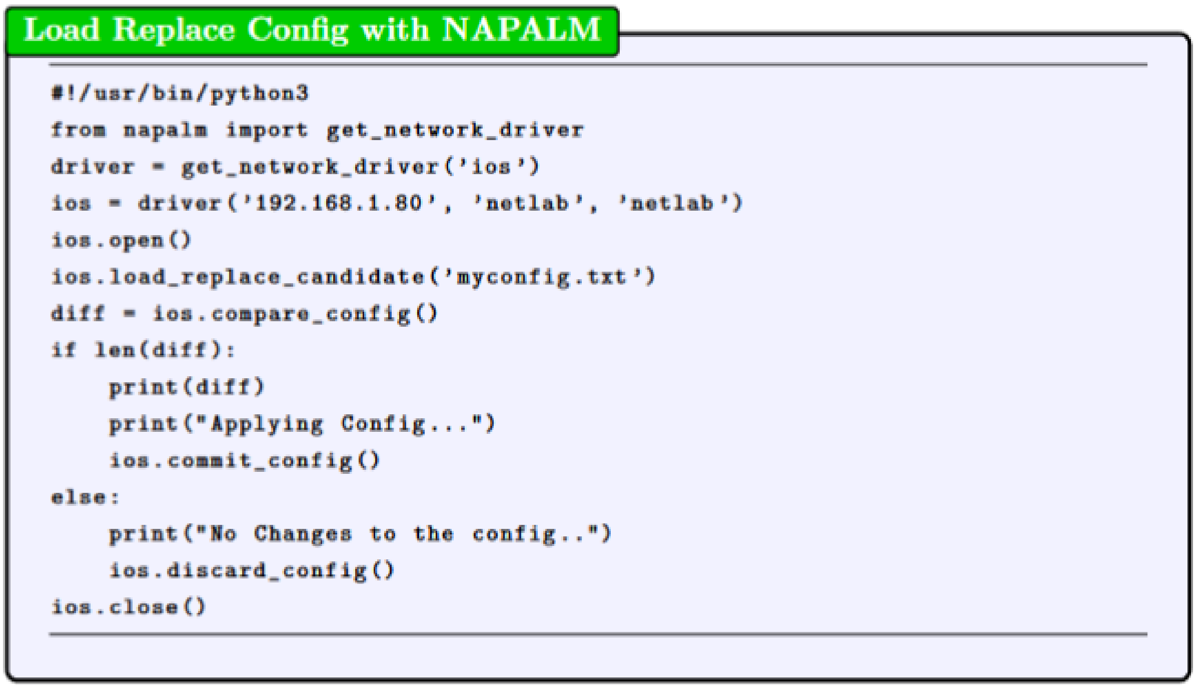


```
======================== Accessing 192.168.1.40 ==============================

======================== Compare Config ======================================

+ip access-list extended no_telnet_access
+ deny tcp any any eq telnet
+ deny ip any any
+ interface FastEthernet0/0
+  ip access-group no_telnet_access in
Commit changes?[yes|no] 
```

Figure 9 Compare config

Load Replace Configuration

NAPALM provides a mechanism to push configuration and replace the configuration entirely into the existing device config. we can create a script to replace entire current configuration. We can check for differences in configuration display it and if you are happy with it you can change it. Figure 10 shows sample script.



Load Replace Config with NAPALM

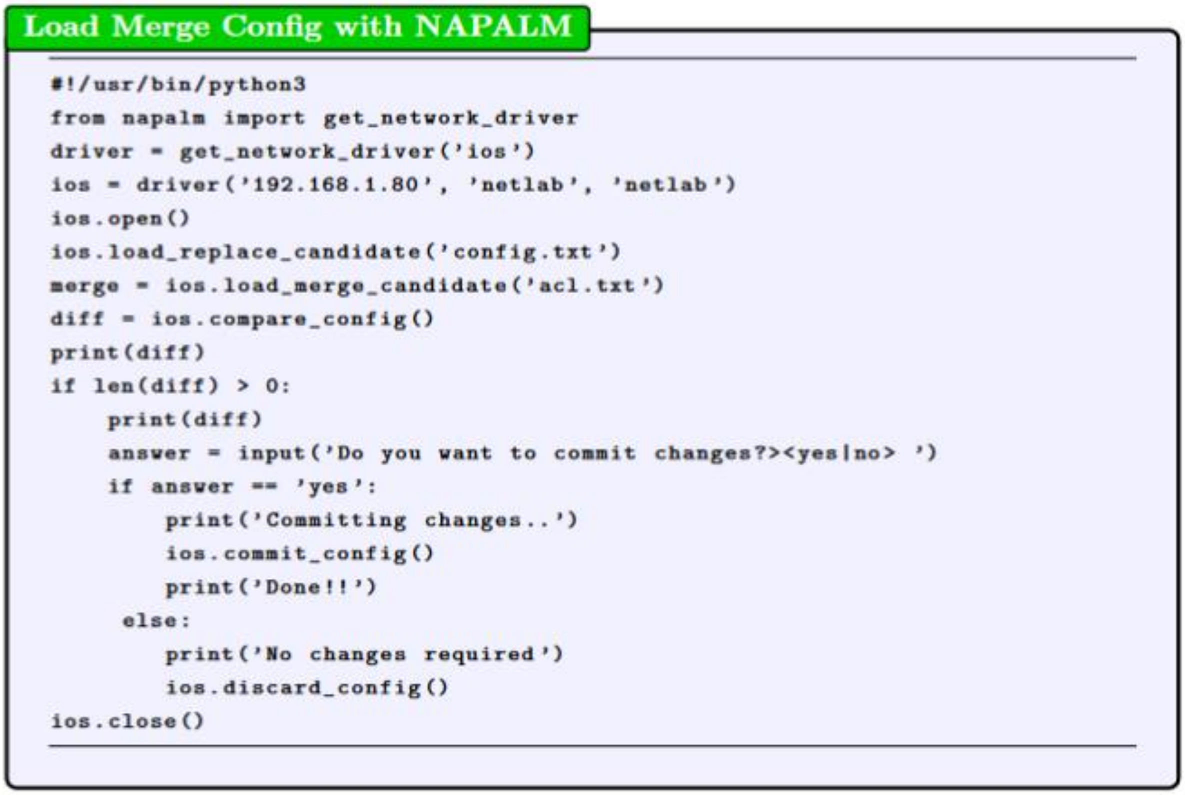
```
#!/usr/bin/python3
from napalm import get_network_driver
driver = get_network_driver(’ios’)
ios = driver(’192.168.1.80’, ’netlab’, ’netlab’)
ios.open()
ios.load_replace_candidate(’myconfig.txt’)
diff = ios.compare_config()
if len(diff):
    print(diff)
    print("Applying Config...")
    ios.commit_config()
else:
    print("No Changes to the config..")
    ios.discard_config()
ios.close()
```

Figure 10 Load Replace Configuration with NAPALM

If you are not happy with the configuration you can discard the configuration changes.

Merge Configuration

Merging the configuration is very similar process to the replace configuration, you must load the configuration with merge method. As shown In figure 11, the load_replace_candidate('config.txt') is used before load_merge_candidate('acl.txt').



Load Merge Config with NAPALM

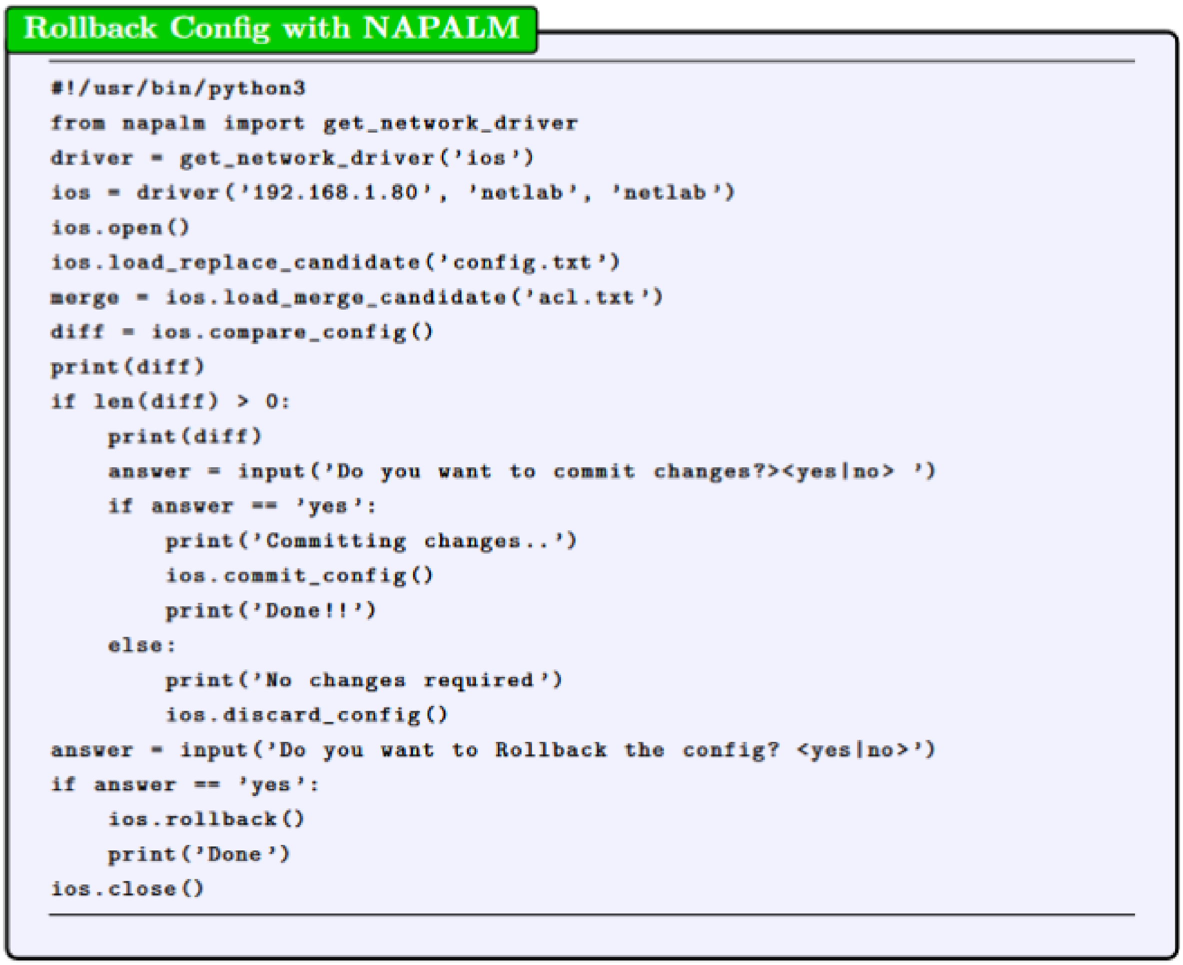
```
#!/usr/bin/python3
from napalm import get_network_driver
driver = get_network_driver('ios')
ios = driver('192.168.1.80', 'netlab', 'netlab')
ios.open()
ios.load_replace_candidate('config.txt')
merge = ios.load_merge_candidate('acl.txt')
diff = ios.compare_config()
print(diff)
if len(diff) > 0:
    print(diff)
    answer = input('Do you want to commit changes?><yes|no> ')
    if answer == 'yes':
        print('Committing changes..')
        ios.commit_config()
        print('Done!!')
     else:
        print('No changes required')
        ios.discard_config()
ios.close()
```

Figure 11 Merge Configuration with NAPALM

Same as load replace configuration, if you are happy with compare configuration changes and you want to commit the config you can go ahead and commit the changes on to network device or you can discard the configuration changes.

Rollback Configuration

NAPALM offers great feature to rollback the configuration. In case of misconfiguration or unforeseen outage due to changes made to network devices, we can rollback the configuration to the original config(previous state). Figure 12 shows the best combination of merge configuration and rollback configuration in action.



Rollback Config with NAPALM

```
#!/usr/bin/python3
from napalm import get_network_driver
driver = get_network_driver(’ios’)
ios = driver(’192.168.1.80’, ’netlab’, ’netlab’)
ios.open()
ios.load_replace_candidate(’config.txt’)
merge = ios.load_merge_candidate(’acl.txt’)
diff = ios.compare_config()
print(diff)
if len(diff) > 0:
    print(diff)
    answer = input(’Do you want to commit changes?><yes|no> ’)
    if answer == ’yes’:
        print(’Committing changes..’)
        ios.commit_config()
        print(’Done!!’)
    else:
        print(’No changes required’)
        ios.discard_config()
answer = input(’Do you want to Rollback the config? <yes|no>’)
if answer == ’yes’:
    ios.rollback()
    print(’Done’)
ios.close()
```

Figure 12 Rollback Configuration with NAPALM

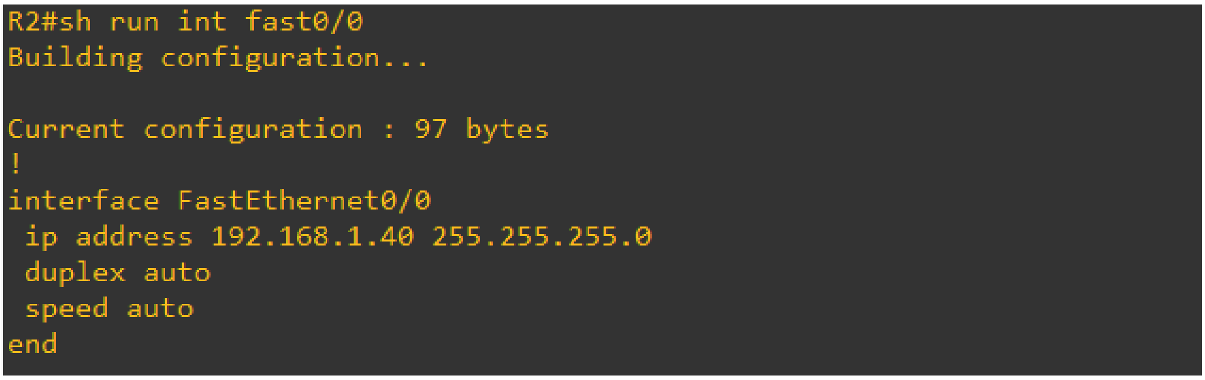
For some platforms, NAPALM provides commit config and set revert timer as well. If the user doesn’t have the confirm commit option set confirm_commit(revert_in=300) (300 is seconds), before the revert timer expires, then the configuration will be automatically rolled back to its previous state(and the candidate configuration will be discarded).

Merge/Rollback Configuration Scenario

In this scenario, we are going to configure some access rules from the acl.txt file. We are going to use merge configuration with existing configuration. We will compare the configuration with the current configuration and display the configuration differences. If we are happy we will commit the changes.

After committing the changes, we will prompt the user if they want to rollback the changes, we will go ahead and rollback the configuration, in case of both the prompts if answer is no then configuration will be discarded and changes will be taken place.

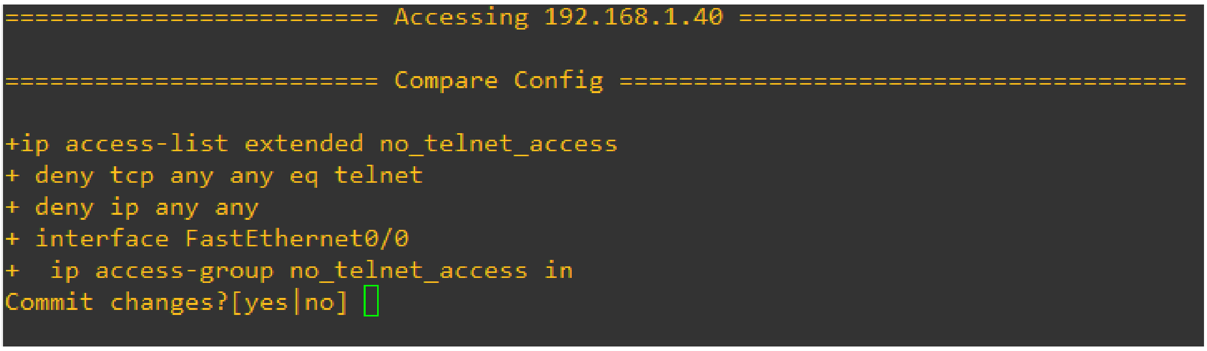
Figure 13 shows the Router interface before configuration, before executing the config_rollback.py.



```
R2#sh run int fast0/0
Building configuration...

Current configuration : 97 bytes
!
interface FastEthernet0/0
 ip address 192.168.1.40 255.255.255.0
 duplex auto
 speed auto
end
```

Figure 13 Router - Interface config

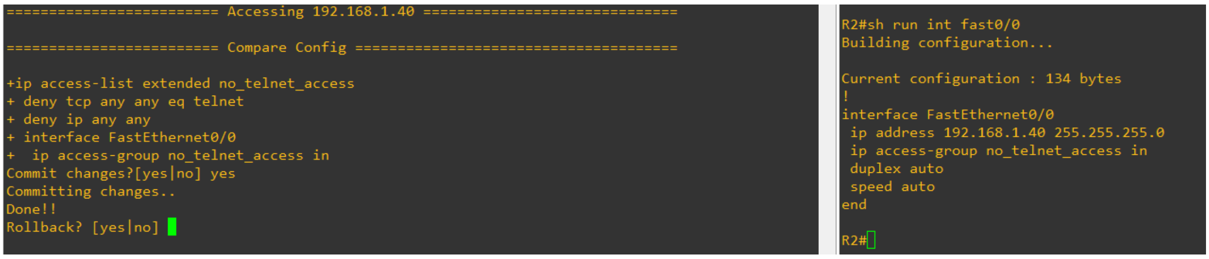
Launching the config_rollback.py, connected to the host, configuration change is shown in figure 14. Configuring Access List on interface fast0/0.



```
======================== Accessing 192.168.1.40 ==============================

======================== Compare Config ======================================

+ip access-list extended no_telnet_access
+ deny tcp any any eq telnet
+ deny ip any any
+ interface FastEthernet0/0
+  ip access-group no_telnet_access in
Commit changes?[yes|no]
```

Figure 14 - Commit ACL changes

Comparing config before merging the configuration. As the figure 14 shows, if a user enters yes, commit will be saved shown in figure 15, sorry for the tiny font output.



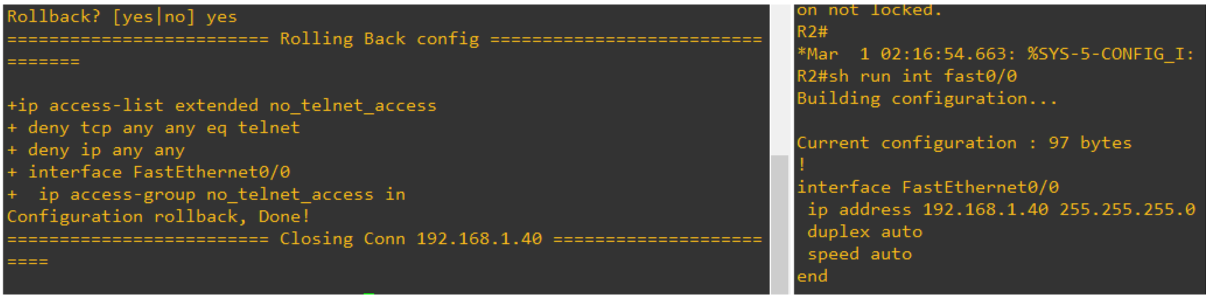
```
======================== Accessing 192.168.1.40 ==============================

======================== Compare Config ======================================

+ip access-list extended no_telnet_access
+ deny tcp any any eq telnet
+ deny ip any any
+ interface FastEthernet0/0
+  ip access-group no_telnet_access in
Commit changes?[yes|no] yes
Committing changes..
Done!!
Rollback? [yes|no]
```

```
R2#sh run int fast0/0
Building configuration...

Current configuration : 134 bytes
!
interface FastEthernet0/0
 ip address 192.168.1.40 255.255.255.0
 ip access-group no_telnet_access in
 duplex auto
 speed auto
end

R2#
```

Figure 15 - Committed ACL on the Router

We can set the confirm_commit() timer too but in this scenario we are not.

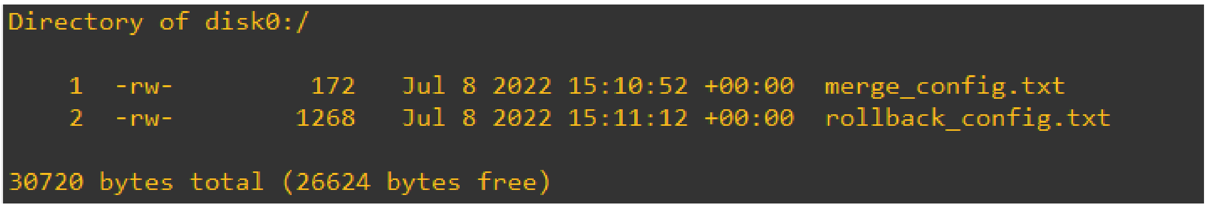
If we click yes for the rollback the commit will be rollback to the previous configuration state as shown in Figure 16, the ip access list on interface has been removed.



```
Rollback? [yes|no] yes
======================== Rolling Back config =========================
=======

+ip access-list extended no_telnet_access
+ deny tcp any any eq telnet
+ deny ip any any
+ interface FastEthernet0/0
+  ip access-group no_telnet_access in
Configuration rollback, Done!
======================== Closing Conn 192.168.1.40 ===================
====
```

```
on not locked.
R2#
*Mar  1 02:16:54.663: %SYS-5-CONFIG_I:
R2#sh run int fast0/0
Building configuration...

Current configuration : 97 bytes
!
interface FastEthernet0/0
 ip address 192.168.1.40 255.255.255.0
 duplex auto
 speed auto
end
```

Figure 16 - Rollback conf



```
Directory of disk0:/

    1  -rw-         172   Jul 8 2022 15:10:52 +00:00  merge_config.txt
    2  -rw-        1268   Jul 8 2022 15:11:12 +00:00  rollback_config.txt

30720 bytes total (26624 bytes free)
```

Figure 127 - Config Files on the disk0

Figure 17 shows the both the merge_config and rollback config files.
You can view the content of the file using more command on cisco device

```
R2#more disk0:merge_config.txt
ip access-list extended no_telnet_access
 deny tcp any any eq telnet
 permit ip any any
 deny ip any any

 interface FastEthernet0/0
  ip access-group no_telnet_access in
```

Task completed successfully!!

Note: Merge configuration on the virtual QEMU can be challenging and subject to many errors.

References

1. "Memory replacement instructions for the network processing engine or network services engine and input/output controller - flash memory card information (CISCO 7200 series routers)," *Cisco*, 12-Oct-2013. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/routers/7200/install_and_upgrade/npe-nse_memory_install/memory/8358ov1.html. [Accessed: 21-Apr-2022].

2. "Changing the configuration," Changing the Configuration - NAPALM 3 documentation. [Online]. Available: https://napalm.readthedocs.io/en/develop/tutorials/changing_the_config.html. [Accessed: 21-Apr-2022].