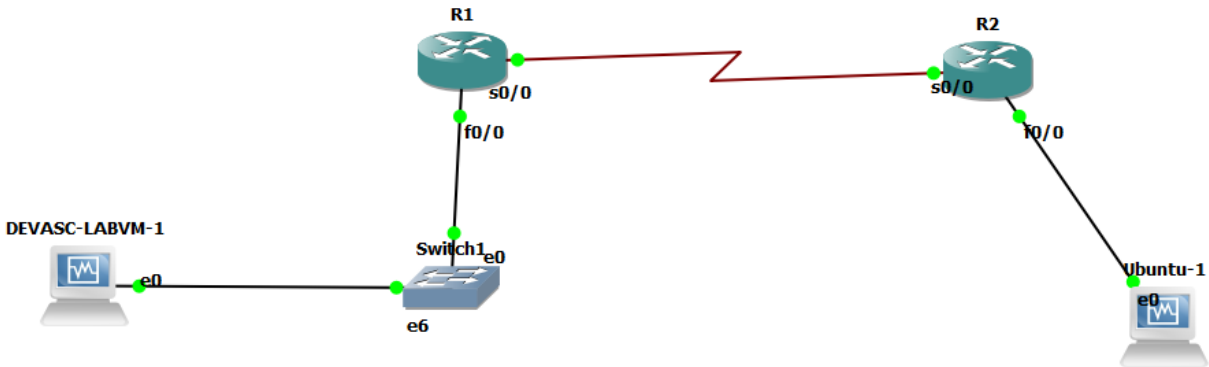


Topology



Addressing Table

Device	Interface	Address	Subnet Mask
R1	S0/0	10.0.0.1	255.255.255.252
	F0/0	192.168.10.254	255.255.255.0
R2	S0/0	10.0.0.2	255.255.255.252
	F0/0	192.168.20.254	255.255.255.0
SW1	Ethernet 0	--	--
	Ethernet 6	--	--
	Ethernet 7	--	--
Ubuntu-1	--	192.168.20.2	255.255.255.0
DEVASC-LABVM-1	--	192.168.10.2	255.255.255.0

Objectives

- Part 1: Launch the DEVASC VM
- Part 2: Launch GNS3
- Part 3: Configure Basic Device Setting
- Part 4: Creating a YAML File
- Part 5: Use the pyATS Testing Library
- Part 6: Use Genie to Learn

Background / Scenario

Required Resources

- 1 PC with operating system of your choice
- Virtual Box or VMWare
- DEVASC Virtual Machine
- GNS3

Instructions

Part 1: Launch the DEVASC VM

If you have not already completed the **Lab - Install the Virtual Machine Lab Environment**, do so now. If you have already completed that lab, launch the DEVASC VM now

Part 2: Launch GNS3

If you have not download GNS3, go to this website: <https://gns3.com/software/download>

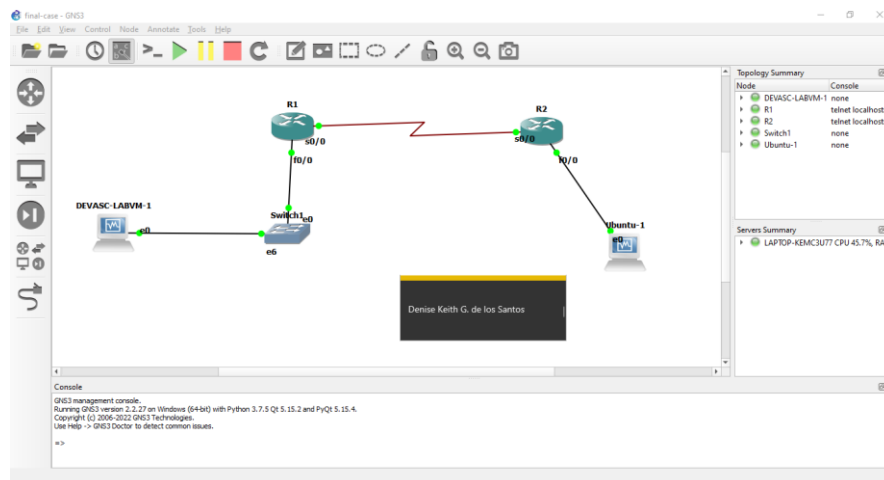
Step 1: Create new Project.

- Click File that can be seen on the upper screen.
- Look and click New blank project.
- Then click OK

Part 3: Configure Basic Device Setting

Step 1: Cable the network as shown in the topology

Attach the devices as shown in the topology diagram, and cable as necessary.



Step 2: Configure basic configuration

- Configure the username and password.

```
Username cisco password cisco123
```

- Set the domain name of the router

```
Ip domain name www.abc.com
```

- Generate a set of crypto keys with a 1024-bit modulus.

```
Crypto key gen rsa
```

```
Ip ssh ver 2
```

- Configure the enable password and secret to "cisco"

```
Enable secret cisco123
```

- Assign cisco as the vty password, configure the vty lines to accept SSH connections only and enable login using the local database.

```
Line vty 0 15
```

```
Login local
```

```
Transport input ssh
```

- f. Configure all interfaces on the routers with IPv4 addressing information from the addressing table above.

For R1:

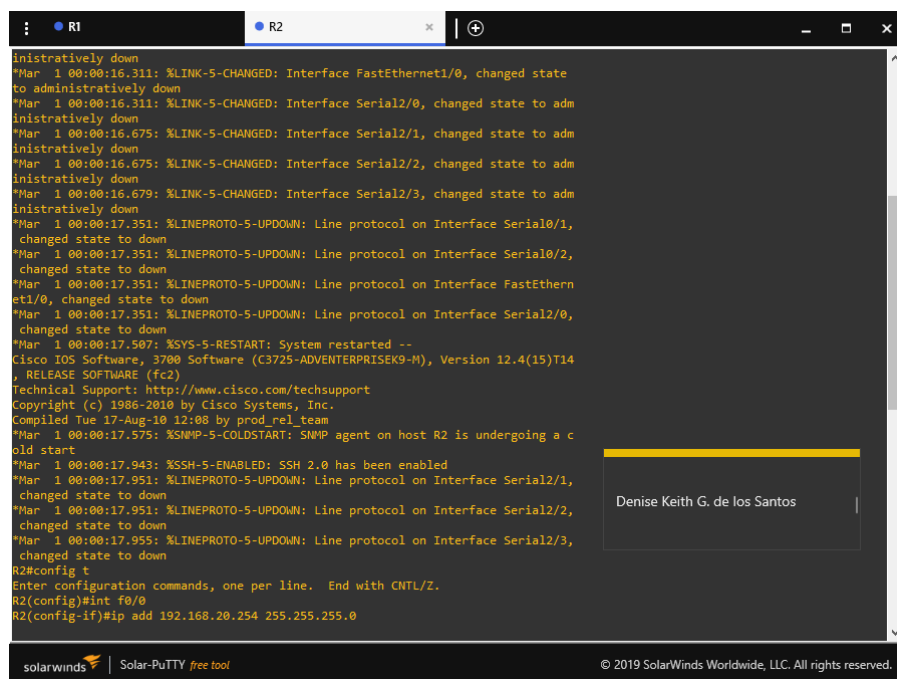
```
Int S0/0
Ip address 10.0.0.1 255.255.255.252
No shut
Int F0/0
Ip address 192.168.10.254 255.255.255.0
No shut
```

For R2:

```
Int S0/0
Ip address 10.0.0.2 255.255.255.252
No shut
Int F0/0
Ip address 192.168.20.254 255.255.255.0
No shut
```

- g. Save the running configuration to the startup configuration file.

```
Do copy run start
```



```
iniistratively down
*Mar 1 00:00:16.311: %LINK-5-CHANGED: Interface FastEthernet1/0, changed state
to administratively down
*Mar 1 00:00:16.311: %LINK-5-CHANGED: Interface Serial2/0, changed state to adm
iniistratively down
*Mar 1 00:00:16.675: %LINK-5-CHANGED: Interface Serial2/1, changed state to adm
iniistratively down
*Mar 1 00:00:16.675: %LINK-5-CHANGED: Interface Serial2/2, changed state to adm
iniistratively down
*Mar 1 00:00:16.679: %LINK-5-CHANGED: Interface Serial2/3, changed state to adm
iniistratively down
*Mar 1 00:00:17.351: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1,
changed state to down
*Mar 1 00:00:17.351: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/2,
changed state to down
*Mar 1 00:00:17.351: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthern
et1/0, changed state to down
*Mar 1 00:00:17.351: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0,
changed state to down
*Mar 1 00:00:17.507: %SYS-5-RESTART: System restarted --
Cisco IOS Software, 3700 Software (C3725-ADVENTERPRISEK9-M), Version 12.4(15)T14
, RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2010 by Cisco Systems, Inc.
Compiled Tue 17-Aug-10 12:08 by prod_rel_team
*Mar 1 00:00:17.575: %SNMP-5-COLDSTART: SNMP agent on host R2 is undergoing a c
old start
*Mar 1 00:00:17.943: %SSH-5-ENABLED: SSH 2.0 has been enabled
*Mar 1 00:00:17.951: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/1,
changed state to down
*Mar 1 00:00:17.951: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/2,
changed state to down
*Mar 1 00:00:17.955: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/3,
changed state to down
R2#config t
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#int f0/0
R2(config-if)#ip add 192.168.20.254 255.255.255.0
```

Step 3: Verify network connectivity.

Ping in the terminal in each device the neighbor ip address.

The screenshot shows a SolarWinds Solar-PuTTY terminal window with two tabs, R1 and R2. The R1 tab is active, displaying the following text:

```
Cisco IOS Software, 3700 Software (C3725-ADVENTERPRISEK9-M), Version 12.4(15)T14
, RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2010 by Cisco Systems, Inc.
Compiled Tue 17-Aug-10 12:08 by prod_rel_team
*Mar 1 00:00:24.075: %SNMP-5-COLDSTART: SNMP agent on host R1 is undergoing a c
old start
*Mar 1 00:00:24.687: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1,
changed state to down
*Mar 1 00:00:24.971: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/2,
changed state to down
*Mar 1 00:00:24.971: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet
et1/0, changed state to down
*Mar 1 00:00:24.971: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0,
changed state to down
*Mar 1 00:00:24.971: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/1,
changed state to down
*Mar 1 00:00:24.971: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/2,
changed state to down
*Mar 1 00:00:24.971: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/3,
changed state to down
*Mar 1 00:44:51.243: %SYS-5-CONFIG_I: Configured from console by cisco on vty0
(192.168.10.2)
*Mar 1 00:45:04.835: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.20.254 on Serial0/0
from LOADING to FULL, Loading Done
R1#ping 10.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/5/12 ms
R1#ping 192.168.10.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.10.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/12/16 ms
R1#
```

A small text box on the right side of the terminal window contains the name "Denise Keith G. de los Santos". The SolarWinds logo and "Solar-PuTTY free tool" are visible in the bottom left, and "© 2019 SolarWinds Worldwide, LLC. All rights reserved." is in the bottom right.

Part 4: Creating a YAML File

- In the PC, create a new folder named FinalCase
- Create file for hosts and ansible.

For ansible:

```
[defaults]

inventory = ./hosts

host_key_checking = false

timeout = 15

deprecation_warnings=False

remote_user = Den
```

For hosts:

```
[routers]

R1 ansible_host=10.0.0.1

R2 ansible_host=10.0.0.2


[routers:vars]

ansible_user=cisco

ansible_password=cisco123

ansible_connection=network_cli

ansible_network_os=ios
```

```

ansible_port=22

ansible_become=yes

ansible_become_method=enable

ansible_become_pass=cisco123


[ubuntu]

192.168.20.2 ansible_ssh_pass=qwerty ansible_ssh_user=den
ansible_password=qwerty ansible_port=22 ansible_become=yes
ansible_sudo_pass=qwerty

```

c. Create file YAML for OSPF and paste the given code.

```

---
- name: Configure OSPF on R1
  hosts: R1
  gather_facts: false
  connection: local

  vars:
    cli:
      username: cisco
      password: cisco123

  tasks:
    - name: Enable OSPF on R1
      ios_config:
        provider: "{{ cli }}"
        authorize: yes
        parents: router ospf 1
        lines:
          - network 0.0.0.0 255.255.255.255 area 0

      register: print_output
    - debug: var=print_output

```

```

- name: Configure OSPF on R2
  hosts: R2
  gather_facts: false
  connection: local

vars:
  cli:
    username: cisco
    password: cisco123

tasks:
  - name: Enable OSPF on R2
    ios_config:
      provider: "{{ cli }}"
      authorize: yes
      parents: router ospf 1
      lines:
        - network 0.0.0.0 255.255.255.255 area 0

    register: print_output
  - debug: var=print_output

```

d. Create file for YAML for ACL.

```

---

- name: Configure ACL on R2
  hosts: R2
  gather_facts: false
  connection: local

tasks:
  - name: Configure ACL inbound
    ios_config:
      lines:

```

```

    - ip access-group 110 in
    parents: interface FastEthernet0/0

- name: Create inbound ACL rules for R2
  ios_config:
    lines:
      - access-list 110 deny icmp any any echo-reply
      - access-list 110 permit ip any any
    before: no access-list 110
    match: exact

```

e. Create file for Backup

```

---
- name: Backup for R1
  hosts: R1
  gather_facts: false
  connection: local

  tasks:
    - name: Show the Startup Configuration
      ios_command:
        commands:
          - show startup-config
      register: startupconfig

    - name: SAVE OUTPUT TO ./backups/
      copy:
        content: "{{ startupconfig.stdout[0] }}"
        dest: "backups/R1_startup.txt"

    - name: Show the Running Configuration
      ios_command:
        commands:

```

```

- show running-config

register: config

- name: SAVE OUTPUT TO ./backups/

copy:

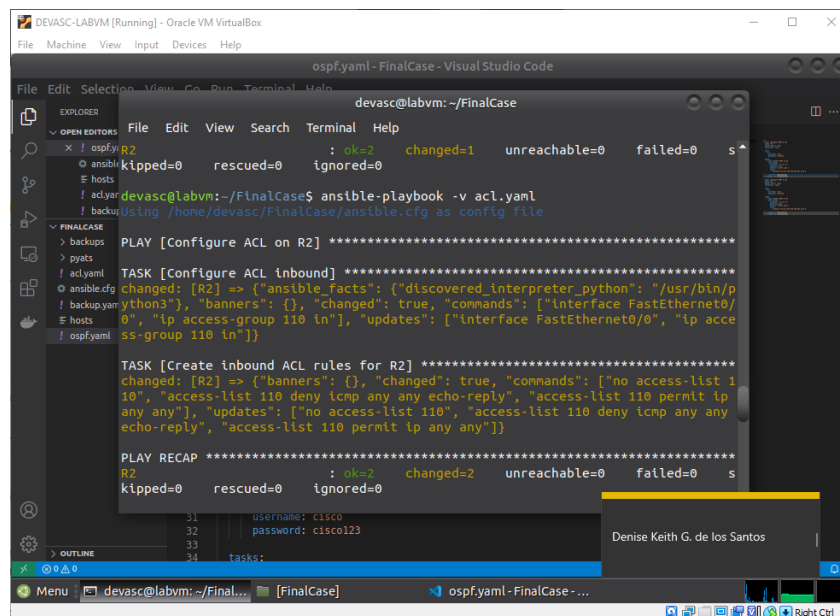
  content: "{{ config.stdout[0] }}"

  dest: "backups/R1_run.txt"

```

- f. Run the YAML file. Change the highlighted text to the name of the file.

```
ansible-playbook -v filename.yaml
```



Part 5: Use the pyATS Testing Library

In this part, you will use pyATS, a python testing library and genie.

Step 1: Create python file for script and job

- a. Python file for script

```

# To get a logger for the script

import logging

# Needed for aetest script

from pyats import aetest

# Get your logger for your script

log = logging.getLogger(__name__)

```



```

#####
##

###          COMMON SETUP SECTION
###

#####
##

# This is how to create a CommonSetup
# You can have one of no CommonSetup
# CommonSetup can be named whatever you want
class common_setup(aetest.CommonSetup):
    """ Common Setup section """
    # CommonSetup have subsection.
    # You can have 1 to as many subsection as wanted
    # here is an example of 2 subsections
    # First subsection
    @aetest.subsection
    def sample_subsection_1(self):
        """ Common Setup subsection """
        log.info("Aetest Common Setup ")
        # If you want to get the name of current section,
        # add section to the argument of the function.
        # Second subsection
        @aetest.subsection
        def sample_subsection_2(self, section):
            """ Common Setup subsection """
            log.info("Inside %s" % (section))
            # And how to access the class itself ?
            # self refers to the instance of that class, and remains
consistent
            # throughout the execution of that container.
            log.info("Inside class %s" % (self.uid))
#####
##

```

```

###                                TESTCASES SECTION
###

#####
##

# This is how to create a testcase
# You can have 0 to as many testcase as wanted
# Testcase name : tc_one
class tc_one(aetest.Testcase):
    """ This is user Testcases section """
    # Testcases are divided into 3 sections
    # Setup, Test and Cleanup.
    # This is how to create a setup section
    @aetest.setup
    def prepare_testcase(self, section):
        """ Testcase Setup section """
        log.info("Preparing the test")
        log.info(section)
    # This is how to create a test section
    # You can have 0 to as many test section as wanted

    # First test section
    @ aetest.test
    def simple_test_1(self):
        """ Sample test section. Only print """
        log.info("First test section ")
    # Second test section
    @ aetest.test
    def simple_test_2(self):
        """ Sample test section. Only print """
        log.info("Second test section ")
    # This is how to create a cleanup section
    @aetest.cleanup
    def clean_testcase(self):

```

```

        """ Testcase cleanup section """

        log.info("Pass testcase cleanup")
# Testcase name : tc_two
class tc_two(aetest.Testcase):
    """ This is user Testcases section """

    @ aestest.test
    def simple_test_1(self):
        """ Sample test section. Only print """

        log.info("First test section ")

        self.failed('This is an intentional failure')

# Second test section
    @ aestest.test
    def simple_test_2(self):
        """ Sample test section. Only print """

        log.info("Second test section ")

# This is how to create a cleanup section
    @aetest.cleanup
    def clean_testcase(self):
        """ Testcase cleanup section """

        log.info("Pass testcase cleanup")

#####
####

####                                COMMON CLEANUP SECTION
###

#####

####

# This is how to create a CommonCleanup
# You can have 0 , or 1 CommonCleanup.
# CommonCleanup can be named whatever you want :)
class common_cleanup(aetest.CommonCleanup):
    """ Common Cleanup for Sample Test """

    # CommonCleanup follow exactly the same rule as CommonSetup
    regarding

    # subsection

```

```

# You can have 1 to as many subsection as wanted

# here is an example of 1 subsections

@aetest.subsection

def clean_everything(self):

    """ Common Cleanup Subsection """

    log.info("Aetest Common Cleanup ")

if __name__ == '__main__': # pragma: no cover
    aetest.main()

```

b. Python file for jobs

```

import os

from pyats.easypy import run

# All run() must be inside a main function

def main():

    # Find the location of the script in relation to the job file
    test_path = os.path.dirname(os.path.abspath(__file__))

    testscript = os.path.join(test_path, 'script.py')

    # Execute the testscript

    run(testscript=testscript)

```

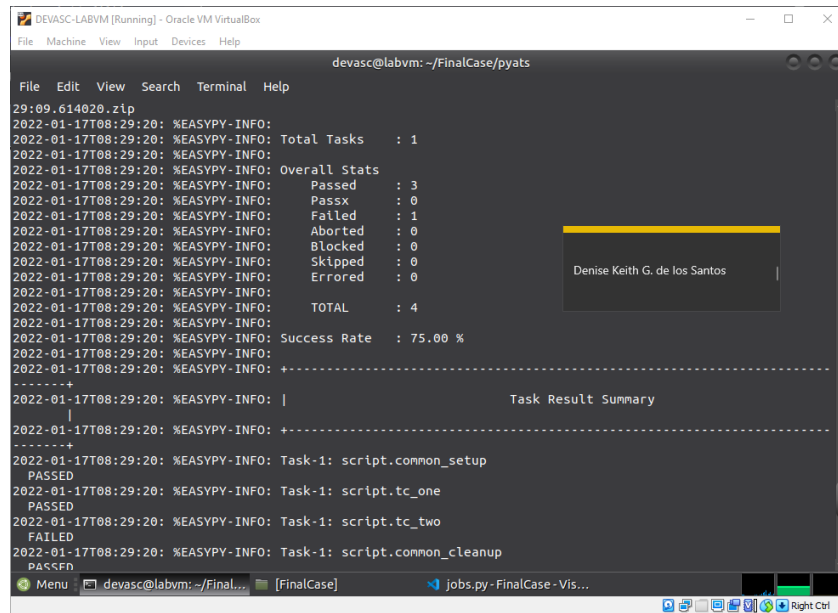
Step 2: Run pyATS manually

- a. Using the pyATS job and script files, run pyATS manually to invoke the basic test case. This will verify the pyATS job and script files work properly.

```

pyats run job jobs.py

```



```
devasc@labvm: ~/FinalCase/pyats
29:09.614020.zlp
2022-01-17T08:29:20: %EASYPY-INFO:
2022-01-17T08:29:20: %EASYPY-INFO: Total Tasks : 1
2022-01-17T08:29:20: %EASYPY-INFO:
2022-01-17T08:29:20: %EASYPY-INFO: Overall Stats
2022-01-17T08:29:20: %EASYPY-INFO: Passed : 3
2022-01-17T08:29:20: %EASYPY-INFO: Passx : 0
2022-01-17T08:29:20: %EASYPY-INFO: Failed : 1
2022-01-17T08:29:20: %EASYPY-INFO: Aborted : 0
2022-01-17T08:29:20: %EASYPY-INFO: Blocked : 0
2022-01-17T08:29:20: %EASYPY-INFO: Skipped : 0
2022-01-17T08:29:20: %EASYPY-INFO: Errored : 0
2022-01-17T08:29:20: %EASYPY-INFO:
2022-01-17T08:29:20: %EASYPY-INFO: TOTAL : 4
2022-01-17T08:29:20: %EASYPY-INFO: Success Rate : 75.00 %
2022-01-17T08:29:20: %EASYPY-INFO:
-----+
2022-01-17T08:29:20: %EASYPY-INFO: | Task Result Summary
2022-01-17T08:29:20: %EASYPY-INFO: +-----+
-----+
2022-01-17T08:29:20: %EASYPY-INFO: Task-1: script.common_setup
PASSED
2022-01-17T08:29:20: %EASYPY-INFO: Task-1: script.tc_one
PASSED
2022-01-17T08:29:20: %EASYPY-INFO: Task-1: script.tc_two
FAILED
2022-01-17T08:29:20: %EASYPY-INFO: Task-1: script.common_cleanup
PASSED
```

Part 6: Use Genie to Learn

Step 1: Create Virtual Environment

- Enter the Virtual Environment

```
python3 -m venv Name of the Environment
```

- Activate the Virtual Environment

```
source bin/activate
```

Step 2: Create a testbed YAML file

- To create your testbed YAML file, enter the command below.

```
genie create testbed interactive --output filename.yaml
```

- Using your testbed YAML file, invoke Genie to learn

```
genie learn acl --testbed-file filename.yaml --output Name of the folder
```

```
devasc@labvm: ~/FinalCase/Case
File Edit View Search Terminal Help

100% | 1/1 [01:21<00:00, 81.26s/it]
=====
Genie Learn Summary for device R1
=====
Connected to R1
- Log: ospf_configure/connection_R1.txt

Learnt feature 'ospf'
- Ops structure: ospf_configure/ospf_ios_R1_ops.txt
- Device Console: ospf_configure/ospf_ios_R1_console.txt
=====

Genie Learn Summary for device R2
=====
Connected to R2
- Log: ospf_configure/connection_R2.txt

Learnt feature 'ospf'
- Ops structure: ospf_configure/ospf_ios_R2_ops.txt
- Device Console: ospf_configure/ospf_ios_R2_console.txt
=====

Network:
Receiving 0 bytes/s
Sending 0 bytes/s

Python 3.8.10 64-bit Ln 23, Col 24 Spaces: 2
```

```
devasc@labvm: ~/FinalCase/Case
File Edit View Search Terminal Help

100% | 1/1 [00:04<00:00, 4.85s/it]
=====
Genie Learn Summary for device R1
=====
Connected to R1
- Log: acl_configure/connection_R1.txt

Learnt feature 'acl'
- Ops structure: acl_configure/acl_ios_R1_ops.txt
- Device Console: acl_configure/acl_ios_R1_console.txt
=====

Genie Learn Summary for device R2
=====
Connected to R2
- Log: acl_configure/connection_R2.txt

Learnt feature 'acl'
- Ops structure: acl_configure/acl_ios_R2_ops.txt
- Device Console: acl_configure/acl_ios_R2_console.txt
=====

Python 3.8.10 64-bit Ln 23, Col 24 Spaces: 2 UTF-8 LF YAML
```

"I affirm that I have not given or received any unauthorized help on this assignment, and that this work is my own."