Reinhart Wilson L. Regulacion	**01/17/2022**
CPE41S1	**Engr. Alonica Villanueva**

Final Case Study – Network Automation and Programmability

Objectives

The objective of this activity is to design a laboratory activity that discusses the three network topics excluding basic configuration, IP address, and show commands regarding network automation or network programmability. Utilizing of Ansible for implementation of OSPF and ACL configuration, and backup are selected for the topic. Additionally, the case study follows PyATS to test the network through automation.

Required Resources

- 1 PC with any operating system
- Virtual Box, VMWare or any preferred software for virtualization and emulation of virtual machines
- DEVASC Linux Virtual Machine
- GNS3 software
- Visual Studio Code or any preferred Code Editor.
- PyATS
- Cisco file images (C3725, C2960, etc.)

Topology

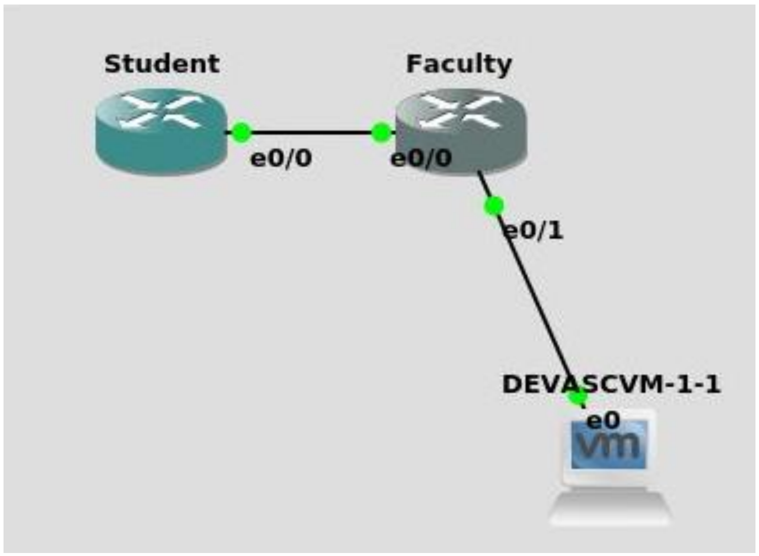


Figure 1. Network Topology

Addressing Table

Device	Interface	IP Address	Subnet Masks	Default Gateway
R1	Ethernet0/1	192.168.10.14	255.255.255.0	N/A
	Ethernet0/0	10.10.10.1	255.255.255.252	
R2	Ethernet0/0	10.10.10.2	255.255.255.252	
DEVASCVM-1-1	NIC	192.168.10.15	255.255.255.0	192.168.10.14

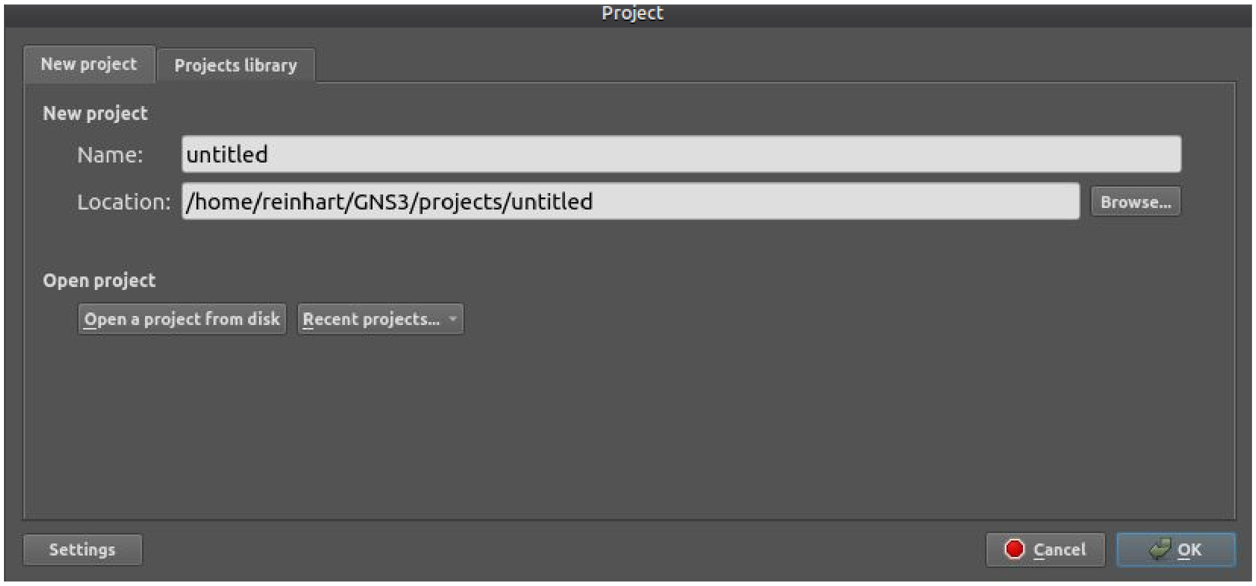
Procedure

Part 1. Launch the DEVASC VM

Note: 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. Open GNS3 and Create the Network

Step 1: Create a new Project



New project

Name: untitled

Location: /home/reinhart/GNS3/projects/untitled

Figure 2. Open a new project

Step 2: Install the CISCO router and switch images necessary for network simulation.

Step 3: Create the topology

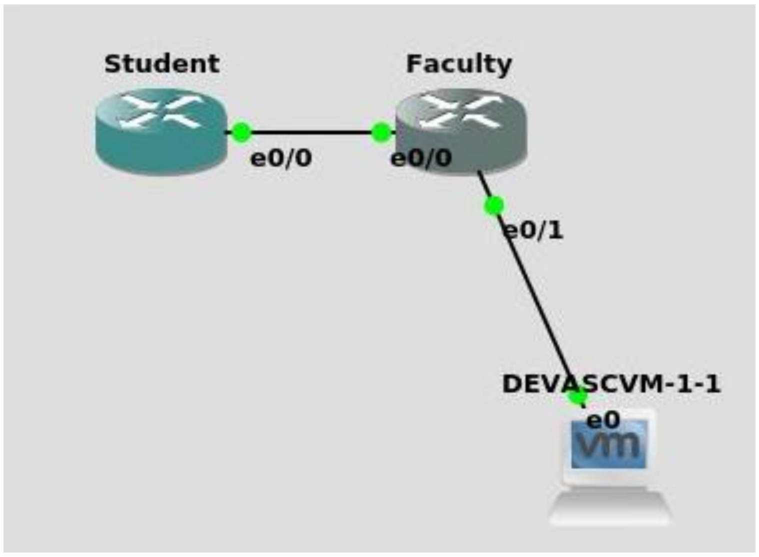


Figure 3. Network Topology

Step 4: Implement basic, IP routing, and SSH configurations for routers and switches.

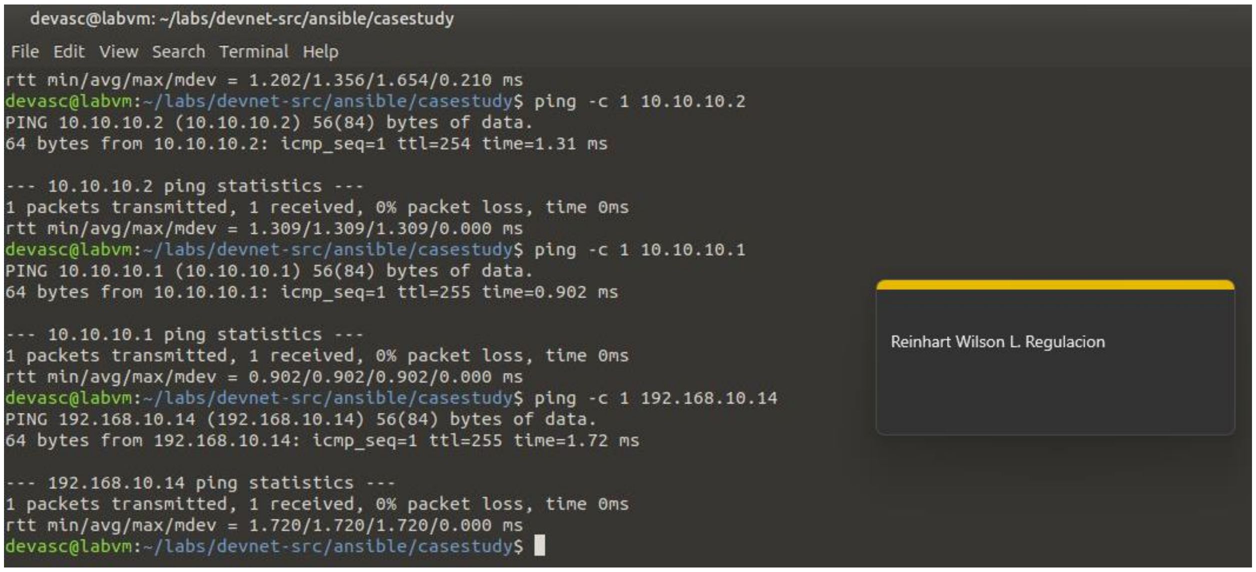
FOR R1 router:

```
enable
configure terminal
hostname Faculty
ip domain-name netacad.com
crypto key generate rsa
2048
ip ssh version 2
line vty 0 4
transport input ssh
login local
exit
service password-encryption
banner motd "Unauthorized access is prohibited!"
line console 0
logging synchronous
login local
username cisco privilege 15 password cisco123

interface e0/1
ip address 192.168.10.14 255.255.255.0
no shutdown
interface e0/0
ip address 10.10.10.1 255.255.255.252
no shutdown
```

FOR R2 router

```
enable
configure terminal
hostname Student
ip domain-name netacad.com
crypto key generate rsa
2048
ip ssh version 2
line vty 0 4
transport input ssh
login local
exit
service password-encryption
banner motd "Unauthorized access is prohibited!"
line console 0
logging synchronous
login local
username cisco privilege 15 password cisco123

interface e0/0
ip address 10.10.10.2 255.255.255.252
no shutdown
```

Step 5: Check the connections of all the network and end devices through ping

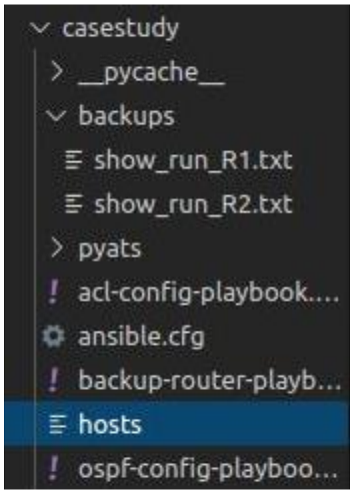

devasc@labvm: ~/labs/devnet-src/ansible/casestudy
File Edit View Search Terminal Help
Reinhart Wilson L. Regulacion

```
rtt min/avg/max/mdev = 1.202/1.356/1.654/0.210 ms
devasc@labvm:~/labs/devnet-src/ansible/casestudy$ ping -c 1 10.10.10.2
PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data.
64 bytes from 10.10.10.2: icmp_seq=1 ttl=254 time=1.31 ms

--- 10.10.10.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.309/1.309/1.309/0.000 ms
devasc@labvm:~/labs/devnet-src/ansible/casestudy$ ping -c 1 10.10.10.1
PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
64 bytes from 10.10.10.1: icmp_seq=1 ttl=255 time=0.902 ms

--- 10.10.10.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.902/0.902/0.902/0.000 ms
devasc@labvm:~/labs/devnet-src/ansible/casestudy$ ping -c 1 192.168.10.14
PING 192.168.10.14 (192.168.10.14) 56(84) bytes of data.
64 bytes from 192.168.10.14: icmp_seq=1 ttl=255 time=1.72 ms

--- 192.168.10.14 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.720/1.720/1.720/0.000 ms
devasc@labvm:~/labs/devnet-src/ansible/casestudy$
```

Figure 4. Establishing Ping commands

Part 3. Using Ansible to configure the major components of Network and implement backup configuration files.

Step 1: Create a File Directory named “casestudy”.



```
casestudy
  __pycache__
  backups
    show_run_R1.txt
    show_run_R2.txt
  pyats
  acl-config-playbook....
  ansible.cfg
  backup-router-playb...
  hosts
  ospf-config-playboo...
```

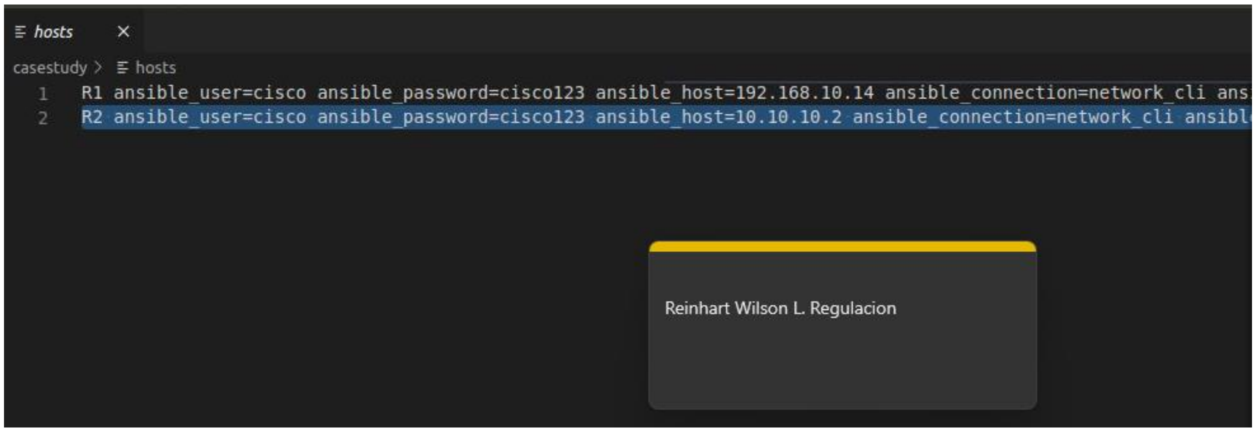
Figure 5. File directory

Step 2: Create a file named hosts.

Enter the following lines:

R1 ansible_user=cisco ansible_password=cisco123 ansible_host=192.168.10.14 ansible_connection=network_cli ansible_network_os=ios ansible_become=yes ansible_become_method=enable ansible_become_pass=cisco

R2 ansible_user=cisco ansible_password=cisco123 ansible_host=10.10.10.2 ansible_connection=network_cli ansible_network_os=ios ansible_become=yes ansible_become_method=enable ansible_become_pass=cisco



```
R1 ansible_user=cisco ansible_password=cisco123 ansible_host=192.168.10.14 ansible_connection=network_cli ans
R2 ansible_user=cisco ansible_password=cisco123 ansible_host=10.10.10.2 ansible_connection=network_cli ansibl
```

Reinhart Wilson L. Regulacion

Figure 6. Contents of the “hosts” file

Step 3: Create an Ansible configuration file

Enter the following lines:

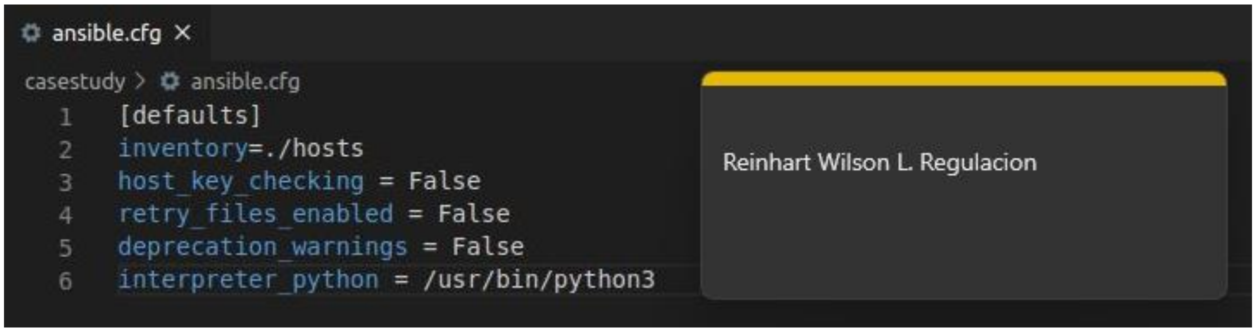
[defaults]

inventory=./hosts

host_key_checking = False # Don't worry about RSA Fingerprints

retry_files_enabled = False # Do not create them

deprecation_warnings = False # Do not show warnings



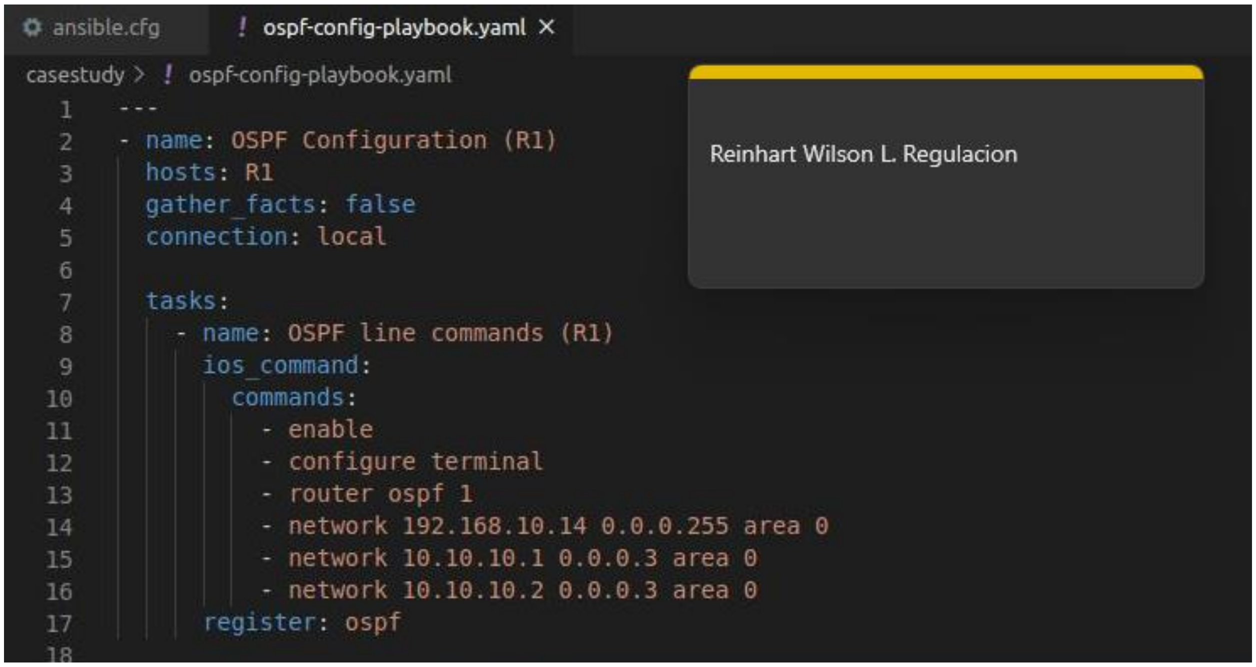
```
[defaults]
inventory=./hosts
host_key_checking = False
retry_files_enabled = False
deprecation_warnings = False
interpreter_python = /usr/bin/python3
```

Figure 7. Contents of “ansible cfg”

Step 4: Create a yaml file named “ospf_config_playbook” to configure the OSPF in a single area

Code lines:

```
---
- name: OSPF Configuration (R1)
  hosts: R1
  gather_facts: false
  connection: local

  tasks:
    - name: OSPF line commands (R1)
      ios_command:
        commands:
          - enable
          - configure terminal
          - router ospf 1
          - network 192.168.10.14 0.0.0.255 area 0
          - network 10.10.10.1 0.0.0.3 area 0
          - network 10.10.10.2 0.0.0.3 area 0
      register: ospf

- name: OSPF Configuration (R2)
  hosts: R2
  gather_facts: false
  connection: local

  tasks:
    - name: OSPF line commands (R2)
      ios_command:
        commands:
          - enable
          - configure terminal
          - router ospf 1
          - network 192.168.10.14 0.0.0.255 area 0
          - network 10.10.10.1 0.0.0.3 area 0
          - network 10.10.10.2 0.0.0.3 area 0
      register: ospf
```


ansible.cfg
ospf-config-playbook.yaml
casestudy > ospf-config-playbook.yaml
Reinhart Wilson L. Regulacion

```
---
- name: OSPF Configuration (R1)
  hosts: R1
  gather_facts: false
  connection: local

  tasks:
    - name: OSPF line commands (R1)
      ios_command:
        commands:
          - enable
          - configure terminal
          - router ospf 1
          - network 192.168.10.14 0.0.0.255 area 0
          - network 10.10.10.1 0.0.0.3 area 0
          - network 10.10.10.2 0.0.0.3 area 0
      register: ospf
```

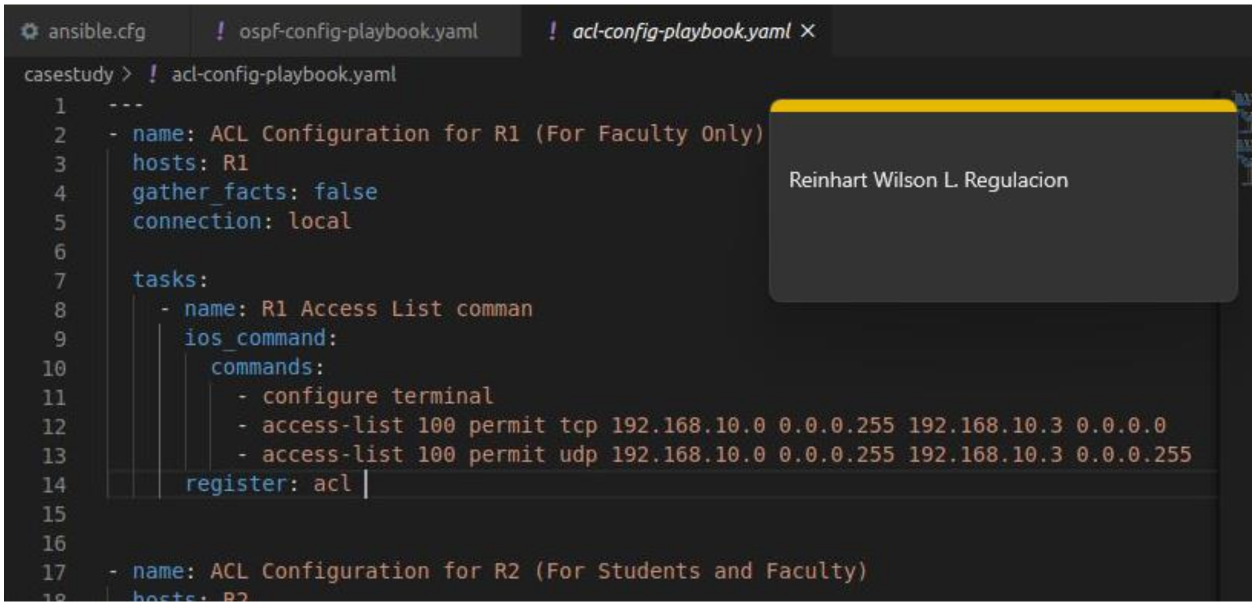
Figure 8. OSPF config ansible file

Step 5: Create a yaml file named “acl_config_playbook” to implement extended ACLs in the network

Code lines:

```
---
- name: ACL Configuration for R1 (For Faculty Only)
  hosts: R1
  gather_facts: false
  connection: local

  tasks:
```

```
  - name: R1 Access List comman
    ios_command:
      commands:
        - configure terminal
        - access-list 100 permit tcp 192.168.10.0 0.0.0.255 192.168.10.3 0.0.0.0
        - access-list 100 permit udp 192.168.10.0 0.0.0.255 192.168.10.3 0.0.0.255
    register: acl


- name: ACL Configuration for R2 (For Students and Faculty)
  hosts: R2
  gather_facts: false
  connection: local

  tasks:
  - name: R2 Access List command
    ios_command:
      commands:
        - configure terminal
        - access-list 101 permit tcp 192.168.10.0 0.0.0.255 192.168.10.3 0.0.0.0
        - access-list 101 permit udp 192.168.10.0 0.0.0.255 192.168.10.3 0.0.0.255
        - access-list 101 permit tcp 192.168.20.0 0.0.0.255 192.168.20.3 0.0.0.0
        - access-list 101 permit udp 192.168.20.0 0.0.0.255 192.168.20.3 0.0.0.255
    register: acl
```


ansible.cfg
ospf-config-playbook.yaml
acl-config-playbook.yaml
casestudy > acl-config-playbook.yaml
Reinhart Wilson L. Regulacion

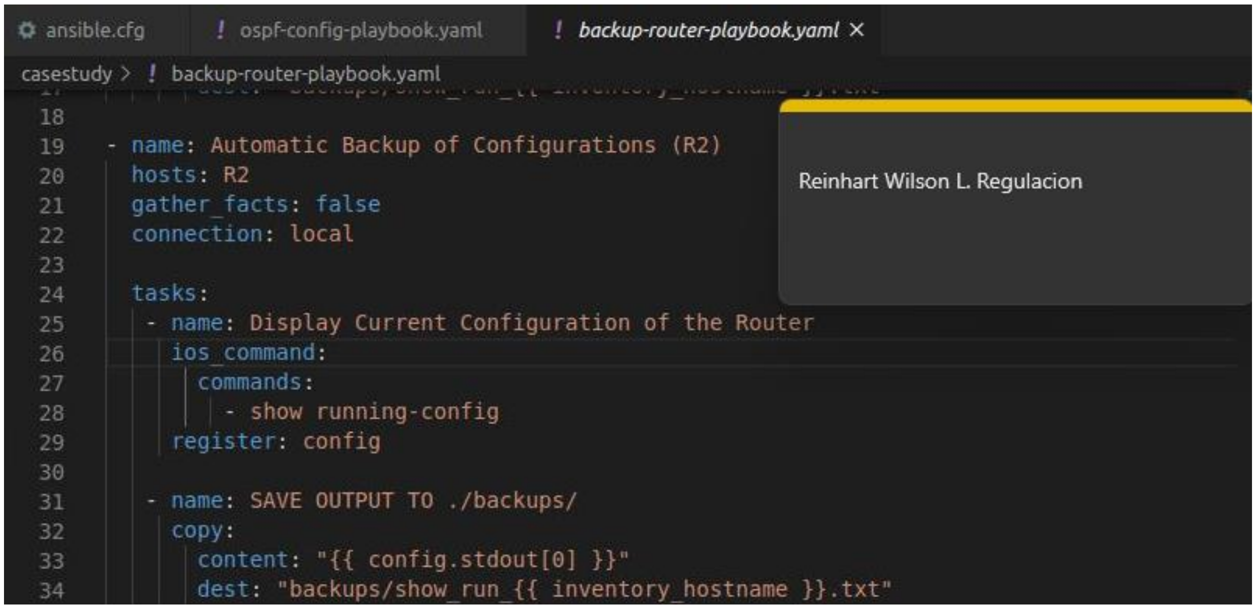
```
---
- name: ACL Configuration for R1 (For Faculty Only)
  hosts: R1
  gather_facts: false
  connection: local

  tasks:
    - name: R1 Access List comman
      ios_command:
        commands:
          - configure terminal
          - access-list 100 permit tcp 192.168.10.0 0.0.0.255 192.168.10.3 0.0.0.0
          - access-list 100 permit udp 192.168.10.0 0.0.0.255 192.168.10.3 0.0.0.255
      register: acl


- name: ACL Configuration for R2 (For Students and Faculty)
```

Figure 9. ACL ansible file

Step 6: Create a yaml file named “backup_router_playbook” to backup the running configurations of the two routers.

Code lines:

```
---
- name: Automatic Backup of Configurations (R1)
  hosts: R1
  gather_facts: false
  connection: local

  tasks:
   - name: Display Current Configuration of the Router
     ios_command:
       commands:
         - show running-config
     register: config

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

- name: Automatic Backup of Configurations (R2)
  hosts: R2
  gather_facts: false
  connection: local

  tasks:
   - name: Display Current Configuration of the Router
     ios_command:
       commands:
         - show running-config
     register: config

   - name: SAVE OUTPUT TO ./backups/
     copy:
       content: "{{ config.stdout[0] }}"
       dest: "backups/show_run_{{ inventory_hostname }}.txt"
```



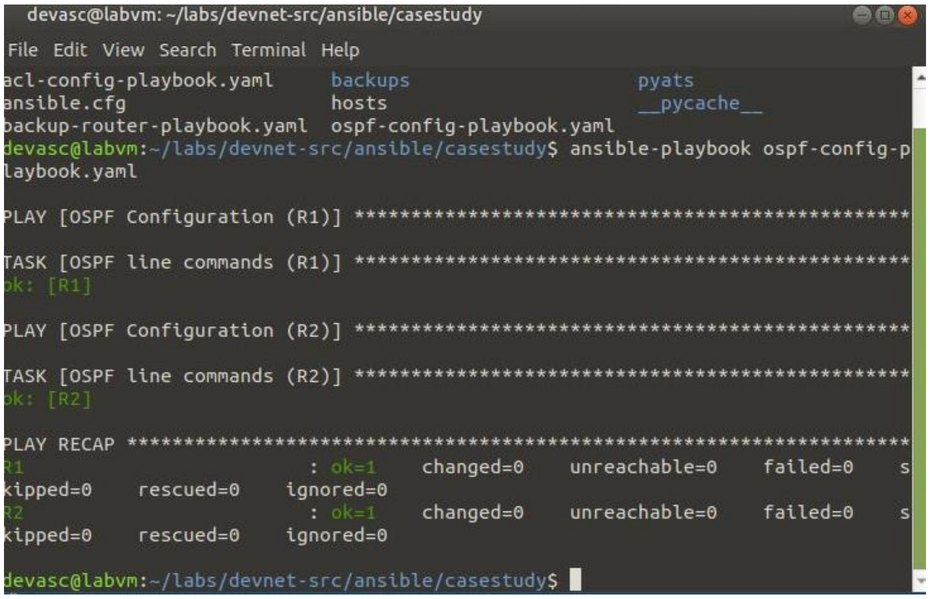
```
  - name: Automatic Backup of Configurations (R2)
    hosts: R2
    gather_facts: false
    connection: local

    tasks:
      - name: Display Current Configuration of the Router
        ios_command:
          commands:
            - show running-config
        register: config

      - name: SAVE OUTPUT TO ./backups/
        copy:
          content: "{{ config.stdout[0] }}"
          dest: "backups/show_run_{{ inventory_hostname }}.txt"
```

ansible.cfg
ospf-config-playbook.yaml
backup-router-playbook.yaml
casestudy > backup-router-playbook.yaml
Reinhart Wilson L. Regulacion

Figure 10. Backup config ansible file

Part 4. Executing and checking the outputs from the Ansible Playbook yaml files

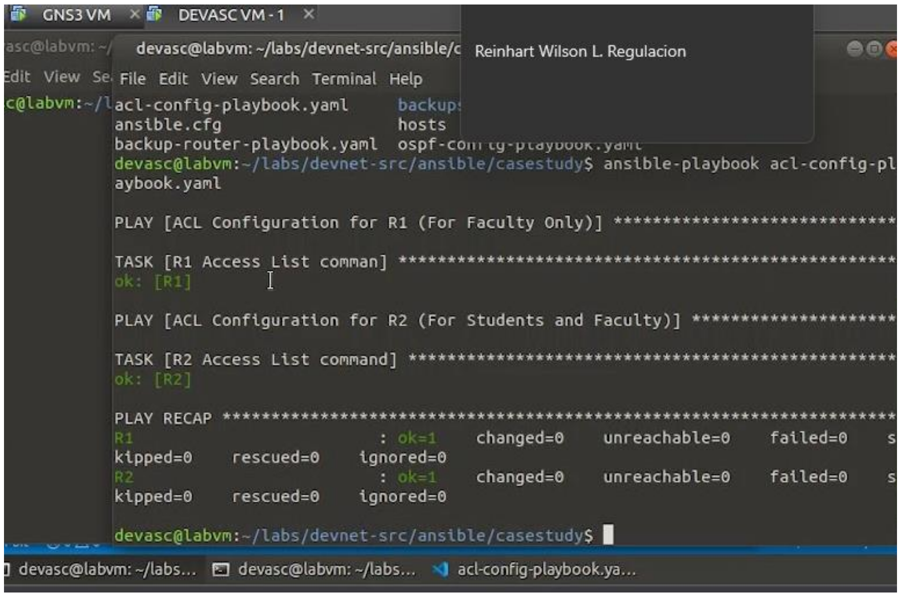
Step 1: Checking the results of “ospf_config_playbook” yaml file



```
devasc@labvm: ~/labs/devnet-src/ansible/casestudy
File  Edit  View  Search  Terminal  Help
acl-config-playbook.yaml     backups                  pyats
ansible.cfg                  hosts                    __pycache__
backup-router-playbook.yaml  ospf-config-playbook.yaml
devasc@labvm:~/labs/devnet-src/ansible/casestudy$ ansible-playbook ospf-config-p
laybook.yaml

PLAY [OSPF Configuration (R1)] ************************************************

TASK [OSPF line commands (R1)] ************************************************
ok: [R1]

PLAY [OSPF Configuration (R2)] ************************************************

TASK [OSPF line commands (R2)] ************************************************
ok: [R2]

PLAY RECAP ********************************************************************
R1                         : ok=1    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R2                         : ok=1    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

devasc@labvm:~/labs/devnet-src/ansible/casestudy$
```

Figure 11. OSPF playbook output

Step 2: Checking the results of “acl_config_playbook” yaml file

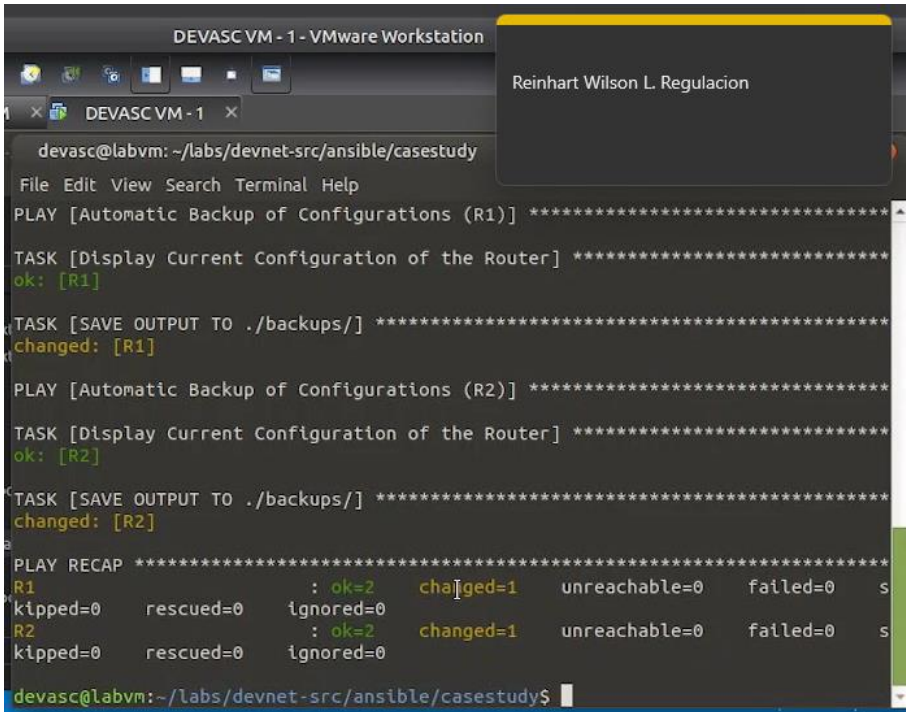


GNS3 VM
DEVASC VM - 1
devasc@labvm: ~/labs/devnet-src/ansible/c
Reinhart Wilson L. Regulacion
File Edit View Search Terminal Help

```
acl-config-playbook.yaml     backups
ansible.cfg                  hosts
backup-router-playbook.yaml  ospf-config-playbook.yaml
devasc@labvm:~/labs/devnet-src/ansible/casestudy$ ansible-playbook acl-config-pl
aybook.yaml

PLAY [ACL Configuration for R1 (For Faculty Only)] ****************************

TASK [R1 Access List comman] **************************************************
ok: [R1]

PLAY [ACL Configuration for R2 (For Students and Faculty)] ********************

TASK [R2 Access List command] *************************************************
ok: [R2]

PLAY RECAP ********************************************************************
R1                         : ok=1    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R2                         : ok=1    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

devasc@labvm:~/labs/devnet-src/ansible/casestudy$
```

Figure 12. ACL Ansible results

Step 3: Checking the results of “backup_router_playbook” yaml file



DEVASC VM - 1 - VMware Workstation
Reinhart Wilson L. Regulacion
DEVASC VM - 1
devasc@labvm: ~/labs/devnet-src/ansible/casestudy
File Edit View Search Terminal Help

```
PLAY [Automatic Backup of Configurations (R1)] ********************************

TASK [Display Current Configuration of the Router] ****************************
ok: [R1]

TASK [SAVE OUTPUT TO ./backups/] **********************************************
changed: [R1]

PLAY [Automatic Backup of Configurations (R2)] ********************************

TASK [Display Current Configuration of the Router] ****************************
ok: [R2]

TASK [SAVE OUTPUT TO ./backups/] **********************************************
changed: [R2]

PLAY RECAP ********************************************************************
R1                         : ok=2    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R2                         : ok=2    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

devasc@labvm:~/labs/devnet-src/ansible/casestudy$
```

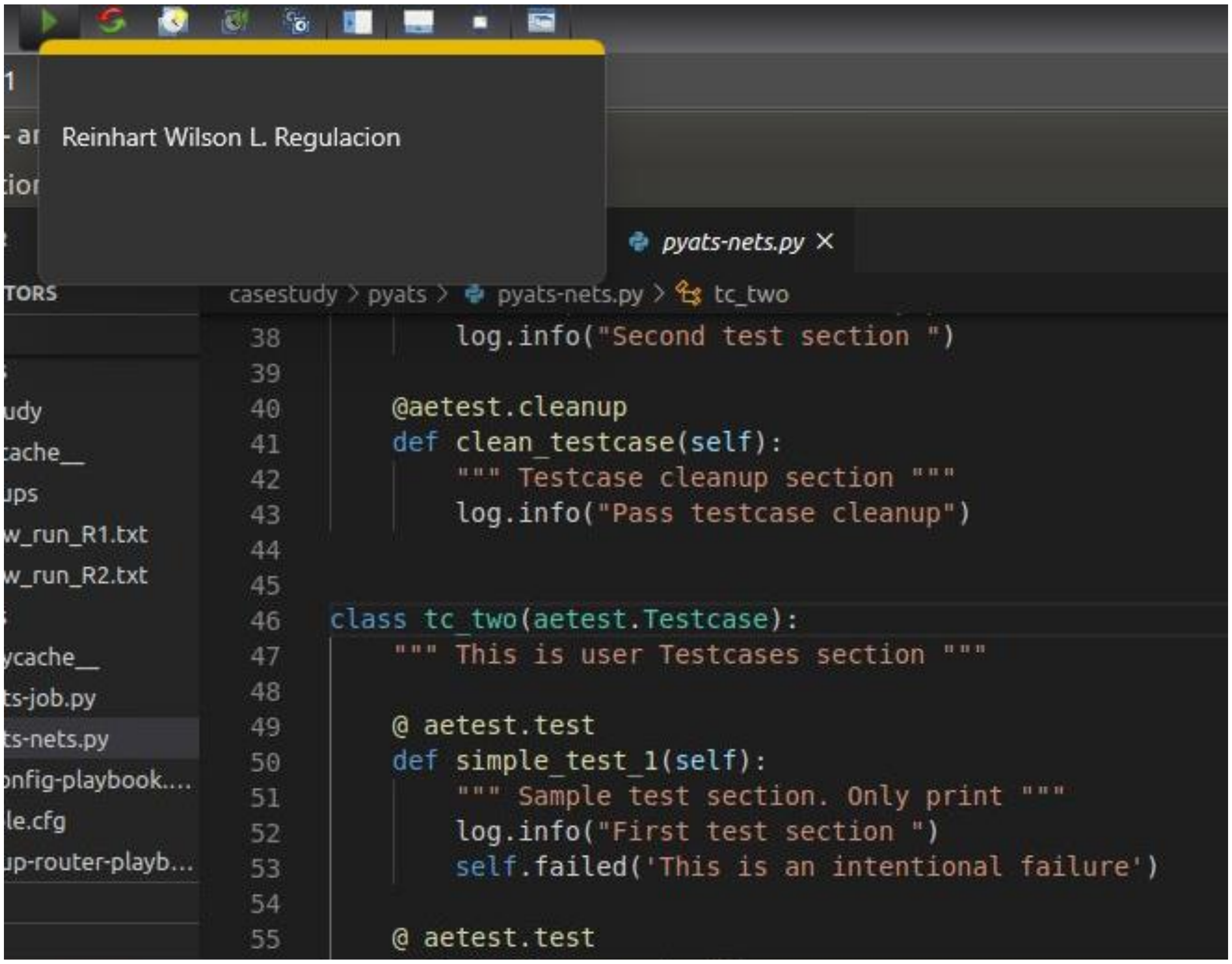
Figure 13. Backup configuration Ansible results

Part 5. Using PyATS to automate the testing of the config files

Step 1: Create a PyATS script

```
import logging
```

```
from pyats import aetest

log = logging.getLogger(__name__)

class common_setup(aetest.CommonSetup):
    """ Common Setup section """

    @aetest.subsection
    def sample_subsection_1(self):
        """ Common Setup subsection """
        log.info("Aetest Common Setup ")

    @aetest.subsection
    def sample_subsection_2(self, section):
        """ Common Setup subsection """
        log.info("Inside %s" % (section))

        log.info("Inside class %s" % (self.uid))

class tc_one(aetest.Testcase):
    """ This is user Testcases section """

    @aetest.setup
    def prepare_testcase(self, section):
        """ Testcase Setup section """
        log.info("Preparing the test")
        log.info(section)

    @ aetest.test
    def simple_test_1(self):
        """ Sample test section. Only print """
        log.info("First test section ")

    @ aetest.test
    def simple_test_2(self):
        """ Sample test section. Only print """
        log.info("Second test section ")

    @aetest.cleanup
    def clean_testcase(self):
        """ Testcase cleanup section """
        log.info("Pass testcase cleanup")


class tc_two(aetest.Testcase):
    """ This is user Testcases section """
    @ aetest.test
    def simple_test_1(self):
        """ Sample test section. Only print """
        log.info("First test section ")
        self.failed('This is an intentional failure')

    @ aetest.test
    def simple_test_2(self):
        """ Sample test section. Only print """
        log.info("Second test section ")

    @aetest.cleanup
    def clean_testcase(self):
        """ Testcase cleanup section """
        log.info("Pass testcase cleanup")

class common_cleanup(aetest.CommonCleanup):
    """ Common Cleanup for Sample Test """

    @aetest.subsection
    def clean_everything(self):
        """ Common Cleanup Subsection """
        log.info("Aetest Common Cleanup ")

if __name__ == '__main__':
    result = aetest.main()
```

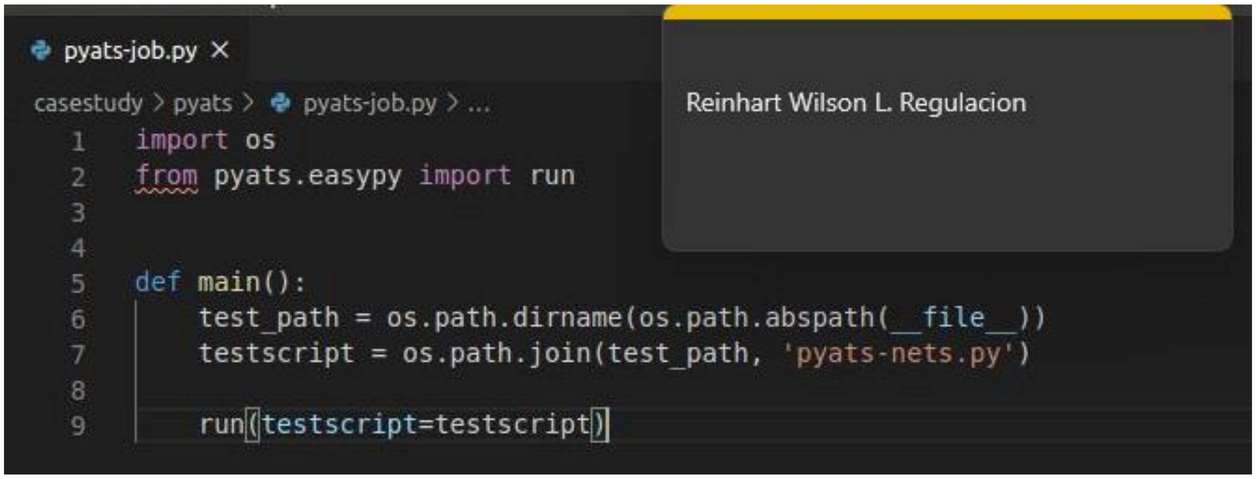


```
            log.info("Second test section ")

    @aetest.cleanup
    def clean_testcase(self):
        """ Testcase cleanup section """
        log.info("Pass testcase cleanup")


class tc_two(aetest.Testcase):
    """ This is user Testcases section """

    @ aetest.test
    def simple_test_1(self):
        """ Sample test section. Only print """
        log.info("First test section ")
        self.failed('This is an intentional failure')

    @ aetest.test
```

Figure 14. PyATS script file

Step 2: Create a PyATS job file

```
import os
from pyats.easypy import run


def main():
    test_path = os.path.dirname(os.path.abspath(__file__))
    testscript = os.path.join(test_path, 'pyats-nets.py')

    run(testscript=testscript)
```



pyats-job.py

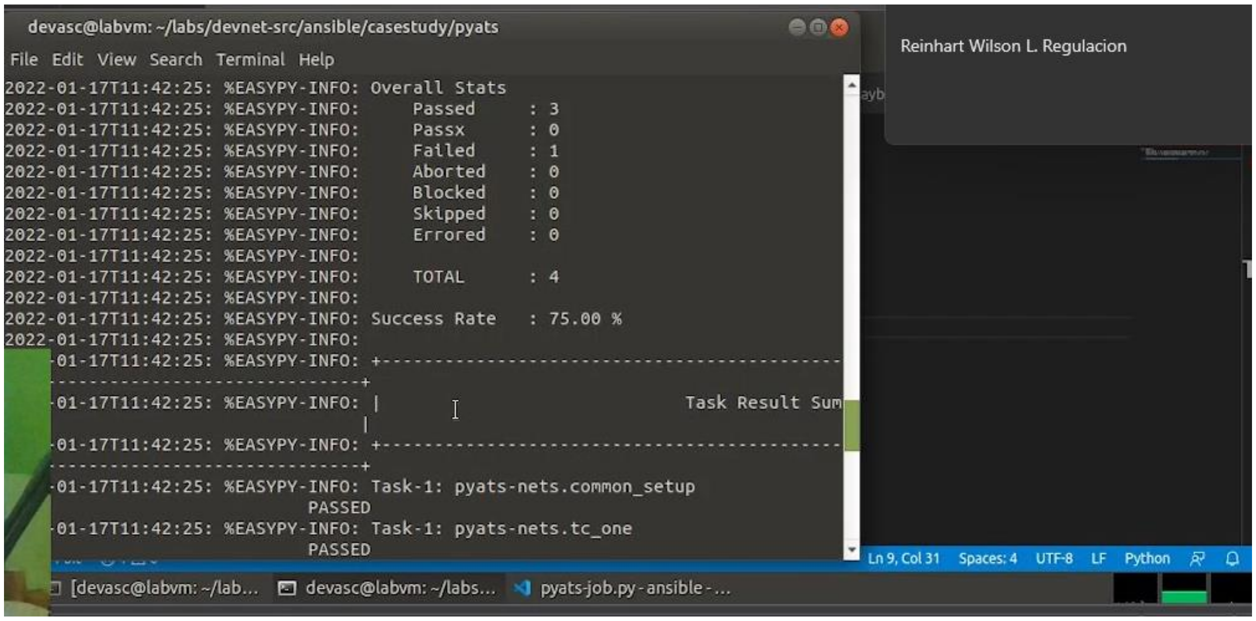
casestudy > pyats > pyats-job.py > ...

Reinhart Wilson L. Regulacion

```python
import os
from pyats.easypy import run


def main():
    test_path = os.path.dirname(os.path.abspath(__file__))
    testscript = os.path.join(test_path, 'pyats-nets.py')

    run(testscript=testscript)
```

Figure 15. PyATS job file

Step 3: Test the python scripts for network automation



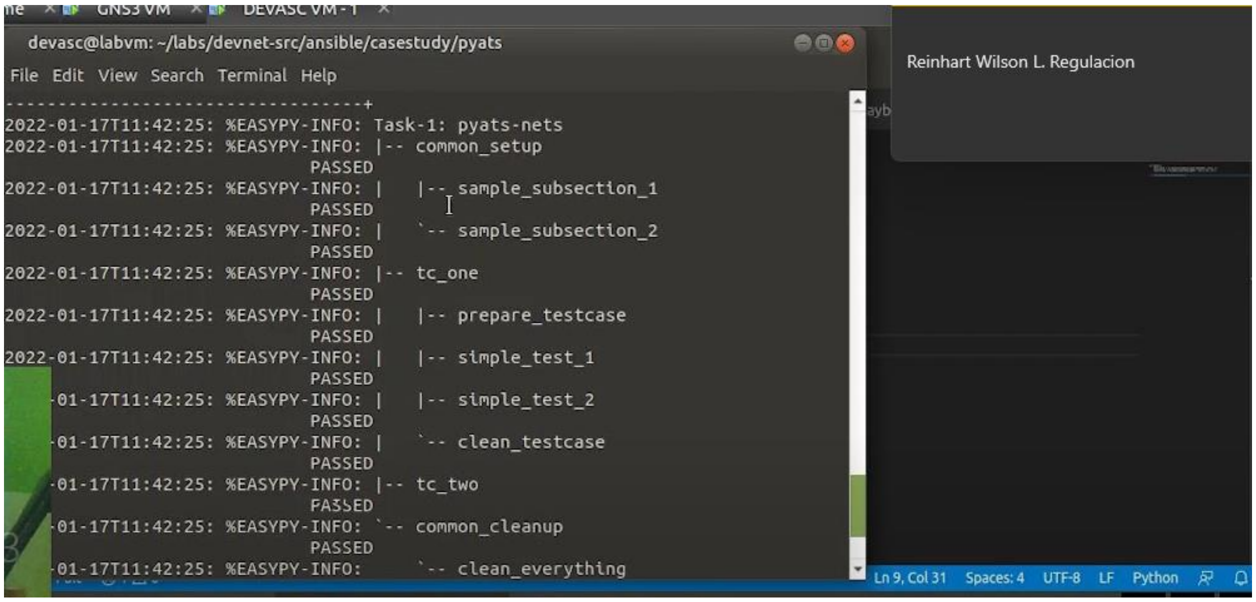
devasc@labvm: ~/labs/devnet-src/ansible/casestudy/pyats

Reinhart Wilson L. Regulacion

```
2022-01-17T11:42:25: %EASYPY-INFO: Overall Stats
2022-01-17T11:42:25: %EASYPY-INFO:     Passed     : 3
2022-01-17T11:42:25: %EASYPY-INFO:     Passx      : 0
2022-01-17T11:42:25: %EASYPY-INFO:     Failed     : 1
2022-01-17T11:42:25: %EASYPY-INFO:     Aborted    : 0
2022-01-17T11:42:25: %EASYPY-INFO:     Blocked    : 0
2022-01-17T11:42:25: %EASYPY-INFO:     Skipped    : 0
2022-01-17T11:42:25: %EASYPY-INFO:     Errored    : 0
2022-01-17T11:42:25: %EASYPY-INFO:
2022-01-17T11:42:25: %EASYPY-INFO:     TOTAL      : 4
2022-01-17T11:42:25: %EASYPY-INFO:
2022-01-17T11:42:25: %EASYPY-INFO: Success Rate   : 75.00 %
2022-01-17T11:42:25: %EASYPY-INFO:
-01-17T11:42:25: %EASYPY-INFO: +------------------------------------------
----------------------------+
-01-17T11:42:25: %EASYPY-INFO: |                              Task Result Sum
                            |
-01-17T11:42:25: %EASYPY-INFO: +------------------------------------------
----------------------------+
-01-17T11:42:25: %EASYPY-INFO: Task-1: pyats-nets.common_setup
                          PASSED
-01-17T11:42:25: %EASYPY-INFO: Task-1: pyats-nets.tc_one
                          PASSED
```

File Edit View Search Terminal Help
Ln 9, Col 31 Spaces: 4 UTF-8 LF Python
[devasc@labvm: ~/lab...
devasc@labvm: ~/labs...
pyats-job.py - ansible - ...

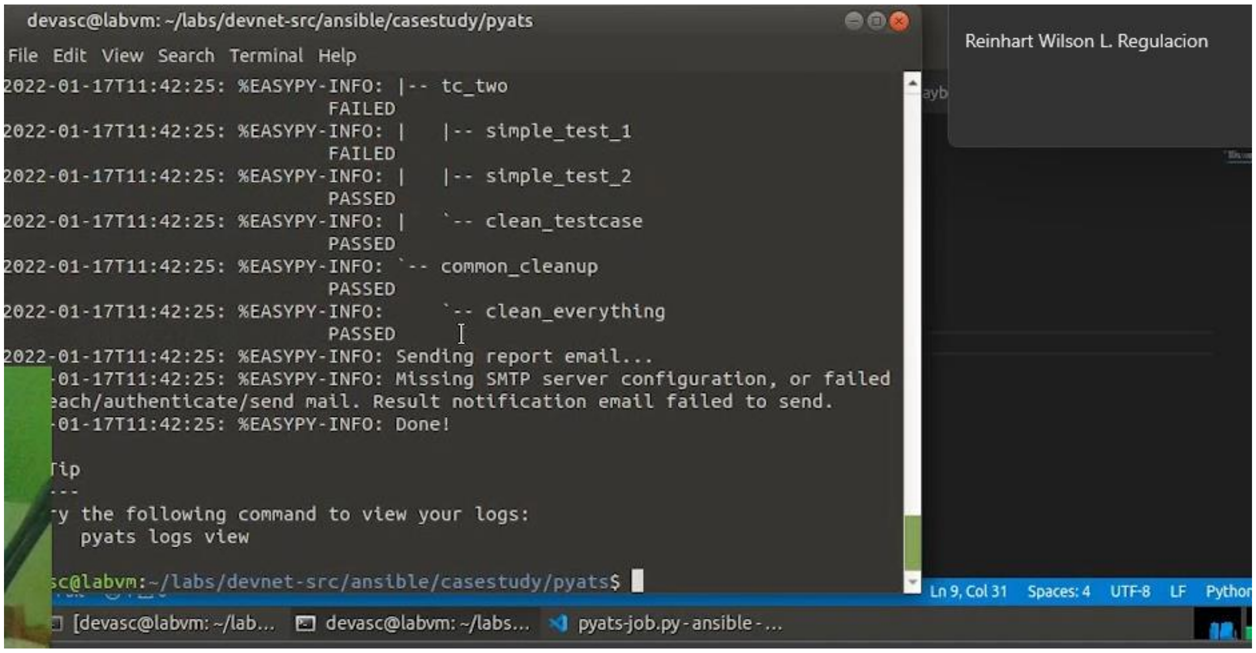
Figure 16. PyATS output (part 1)



devasc@labvm: ~/labs/devnet-src/ansible/casestudy/pyats
File Edit View Search Terminal Help
Reinhart Wilson L. Regulacion
Ln 9, Col 31 Spaces: 4 UTF-8 LF Python

```
--------------------------------+
2022-01-17T11:42:25: %EASYPY-INFO: Task-1: pyats-nets
2022-01-17T11:42:25: %EASYPY-INFO: |-- common_setup
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: |   |-- sample_subsection_1
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: |   `-- sample_subsection_2
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: |-- tc_one
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: |   |-- prepare_testcase
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: |   |-- simple_test_1
                                 PASSED
-01-17T11:42:25: %EASYPY-INFO: |   |-- simple_test_2
                                 PASSED
-01-17T11:42:25: %EASYPY-INFO: |   `-- clean_testcase
                                 PASSED
-01-17T11:42:25: %EASYPY-INFO: |-- tc_two
                                 PASSED
-01-17T11:42:25: %EASYPY-INFO: `-- common_cleanup
                                 PASSED
-01-17T11:42:25: %EASYPY-INFO:     `-- clean_everything
```

Figure 17. PyATS output (part 2)



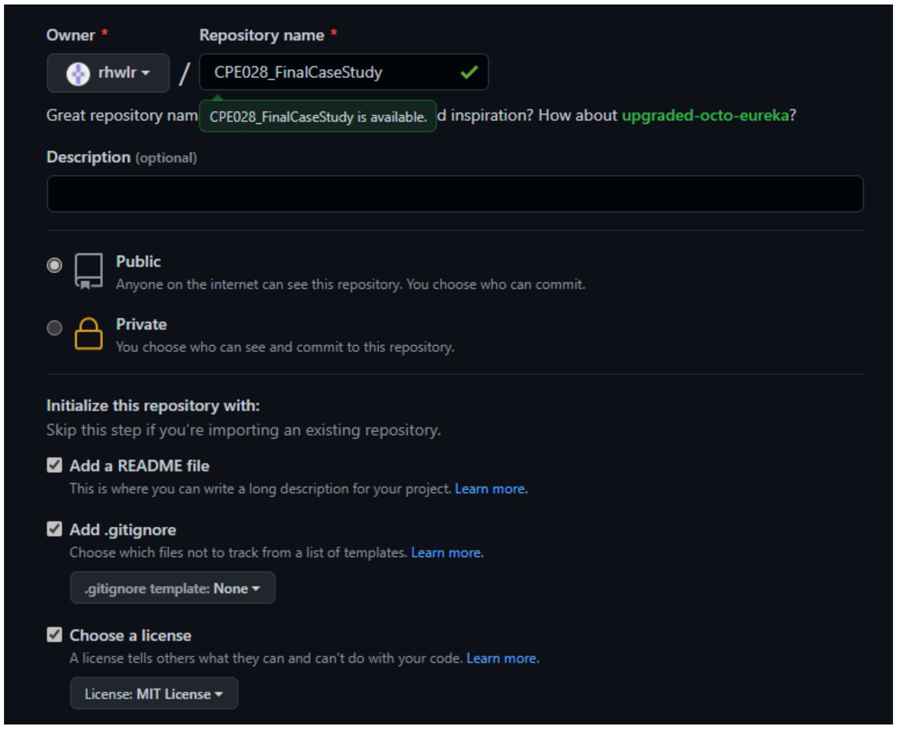
devasc@labvm: ~/labs/devnet-src/ansible/casestudy/pyats
File Edit View Search Terminal Help
Reinhart Wilson L. Regulacion
Ln 9, Col 31 Spaces: 4 UTF-8 LF Python
[devasc@labvm: ~/lab...
devasc@labvm: ~/labs...
pyats-job.py - ansible - ...

```
2022-01-17T11:42:25: %EASYPY-INFO: |-- tc_two
                                 FAILED
2022-01-17T11:42:25: %EASYPY-INFO: |   |-- simple_test_1
                                 FAILED
2022-01-17T11:42:25: %EASYPY-INFO: |   |-- simple_test_2
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: |   `-- clean_testcase
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: `-- common_cleanup
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO:     `-- clean_everything
                                 PASSED
2022-01-17T11:42:25: %EASYPY-INFO: Sending report email...
-01-17T11:42:25: %EASYPY-INFO: Missing SMTP server configuration, or failed
each/authenticate/send mail. Result notification email failed to send.
-01-17T11:42:25: %EASYPY-INFO: Done!

Tip
---
ry the following command to view your logs:
  pyats logs view

sc@labvm:~/labs/devnet-src/ansible/casestudy/pyats$
```

Figure 18. PyATS output (part 3)

Part 6. GitHub integration for file repository, version control, and collaboration

Step 1: Create a GitHub repository


Owner *
rhwlr
Repository name *
CPE028_FinalCaseStudy
Great repository nam CPE028_FinalCaseStudy is available. d inspiration? How about upgraded-octo-eureka?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

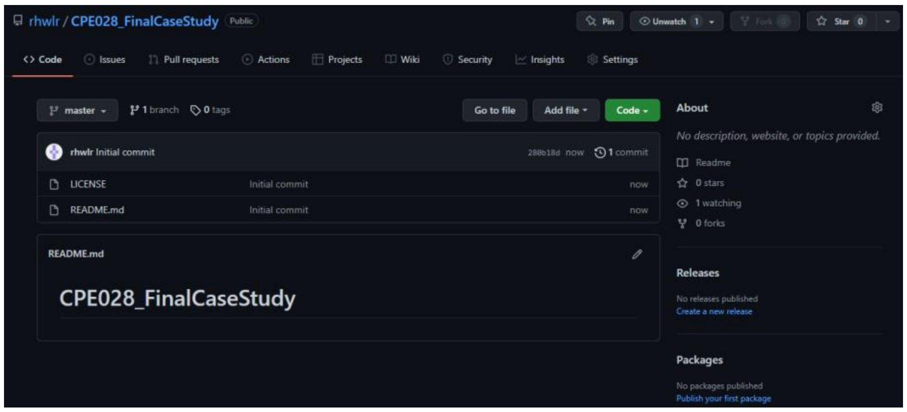
Initialize this repository with:
Skip this step if you're importing an existing repository.

[✓] Add a README file
This is where you can write a long description for your project. Learn more.

[✓] Add .gitignore
Choose which files not to track from a list of templates. Learn more.
.gitignore template: None

[✓] Choose a license
A license tells others what they can and can't do with your code. Learn more.
License: MIT License

Figure 19. Creating a GitHub repository (part 1)



rhwlr / CPE028_FinalCaseStudy Public

master 1 branch 0 tags

rhwlr Initial commit 280b18d now 1 commit
LICENSE Initial commit now
README.md Initial commit now

README.md
CPE028_FinalCaseStudy

About
No description, website, or topics provided.
Readme
0 stars
1 watching
0 forks

Releases
No releases published
Create a new release

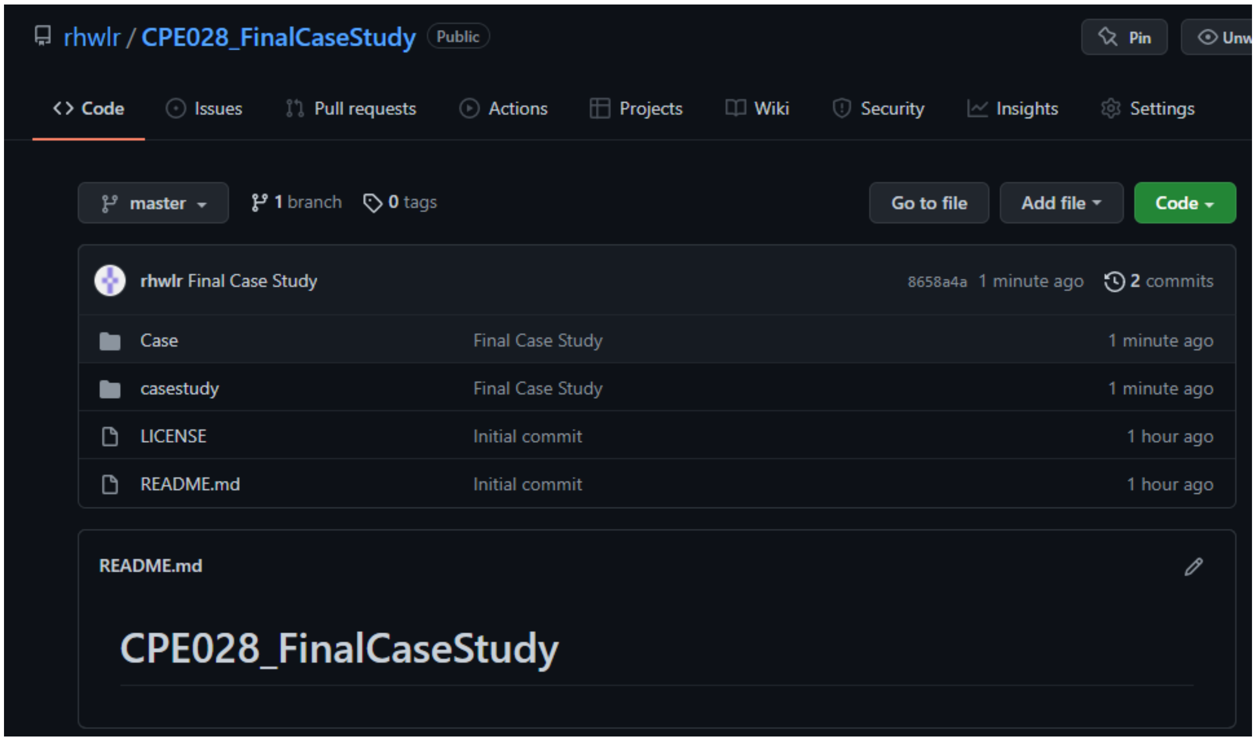
Packages
No packages published
Publish your first package

Figure 20. Creating a GitHub repository (part 2)

Step 2: Upload local files and repository to a remote repository.

- git init
- git remote add origin
- git add -A
- git commit -m “”
- git push -u origin master
- enter username and password

Step 3: Check the repository



rhwlr Final Case Study 8658a4a 1 minute ago 2 commits

Case	Final Case Study	1 minute ago
casestudy	Final Case Study	1 minute ago
LICENSE	Initial commit	1 hour ago
README.md	Initial commit	1 hour ago

README.md

CPE028_FinalCaseStudy

Figure 21. GitHub repository

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