CSCI 5180 – Network Management and Automation

Lab 1
Network management using SNMP and NMAP

University of Colorado Boulder
Department of Computer Science

Professor Levi Perigo, Ph.D.

Summary

SNMP is used widely by network and system administrators to monitor the health and metrics of a diverse array of network devices.

The objectives in this lab will enable you to understand how different SNMP versions work, gather operational statistics and monitor your network using simple commands, and modify parameters remotely on SNMP agents.

Pre-Lab

You will need the following commands to enable SNMP on the Cisco router in the VM's GNS3. (Note: Use the instructions from Lab 0 for gaining access to the VM and GNS3 setup.)

- Run the simulation by clicking on the Play button in GNS3.

- Console into the router, check if SNMP is running using **show snmp host**.

If SNMP is not enabled, follow these steps to configure SNMP host on a Cisco router:

- Enable SNMP traps on the router by entering: (config)#**snmp-server enable traps**

- Assign an IP address (make suree it is in a different subnet than the primary interface, use any private subnet) to the 2nd interface of the router that you added & bring the interface up.

- Enter configuration commands, one per line. End with CNTL/Z.

    (config)# **snmp-server host 198.51.100.2 public**

    (config)# **snmp-server community public rw**

    *Note: The "snmp-server host" IP address is the IP address of the VM terminal. Thus, in this example the IP address would be 198.51.100.2.

On the terminal of the VM start Wireshark and monitor the tap0 interface.

Next type the below commands in the VM terminal and check the output (you can receive SNMP data from the router using **SNMPGET/SNMPWALK**).

```
netman@netman:~$ snmpget -v 1 -c public 198.51.100.3 ifName.1
IF-MIB::ifName.1 = STRING: Fa0/0 ------( This is the output )
```

```
netman@netman:~$ snmpget -v 1 -c public 198.51.100.3 .1.3.6.1.2.1.2.1.0
```

IF-MIB::ifNumber.0 = INTEGER: 5 ------(This is the output)

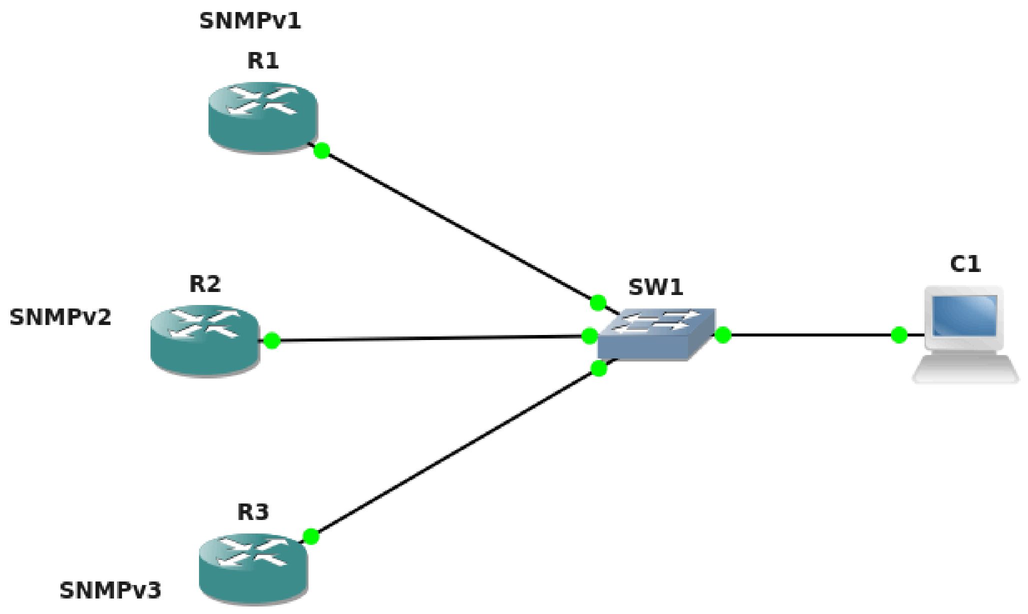
*NOTE: The IP address used within the terminal is the IP address of the Cisco router. In this example the Cisco router has the IP address of 198.51.100.3.

You should be able to see a similar output on the terminal as well as an SNMP packet on Wireshark.

Objective 1: Configuring SNMP on Cisco IOS

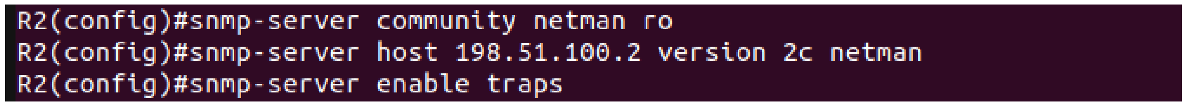
Create the topology in GNS3 as shown below and assign management IPs (198.51.100.0/24 subnet) to them on fa0/0. Configure the nodes for different versions of SNMP & enable traps.

- R1: SNMPv1 (Already configured)
- R2: SNMPv2
- R3: SNMPv3



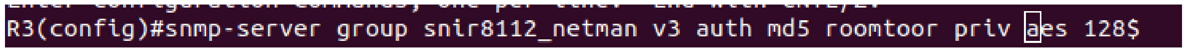
1. How did you configure SNMPv2 and v3 on routers R2 and R3? Provide running configuration screenshots (only portions relevant to SNMP). **[10 points]**

R2



```
R2(config)#snmp-server community netman ro
R2(config)#snmp-server host 198.51.100.2 version 2c netman
R2(config)#snmp-server enable traps
```

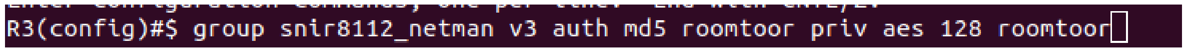
R3



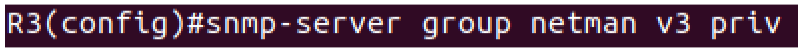
```
R3(config)#snmp-server group snir8112_netman v3 auth md5 roomtoor priv aes 128$
```

…

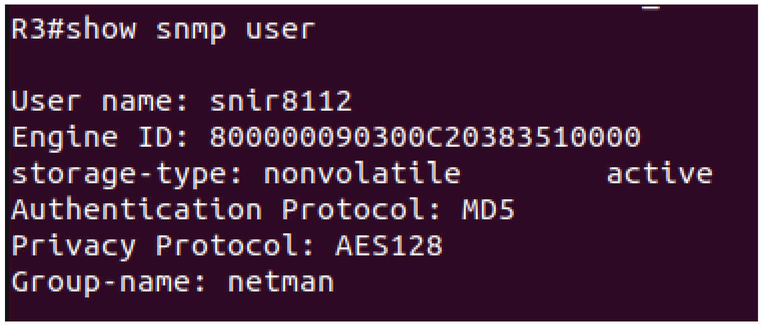
continuation of the screenshot…



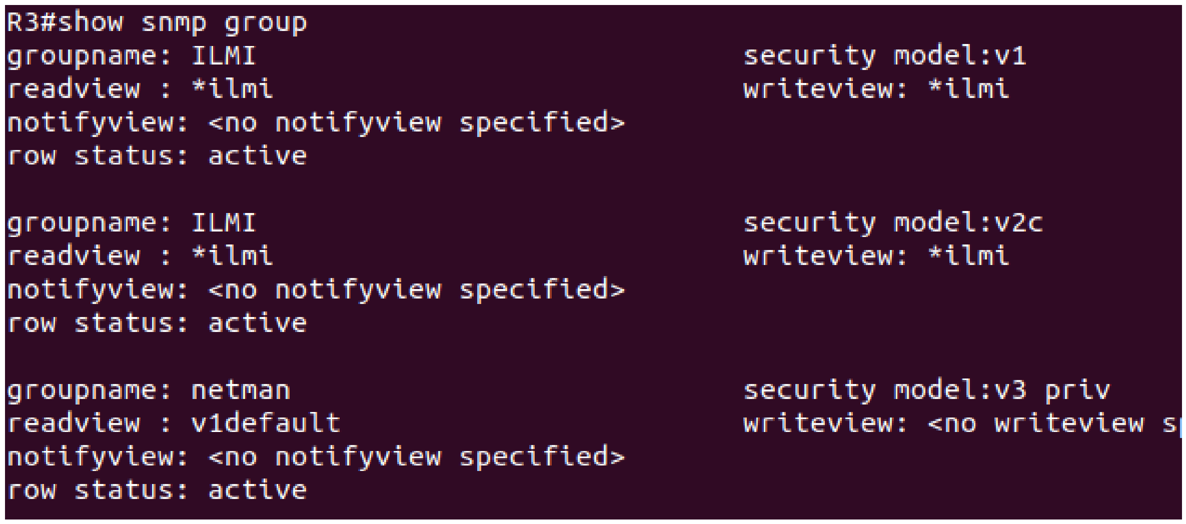
```
R3(config)#$ group snir8112_netman v3 auth md5 roomtoor priv aes 128 roomtoor
```



```
R3(config)#snmp-server group netman v3 priv
```



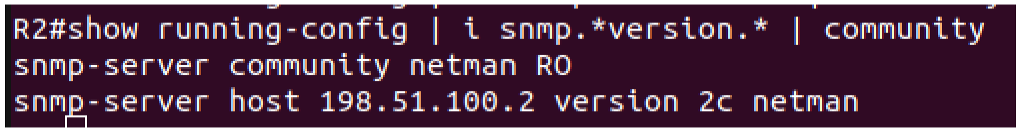
```
R3#show snmp user

User name: snir8112
Engine ID: 800000090300C20383510000
storage-type: nonvolatile        active
Authentication Protocol: MD5
Privacy Protocol: AES128
Group-name: netman
```



```
R3#show snmp group
groupname: ILMI                                 security model:v1
readview : *ilmi                                writeview: *ilmi
notifyview: <no notifyview specified>
row status: active

groupname: ILMI                                 security model:v2c
readview : *ilmi                                writeview: *ilmi
notifyview: <no notifyview specified>
row status: active

groupname: netman                               security model:v3 priv
readview : v1default                            writeview: <no writeview s
notifyview: <no notifyview specified>
row status: active
```

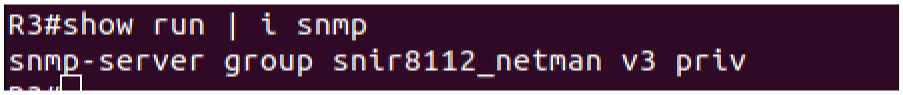
Running configuration

R2:



```
R2#show running-config | i snmp.*version.* | community
snmp-server community netman RO
snmp-server host 198.51.100.2 version 2c netman
```

R3:

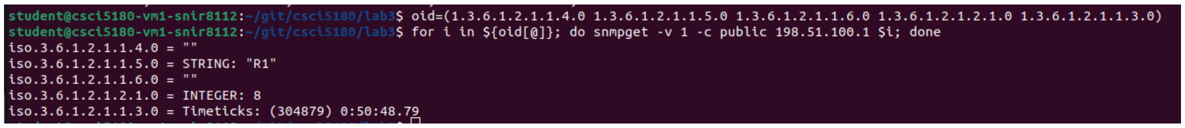

```
R3#show run | i snmp
snmp-server group snir8112_netman v3 priv
```

Objective 2: SNMPGET and Dashboard

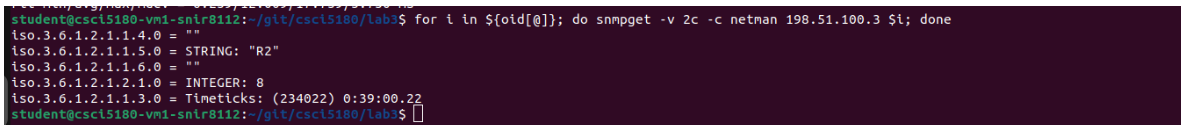
The list of OIDs that need to be fetched from the routers:
sysContact = 1.3.6.1.2.1.1.4.0
sysName = 1.3.6.1.2.1.1.5.0
sysLocation = 1.3.6.1.2.1.1.6.0
ifNumber = 1.3.6.1.2.1.2.1.0
sysUptime = 1.3.6.1.2.1.1.3.0
Sample command to run on terminal:
snmpget -v 1 -c public 198.51.100.3 .1.3.6.1.2.1.1.4.0

1. Enter the above SNMPGET commands for the OIDs mentioned for SNMP v1, v2, and v3. Paste relevant screenshots. [**10 points**]

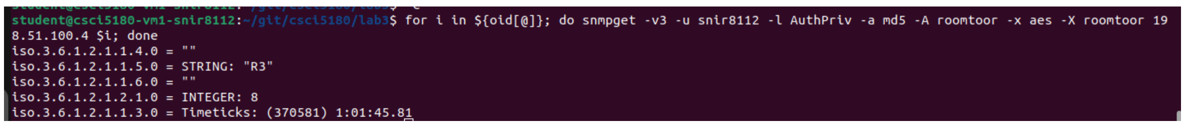
SNMP v1:


```
student@csci5180-vm1-snir8112:~/git/csci5180/lab3$ oid=(1.3.6.1.2.1.1.4.0 1.3.6.1.2.1.1.5.0 1.3.6.1.2.1.1.6.0 1.3.6.1.2.1.2.1.0 1.3.6.1.2.1.1.3.0)
student@csci5180-vm1-snir8112:~/git/csci5180/lab3$ for i in ${oid[@]}; do snmpget -v 1 -c public 198.51.100.1 $i; done
iso.3.6.1.2.1.1.4.0 = ""
iso.3.6.1.2.1.1.5.0 = STRING: "R1"
iso.3.6.1.2.1.1.6.0 = ""
iso.3.6.1.2.1.2.1.0 = INTEGER: 8
iso.3.6.1.2.1.1.3.0 = Timeticks: (304879) 0:50:48.79
```

SNMP v2:

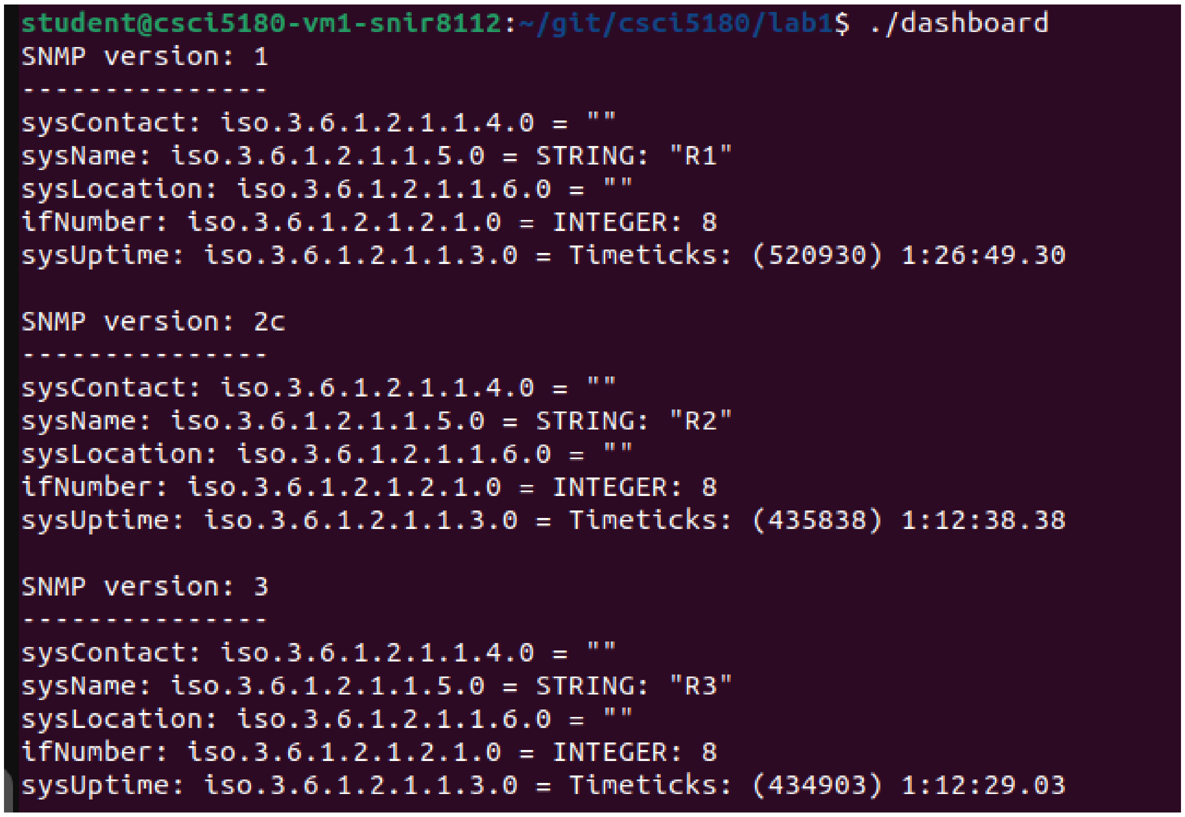

```
student@csci5180-vm1-snir8112:~/git/csci5180/lab3$ for i in ${oid[@]}; do snmpget -v 2c -c netman 198.51.100.3 $i; done
iso.3.6.1.2.1.1.4.0 = ""
iso.3.6.1.2.1.1.5.0 = STRING: "R2"
iso.3.6.1.2.1.1.6.0 = ""
iso.3.6.1.2.1.2.1.0 = INTEGER: 8
iso.3.6.1.2.1.1.3.0 = Timeticks: (234022) 0:39:00.22
student@csci5180-vm1-snir8112:~/git/csci5180/lab3$ 
```

SNMP v3:


```
student@csci5180-vm1-snir8112:~/git/csci5180/lab3$ for i in ${oid[@]}; do snmpget -v3 -u snir8112 -l AuthPriv -a md5 -A roomtoor -x aes -X roomtoor 19
8.51.100.4 $i; done
iso.3.6.1.2.1.1.4.0 = ""
iso.3.6.1.2.1.1.5.0 = STRING: "R3"
iso.3.6.1.2.1.1.6.0 = ""
iso.3.6.1.2.1.2.1.0 = INTEGER: 8
iso.3.6.1.2.1.1.3.0 = Timeticks: (370581) 1:01:45.81
```

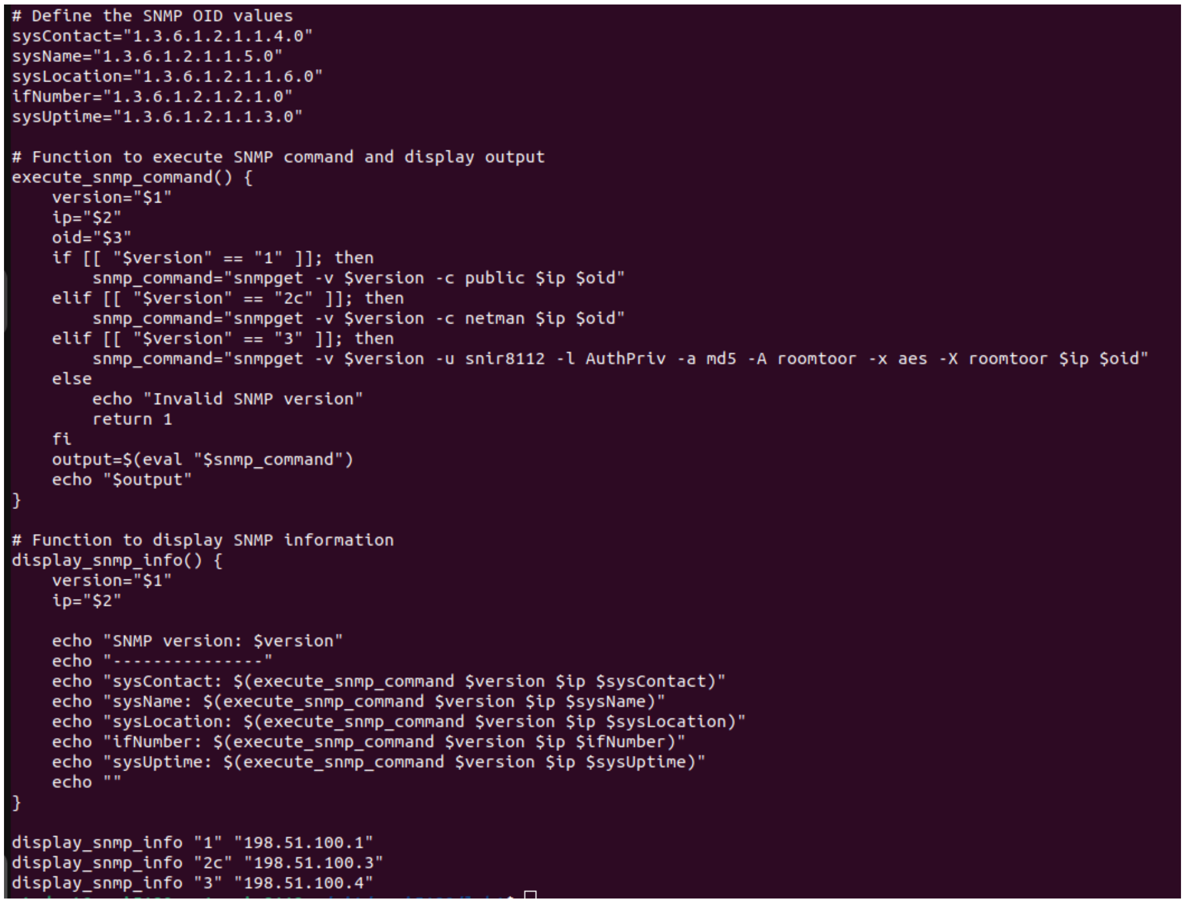
2. Create a dashboard to display the output from those commands using UNIX/Python. Paste relevant screenshots. [**15 points**]

Output:



```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ ./dashboard
SNMP version: 1
---------------
sysContact: iso.3.6.1.2.1.1.4.0 = ""
sysName: iso.3.6.1.2.1.1.5.0 = STRING: "R1"
sysLocation: iso.3.6.1.2.1.1.6.0 = ""
ifNumber: iso.3.6.1.2.1.2.1.0 = INTEGER: 8
sysUptime: iso.3.6.1.2.1.1.3.0 = Timeticks: (520930) 1:26:49.30

SNMP version: 2c
---------------
sysContact: iso.3.6.1.2.1.1.4.0 = ""
sysName: iso.3.6.1.2.1.1.5.0 = STRING: "R2"
sysLocation: iso.3.6.1.2.1.1.6.0 = ""
ifNumber: iso.3.6.1.2.1.2.1.0 = INTEGER: 8
sysUptime: iso.3.6.1.2.1.1.3.0 = Timeticks: (435838) 1:12:38.38

SNMP version: 3
---------------
sysContact: iso.3.6.1.2.1.1.4.0 = ""
sysName: iso.3.6.1.2.1.1.5.0 = STRING: "R3"
sysLocation: iso.3.6.1.2.1.1.6.0 = ""
ifNumber: iso.3.6.1.2.1.2.1.0 = INTEGER: 8
sysUptime: iso.3.6.1.2.1.1.3.0 = Timeticks: (434903) 1:12:29.03
```

Code:

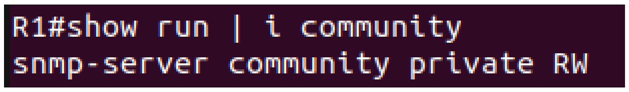


```
# Define the SNMP OID values
sysContact="1.3.6.1.2.1.1.4.0"
sysName="1.3.6.1.2.1.1.5.0"
sysLocation="1.3.6.1.2.1.1.6.0"
ifNumber="1.3.6.1.2.1.2.1.0"
sysUptime="1.3.6.1.2.1.1.3.0"

# Function to execute SNMP command and display output
execute_snmp_command() {
    version="$1"
    ip="$2"
    oid="$3"
    if [[ "$version" == "1" ]]; then
        snmp_command="snmpget -v $version -c public $ip $oid"
    elif [[ "$version" == "2c" ]]; then
        snmp_command="snmpget -v $version -c netman $ip $oid"
    elif [[ "$version" == "3" ]]; then
        snmp_command="snmpget -v $version -u snir8112 -l AuthPriv -a md5 -A roomtoor -x aes -X roomtoor $ip $oid"
    else
        echo "Invalid SNMP version"
        return 1
    fi
    output=$(eval "$snmp_command")
    echo "$output"
}

# Function to display SNMP information
display_snmp_info() {
    version="$1"
    ip="$2"

    echo "SNMP version: $version"
    echo "---------------"
    echo "sysContact: $(execute_snmp_command $version $ip $sysContact)"
    echo "sysName: $(execute_snmp_command $version $ip $sysName)"
    echo "sysLocation: $(execute_snmp_command $version $ip $sysLocation)"
    echo "ifNumber: $(execute_snmp_command $version $ip $ifNumber)"
    echo "sysUptime: $(execute_snmp_command $version $ip $sysUptime)"
    echo ""
}

display_snmp_info "1" "198.51.100.1"
display_snmp_info "2c" "198.51.100.3"
display_snmp_info "3" "198.51.100.4"
```

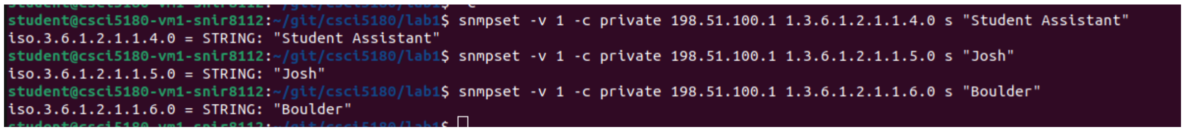
3. Use SNMPSET commands to modify Contact, Name, and Location to display varied output for each version: 1 and 2. Paste relevant screenshots. [**10 points**]

SNMP Set for v1:

First gave write permission to private community:



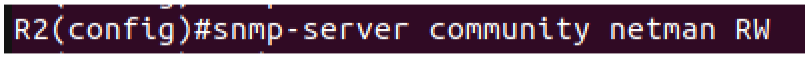
```
R1#show run | i community
snmp-server community private RW
```



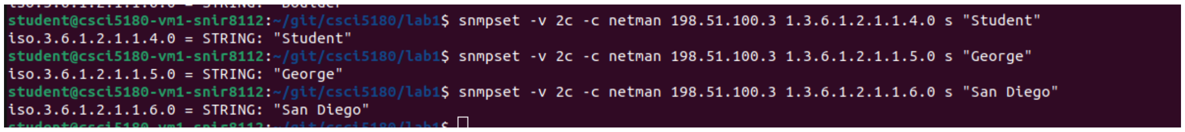
```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 1 -c private 198.51.100.1 1.3.6.1.2.1.1.4.0 s "Student Assistant"
iso.3.6.1.2.1.1.4.0 = STRING: "Student Assistant"
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 1 -c private 198.51.100.1 1.3.6.1.2.1.1.5.0 s "Josh"
iso.3.6.1.2.1.1.5.0 = STRING: "Josh"
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 1 -c private 198.51.100.1 1.3.6.1.2.1.1.6.0 s "Boulder"
iso.3.6.1.2.1.1.6.0 = STRING: "Boulder"
```

SNMPSET for v2:

First gave write permissions to netman community:

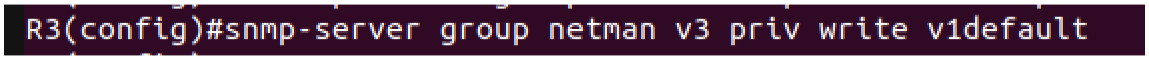


```
R2(config)#snmp-server community netman RW
```

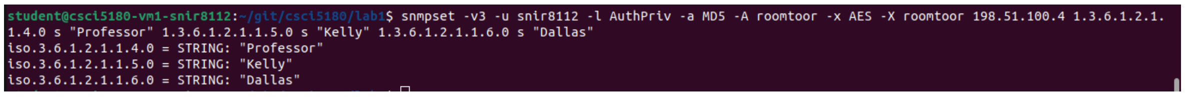


```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 2c -c netman 198.51.100.3 1.3.6.1.2.1.1.4.0 s "Student"
iso.3.6.1.2.1.1.4.0 = STRING: "Student"
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 2c -c netman 198.51.100.3 1.3.6.1.2.1.1.5.0 s "George"
iso.3.6.1.2.1.1.5.0 = STRING: "George"
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 2c -c netman 198.51.100.3 1.3.6.1.2.1.1.6.0 s "San Diego"
iso.3.6.1.2.1.1.6.0 = STRING: "San Diego"
```

SNMPSET for v3:
First gave write permissions to netman group to be able to run snmpset:



```
R3(config)#snmp-server group netman v3 priv write v1default
```



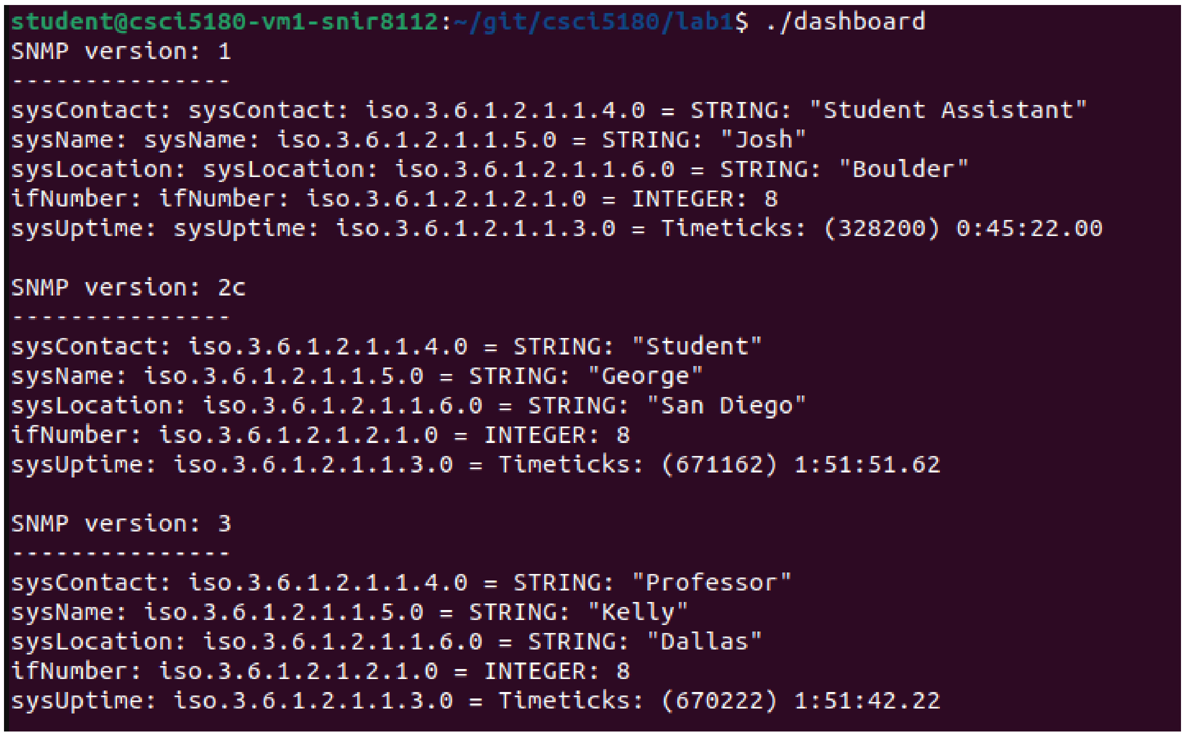
```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v3 -u snir8112 -l AuthPriv -a MD5 -A roomtoor -x AES -X roomtoor 198.51.100.4 1.3.6.1.2.1.
1.4.0 s "Professor" 1.3.6.1.2.1.1.5.0 s "Kelly" 1.3.6.1.2.1.1.6.0 s "Dallas"
iso.3.6.1.2.1.1.4.0 = STRING: "Professor"
iso.3.6.1.2.1.1.5.0 = STRING: "Kelly"
iso.3.6.1.2.1.1.6.0 = STRING: "Dallas"
```

Sample dashboard to be displayed using UNIX/Python:

SNMP v1
Contact: Student Assistant
Name: Josh
Location: Boulder
Number: 2
Uptime: 0:54:20.47

SNMP v2
Contact: Student
Name: George
Location: San Diego
Number: 2
Uptime: 0:67:10.57

SNMP v3 (any of the 2)
Contact: Professor
Name: Kelly
Location: Dallas
Number: 2
Uptime: 1:24:20.47

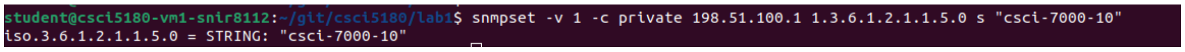


```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ ./dashboard
SNMP version: 1
---------------
sysContact: sysContact: iso.3.6.1.2.1.1.4.0 = STRING: "Student Assistant"
sysName: sysName: iso.3.6.1.2.1.1.5.0 = STRING: "Josh"
sysLocation: sysLocation: iso.3.6.1.2.1.1.6.0 = STRING: "Boulder"
ifNumber: ifNumber: iso.3.6.1.2.1.2.1.0 = INTEGER: 8
sysUptime: sysUptime: iso.3.6.1.2.1.1.3.0 = Timeticks: (328200) 0:45:22.00

SNMP version: 2c
---------------
sysContact: iso.3.6.1.2.1.1.4.0 = STRING: "Student"
sysName: iso.3.6.1.2.1.1.5.0 = STRING: "George"
sysLocation: iso.3.6.1.2.1.1.6.0 = STRING: "San Diego"
ifNumber: iso.3.6.1.2.1.2.1.0 = INTEGER: 8
sysUptime: iso.3.6.1.2.1.1.3.0 = Timeticks: (671162) 1:51:51.62

SNMP version: 3
---------------
sysContact: iso.3.6.1.2.1.1.4.0 = STRING: "Professor"
sysName: iso.3.6.1.2.1.1.5.0 = STRING: "Kelly"
sysLocation: iso.3.6.1.2.1.1.6.0 = STRING: "Dallas"
ifNumber: iso.3.6.1.2.1.2.1.0 = INTEGER: 8
sysUptime: iso.3.6.1.2.1.1.3.0 = Timeticks: (670222) 1:51:42.22
```

Objective 3: SNMPSET Commands

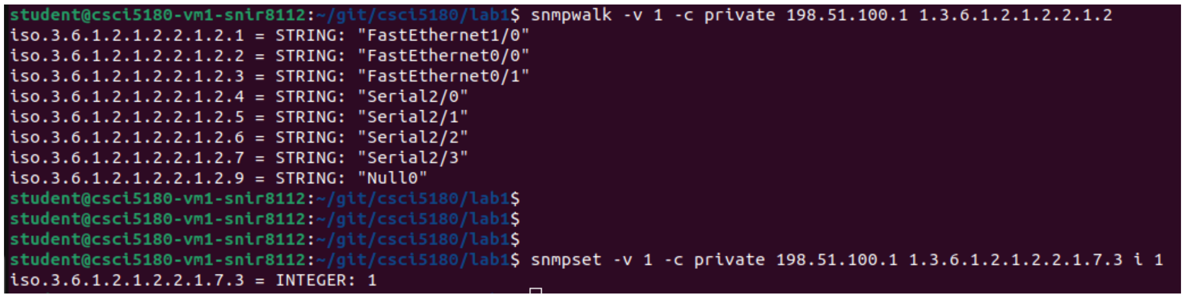
NOTE: Must use SNMPSET commands to perform the below tasks on Router 1 in GNS3:

1. Change the hostname to “**csci-7000-10**” (provide a screenshot) [**10 points**]



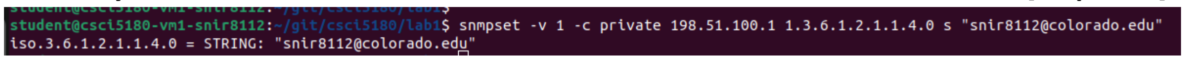
```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 1 -c private 198.51.100.1 1.3.6.1.2.1.1.5.0 s "csci-7000-10"
iso.3.6.1.2.1.1.5.0 = STRING: "csci-7000-10"
```

2. Change the interface status of the secondary interface (NOT THE MANAGEMENT INTERFACE) to “**Up**” (Assuming it’s up, if not, change to “**Admin Down**”). Provide screenshots. [**10 points**]



```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpwalk -v 1 -c private 198.51.100.1 1.3.6.1.2.1.2.2.1.2
iso.3.6.1.2.1.2.2.1.2.1 = STRING: "FastEthernet1/0"
iso.3.6.1.2.1.2.2.1.2.2 = STRING: "FastEthernet0/0"
iso.3.6.1.2.1.2.2.1.2.3 = STRING: "FastEthernet0/1"
iso.3.6.1.2.1.2.2.1.2.4 = STRING: "Serial2/0"
iso.3.6.1.2.1.2.2.1.2.5 = STRING: "Serial2/1"
iso.3.6.1.2.1.2.2.1.2.6 = STRING: "Serial2/2"
iso.3.6.1.2.1.2.2.1.2.7 = STRING: "Serial2/3"
iso.3.6.1.2.1.2.2.1.2.9 = STRING: "Null0"
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 1 -c private 198.51.100.1 1.3.6.1.2.1.2.2.1.7.3 i 1
iso.3.6.1.2.1.2.2.1.7.3 = INTEGER: 1
```

3. Create a SNMP contact profile with the name (provide a screenshot): <**yourname@colorado.edu**> [**10 points**]



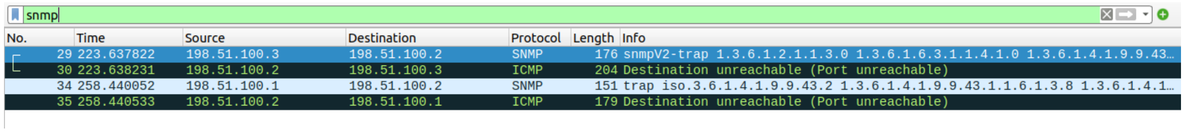
```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ snmpset -v 1 -c private 198.51.100.1 1.3.6.1.2.1.1.4.0 s "snir8112@colorado.edu"
iso.3.6.1.2.1.1.4.0 = STRING: "snir8112@colorado.edu"
```

Objective 4: SNMP Traps and Wireshark/TCPDUMP

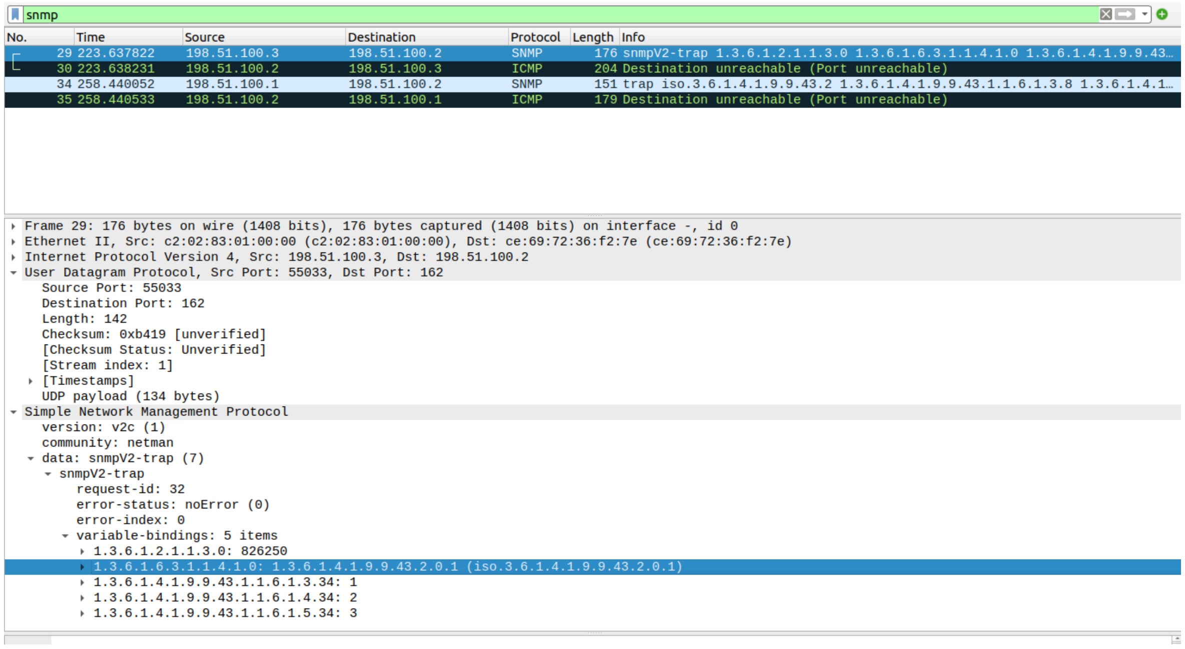
1. Start a new Wireshark capture on the tap0 interface of the VM. Apply a display filter to filter SNMP traffic.

2. Shutdown the interfaces on R2 and R3, and bring them up again. Do you observe different trap messages being exchanged between the SNMP agent and the manager (VM) in the packet capture? Provide relevant screenshots. [**10 points**]

For v2:

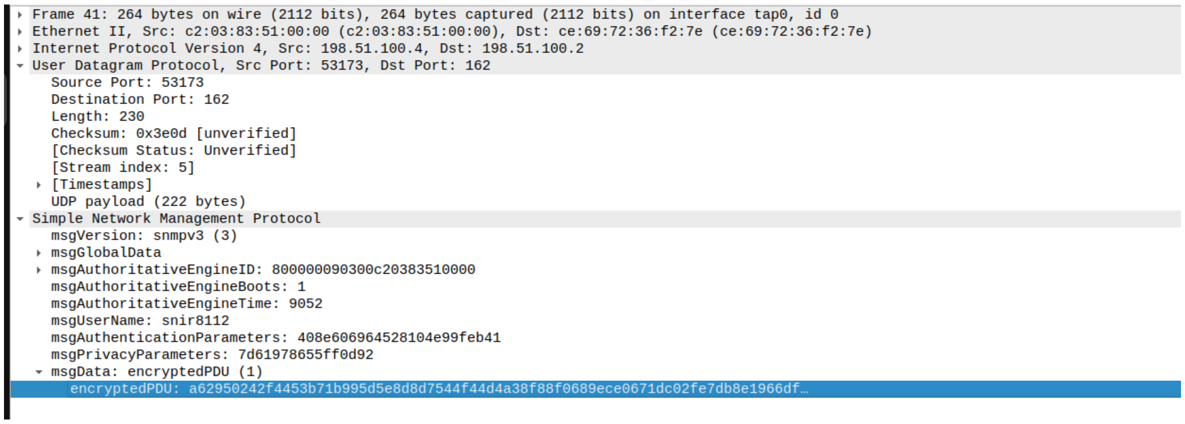


```
snmp
No.  Time        Source        Destination   Protocol Length Info
  29 223.637822  198.51.100.3  198.51.100.2  SNMP     176 snmpV2-trap 1.3.6.1.2.1.1.3.0 1.3.6.1.6.3.1.1.4.1.0 1.3.6.1.4.1.9.9.43…
  30 223.638231  198.51.100.2  198.51.100.3  ICMP     204 Destination unreachable (Port unreachable)
  34 258.440052  198.51.100.1  198.51.100.2  SNMP     151 trap iso.3.6.1.4.1.9.9.43.2 1.3.6.1.4.1.9.9.43.1.1.6.1.3.8 1.3.6.1.4.1…
  35 258.440533  198.51.100.2  198.51.100.1  ICMP     179 Destination unreachable (Port unreachable)
```



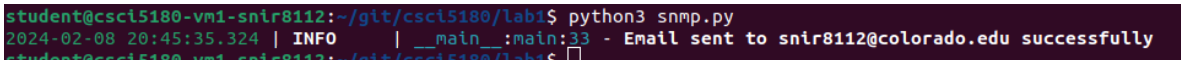
```
snmp
No.  Time        Source        Destination   Protocol Length Info
  29 223.637822  198.51.100.3  198.51.100.2  SNMP     176 snmpV2-trap 1.3.6.1.2.1.1.3.0 1.3.6.1.6.3.1.1.4.1.0 1.3.6.1.4.1.9.9.43…
  30 223.638231  198.51.100.2  198.51.100.3  ICMP     204 Destination unreachable (Port unreachable)
  34 258.440052  198.51.100.1  198.51.100.2  SNMP     151 trap iso.3.6.1.4.1.9.9.43.2 1.3.6.1.4.1.9.9.43.1.1.6.1.3.8 1.3.6.1.4.1…
  35 258.440533  198.51.100.2  198.51.100.1  ICMP     179 Destination unreachable (Port unreachable)

Frame 29: 176 bytes on wire (1408 bits), 176 bytes captured (1408 bits) on interface -, id 0
Ethernet II, Src: c2:02:83:01:00:00 (c2:02:83:01:00:00), Dst: ce:69:72:36:f2:7e (ce:69:72:36:f2:7e)
Internet Protocol Version 4, Src: 198.51.100.3, Dst: 198.51.100.2
User Datagram Protocol, Src Port: 55033, Dst Port: 162
  Source Port: 55033
  Destination Port: 162
  Length: 142
  Checksum: 0xb419 [unverified]
  [Checksum Status: Unverified]
  [Stream index: 1]
  [Timestamps]
  UDP payload (134 bytes)
Simple Network Management Protocol
  version: v2c (1)
  community: netman
  data: snmpV2-trap (7)
    snmpV2-trap
      request-id: 32
      error-status: noError (0)
      error-index: 0
      variable-bindings: 5 items
        1.3.6.1.2.1.1.3.0: 826250
        1.3.6.1.6.3.1.1.4.1.0: 1.3.6.1.4.1.9.9.43.2.0.1 (iso.3.6.1.4.1.9.9.43.2.0.1)
        1.3.6.1.4.1.9.9.43.1.1.6.1.3.34: 1
        1.3.6.1.4.1.9.9.43.1.1.6.1.4.34: 2
        1.3.6.1.4.1.9.9.43.1.1.6.1.5.34: 3
```

For v3:

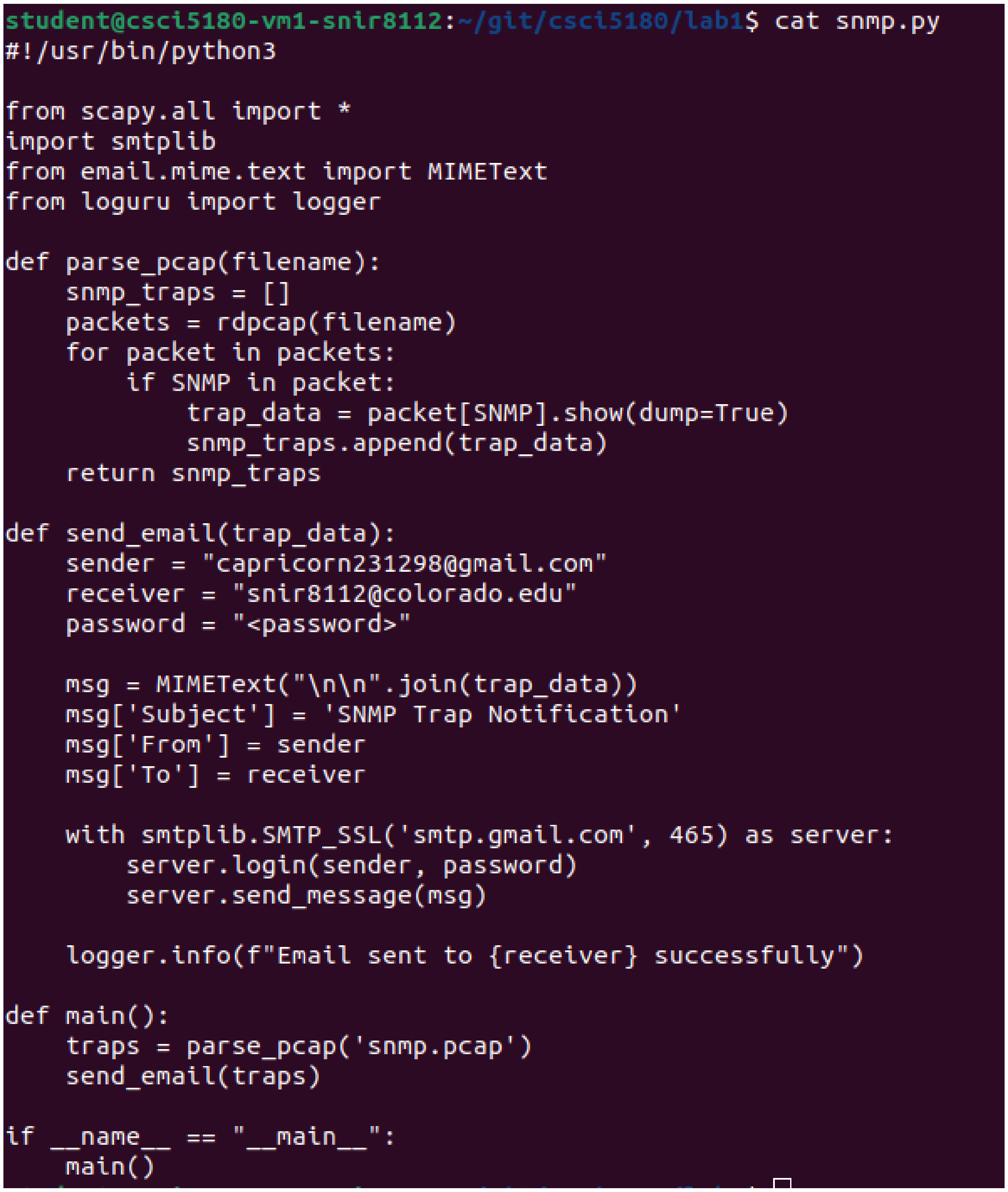


```
Frame 41: 264 bytes on wire (2112 bits), 264 bytes captured (2112 bits) on interface tap0, id 0
Ethernet II, Src: c2:03:83:51:00:00 (c2:03:83:51:00:00), Dst: ce:69:72:36:f2:7e (ce:69:72:36:f2:7e)
Internet Protocol Version 4, Src: 198.51.100.4, Dst: 198.51.100.2
User Datagram Protocol, Src Port: 53173, Dst Port: 162
  Source Port: 53173
  Destination Port: 162
  Length: 230
  Checksum: 0x3e0d [unverified]
  [Checksum Status: Unverified]
  [Stream index: 5]
  [Timestamps]
  UDP payload (222 bytes)
Simple Network Management Protocol
  msgVersion: snmpv3 (3)
  msgGlobalData
  msgAuthoritativeEngineID: 800000090300c20383510000
  msgAuthoritativeEngineBoots: 1
  msgAuthoritativeEngineTime: 9052
  msgUserName: snir8112
  msgAuthenticationParameters: 408e606964528104e99feb41
  msgPrivacyParameters: 7d61978655ff0d92
  msgData: encryptedPDU (1)
    encryptedPDU: a62950242f4453b71b995d5e8d8d7544f44d4a38f88f0689ece0671dc02fe7db8e1966df…
```

3. Start a capture using TCPDUMP. Bring down an interface on any of the routers (this should generate a trap). Store the output in a .pcap file. After stopping the TCPDUMP, create a Python script that will analyze and parse the .pcap file for a Trap. Then the Python script should generate an email, to your email id, with the contents of the Trap [https://www.pythonforbeginners.com/google/sending-emails-using-google]. Provide relevant screenshots and submit the code. [**20 points**]



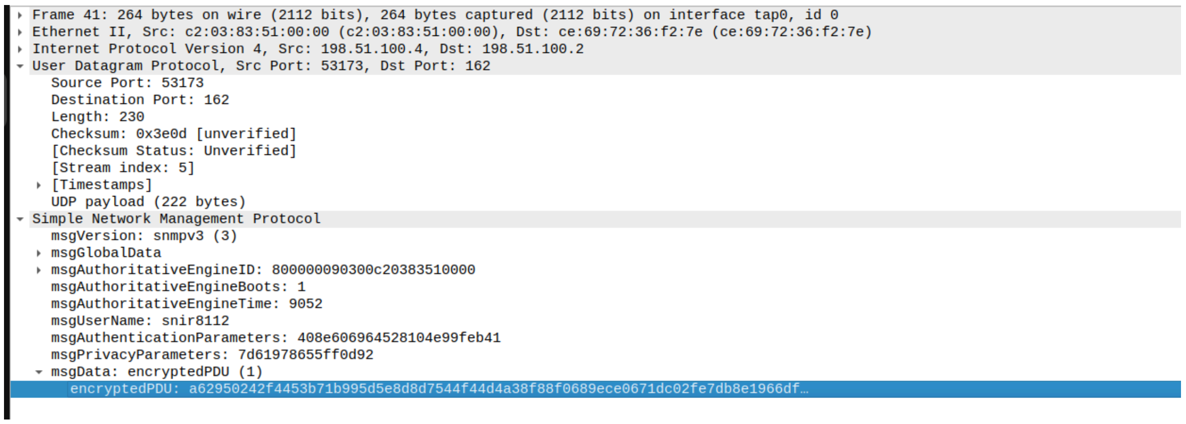
```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ python3 snmp.py
2024-02-08 20:45:35.324 | INFO     | __main__:main:33 - Email sent to snir8112@colorado.edu successfully
```



```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ cat snmp.py
#!/usr/bin/python3

from scapy.all import *
import smtplib
from email.mime.text import MIMEText
from loguru import logger

def parse_pcap(filename):
    snmp_traps = []
    packets = rdpcap(filename)
    for packet in packets:
        if SNMP in packet:
            trap_data = packet[SNMP].show(dump=True)
            snmp_traps.append(trap_data)
    return snmp_traps

def send_email(trap_data):
    sender = "capricorn231298@gmail.com"
    receiver = "snir8112@colorado.edu"
    password = "<password>"

    msg = MIMEText("\n\n".join(trap_data))
    msg['Subject'] = 'SNMP Trap Notification'
    msg['From'] = sender
    msg['To'] = receiver

    with smtplib.SMTP_SSL('smtp.gmail.com', 465) as server:
        server.login(sender, password)
        server.send_message(msg)

    logger.info(f"Email sent to {receiver} successfully")

def main():
    traps = parse_pcap('snmp.pcap')
    send_email(traps)

if __name__ == "__main__":
    main()
```

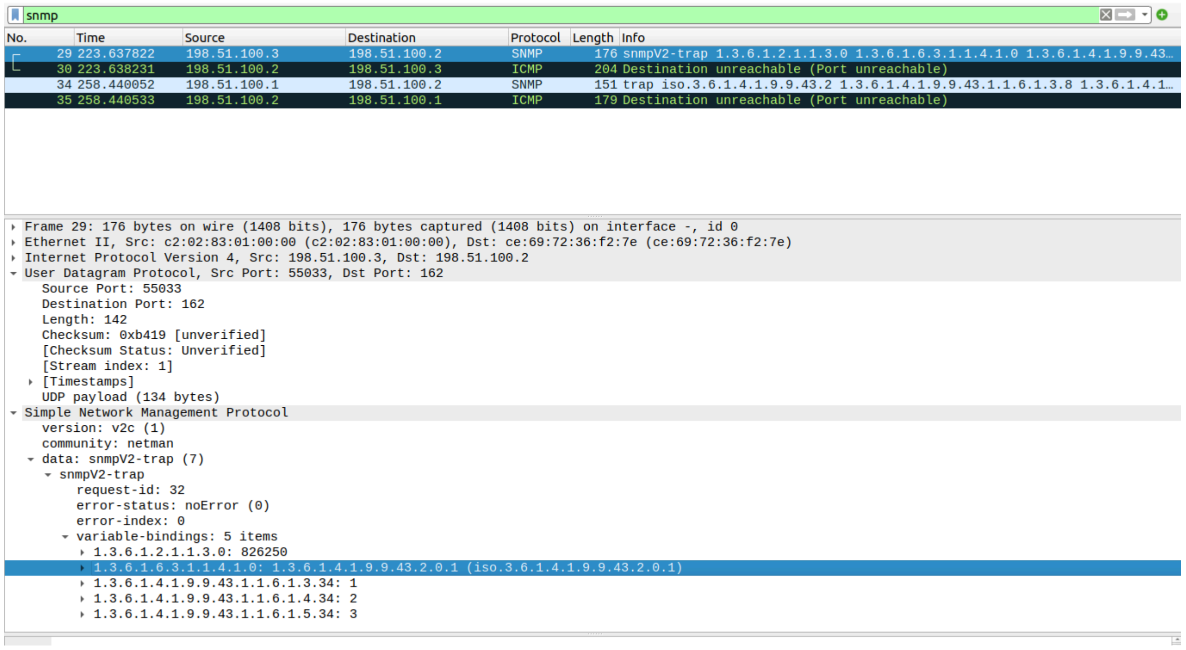
4. What are the key differences you can observe between the trap messages for SNMPv2 and v3? Provide relevant screenshots highlighting the differences. [**10 Points**]

As highlighted, the data is encrypted in SNMP v3:



```
▸ Frame 41: 264 bytes on wire (2112 bits), 264 bytes captured (2112 bits) on interface tap0, id 0
▸ Ethernet II, Src: c2:03:83:51:00:00 (c2:03:83:51:00:00), Dst: ce:69:72:36:f2:7e (ce:69:72:36:f2:7e)
▸ Internet Protocol Version 4, Src: 198.51.100.4, Dst: 198.51.100.2
▾ User Datagram Protocol, Src Port: 53173, Dst Port: 162
    Source Port: 53173
    Destination Port: 162
    Length: 230
    Checksum: 0x3e0d [unverified]
    [Checksum Status: Unverified]
    [Stream index: 5]
  ▸ [Timestamps]
    UDP payload (222 bytes)
▾ Simple Network Management Protocol
    msgVersion: snmpv3 (3)
  ▸ msgGlobalData
  ▸ msgAuthoritativeEngineID: 800000090300c20383510000
    msgAuthoritativeEngineBoots: 1
    msgAuthoritativeEngineTime: 9052
    msgUserName: snir8112
    msgAuthenticationParameters: 408e606964528104e99feb41
    msgPrivacyParameters: 7d61978655ff0d92
  ▾ msgData: encryptedPDU (1)
      encryptedPDU: a62950242f4453b71b995d5e8d8d7544f44d4a38f88f0689ece0671dc02fe7db8e1966df…
```

Whereas it is in clear text in v2:



```
snmp

No.   Time        Source         Destination    Protocol  Length Info
   29 223.637822  198.51.100.3   198.51.100.2   SNMP         176 snmpV2-trap 1.3.6.1.2.1.1.3.0 1.3.6.1.6.3.1.1.4.1.0 1.3.6.1.4.1.9.9.43…
   30 223.638231  198.51.100.2   198.51.100.3   ICMP         204 Destination unreachable (Port unreachable)
   34 258.440052  198.51.100.1   198.51.100.2   SNMP         151 trap iso.3.6.1.4.1.9.9.43.2 1.3.6.1.4.1.9.9.43.1.1.6.1.3.8 1.3.6.1.4.1…
   35 258.440533  198.51.100.2   198.51.100.1   ICMP         179 Destination unreachable (Port unreachable)

▸ Frame 29: 176 bytes on wire (1408 bits), 176 bytes captured (1408 bits) on interface -, id 0
▸ Ethernet II, Src: c2:02:83:01:00:00 (c2:02:83:01:00:00), Dst: ce:69:72:36:f2:7e (ce:69:72:36:f2:7e)
▸ Internet Protocol Version 4, Src: 198.51.100.3, Dst: 198.51.100.2
▾ User Datagram Protocol, Src Port: 55033, Dst Port: 162
    Source Port: 55033
    Destination Port: 162
    Length: 142
    Checksum: 0xb419 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 1]
  ▸ [Timestamps]
    UDP payload (134 bytes)
▾ Simple Network Management Protocol
    version: v2c (1)
    community: netman
  ▾ data: snmpV2-trap (7)
    ▾ snmpV2-trap
        request-id: 32
        error-status: noError (0)
        error-index: 0
      ▾ variable-bindings: 5 items
        ▸ 1.3.6.1.2.1.1.3.0: 826250
        ▸ 1.3.6.1.6.3.1.1.4.1.0: 1.3.6.1.4.1.9.9.43.2.0.1 (iso.3.6.1.4.1.9.9.43.2.0.1)
        ▸ 1.3.6.1.4.1.9.9.43.1.1.6.1.3.34: 1
        ▸ 1.3.6.1.4.1.9.9.43.1.1.6.1.4.34: 2
        ▸ 1.3.6.1.4.1.9.9.43.1.1.6.1.5.34: 3
```

Objective 5: Network Administration using SNMP [Extra Credit]

Imagine a Data Center or Service Provider network. You, being a principle network engineer, get a ticket for eBGP sessions going down on multiple routers. You start analyzing the output of all the possible “show” commands in BGP that you are aware of. However, all configurations and parameters look perfect and you scratch your head for a while trying to know the root cause of the issue. You run down to the data center/lab and check all the physical connections. On doing a “show ip interface brief” on all the affected routers, you see that some of the interfaces have been taken down administratively and the others show a Protocol down. Most networking problems reside at the lower levels and hence troubleshooting layer 1 is the first step of a bottom-up approach. The following objective will help you find an easier and faster way to check the layer 1 status before moving up the OSI model for troubleshooting. (**12 points**)

1. Configure descriptions for the router interfaces for easier administration (e.g. Router(config-if)# description Management Interface).

2. Write a script in a language of your choice (e.g. UNIX/Python) to extract and display interface information from all the routers in the above topology using the following MIB objects (Hint: you can view entire MIB details using SNMPBULKWALK command).

- ifName
- ifDescr
- ifOperStatus
- iPhysAddress
- ifAdminStatus
- ifInUcastPkts

Sample output to be displayed by the script:

	Interface Name	Description	Operational Status	Physical Address	Admin Status	Incoming Unicast Packet Counter
R1	Fa0/0	Management Interface	Up	00-03-47-92-9C-6F	Up	100

Provide relevant screenshots.

3. Modify the above script to retrieve and display: interface IP address and network mask information. Provide relevant screenshots.

4. Implement both the scripts (TCPDUMP Trap obj 4.3 and extract interface info obj 5.2) using just one script. Also, ensure your script shall continuously monitor the interface status, display the interface information (as in obj 5.2) and parse the trap (as in obj 4.3). Provide relevant screenshots.

Report Questions (5 points each)

1. Would you recommend using a management subnet for SNMP? Why/why not?

Placing SNMP management traffic on a dedicated subnet separates it from other network traffic, reducing the risk of unauthorized access or interference. Further, you could de-prioritize SNMP traffic and not choke the network link.

2. Why is a switch used in the network design in GNS3?

In this network design, a switch is used to establish connections between the Network Management System (NMS) or my Virtual machine and the SNMP agents (3 routers).

3. Can you use a router instead? Why/why not?

Yes, we can use a router instead of an L2 switch here. However, a switch provides faster processing since ithas lesser overhead and multiple ports. It can connect multiple end devices and routers usually do not have multiple ports.

4. If you used a router, what would need to change (if anything).

Since a router separates a broadcast domain, we would need to assign iP addresses from different subnets to each of the interfaces to the SNMP agents (other routers). We don't need to have any routing protocols in place for this design since the 3 SNMP agents connect to 1 router, and that 1 router would have "connected" routes to all the other routers. However, the SNMP agents, would need to know the route back to the VM so we would need to configure routes on these SNMP agents accordingly. Further, the VM also needs to have a default gateway configured to talk to the SNMP agents.

5. What command has to be entered on the router, to disable configuration changes to be made through SNMP?

For v1 and v2: snmp-server community <string> ro
For v3: snmp-server group <group name> v3 priv read v1default
This would gives read-only access to the user. If we want to disable that we could just prepend with "no" followed by the above commands to disable complete access.

Network Discovery using NMAP

Objectives

- Learn the basic operations of network discovery using Nmap.
- Learn how to capture and analyze ICMP traffic.
- Learn how to capture and analyze port scanning traffic.
- Perform IP address spoofing.
- Gather OS information.
- Perform Scripting and Automation.

Summary

Nmap is a free open source tool that can be used for performing a variety of network scanning and security functions. To create a “map” of the network, Nmap sends specific packets to the target host (or hosts) and then analyzes the responses. Nmap can also be used to enumerate networks and avoid IDS through spoofing/stealth, please use this responsibly and follow the lab directions.

Nmap is available for download for many Linux distributions (There is also a version available for Windows). It also comes with a GUI (Zmap) that can be used as an alternative to the CLI. The functions of this lab will focus on ping sweeps (find hosts), port scanning (determine vulnerabilities/services), IP spoofing (avoiding detection by IDS), and gathering intelligence on a network.

Objective 1: Download and Install Nmap/Zmap on Your Machine

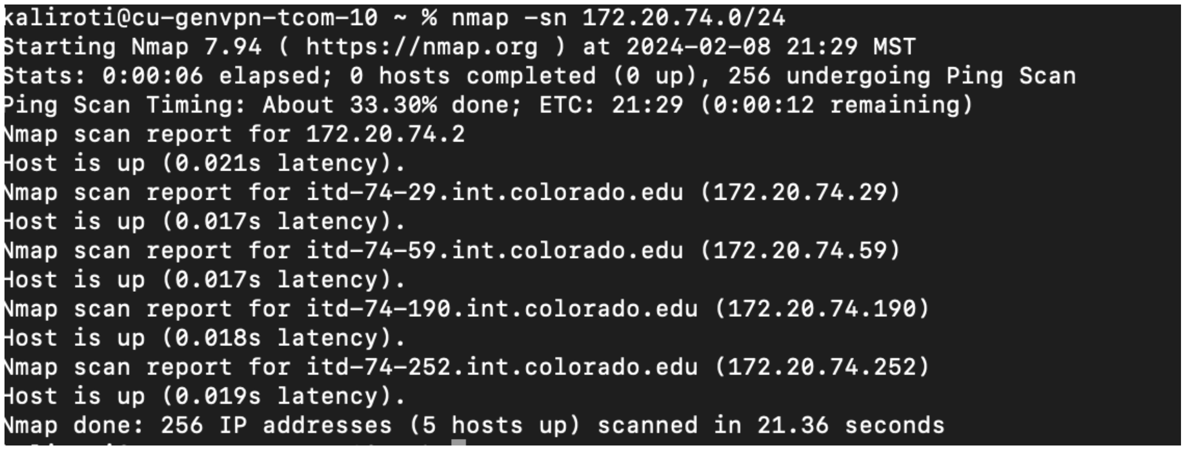
Follow the instructions from the Nmap website for your operating system:

https://nmap.org/

For the remainder of this lab, you can use **Nmap or Zenmap**

Objective 2: Ping Sweeps and Port Scans

1. Perform a ping sweep for the following network (Note: this only works from CU network or VPN; if unavailable use your home/private network): **172.20.74.0/24**
   a. Provide a screenshot showing the command and the results [**5 points**]



```
kaliroti@cu-genvpn-tcom-10 ~ % nmap -sn 172.20.74.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2024-02-08 21:29 MST
Stats: 0:00:06 elapsed; 0 hosts completed (0 up), 256 undergoing Ping Scan
Ping Scan Timing: About 33.30% done; ETC: 21:29 (0:00:12 remaining)
Nmap scan report for 172.20.74.2
Host is up (0.021s latency).
Nmap scan report for itd-74-29.int.colorado.edu (172.20.74.29)
Host is up (0.017s latency).
Nmap scan report for itd-74-59.int.colorado.edu (172.20.74.59)
Host is up (0.017s latency).
Nmap scan report for itd-74-190.int.colorado.edu (172.20.74.190)
Host is up (0.018s latency).
Nmap scan report for itd-74-252.int.colorado.edu (172.20.74.252)
Host is up (0.019s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 21.36 seconds
```

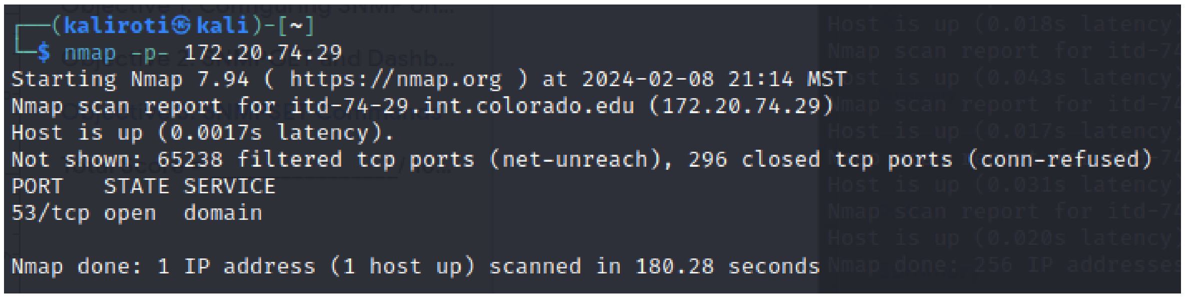
b. How many devices responded to the ping sweep? Provide information about how you can determine this. [**2.5 points**]

5 hosts responded to the ping sweep. As seen in the above screenshot, the last line “Nmap done: 256 IP addresses **(5 hosts up)**...” indicates that there are 5 hosts up in that network.

2. Choose a host that replied from the ping sweep; now perform a full scan on that host

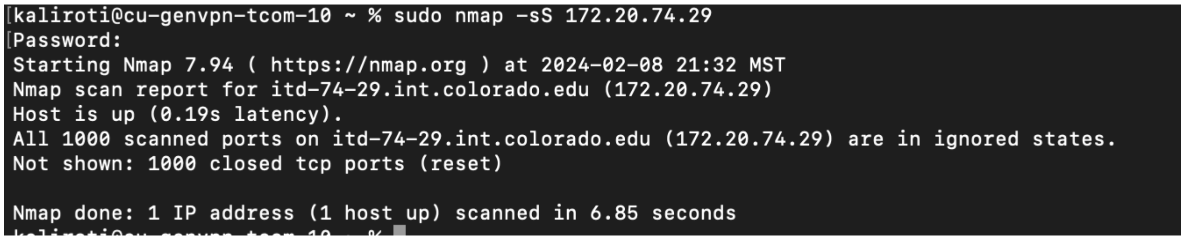
   a. Which well-known ports were open on this machine? Provide the screenshot. [**2.5 points**]

Port 53 was open on this machine



```
┌──(kaliroti㉿kali)-[~]
└─$ nmap -p- 172.20.74.29
Starting Nmap 7.94 ( https://nmap.org ) at 2024-02-08 21:14 MST
Nmap scan report for itd-74-29.int.colorado.edu (172.20.74.29)
Host is up (0.0017s latency).
Not shown: 65238 filtered tcp ports (net-unreach), 296 closed tcp ports (conn-refused)
PORT   STATE SERVICE
53/tcp open  domain

Nmap done: 1 IP address (1 host up) scanned in 180.28 seconds
```

   b. Provide the command you would use to perform a “stealth” scan. [**2.5 points**]

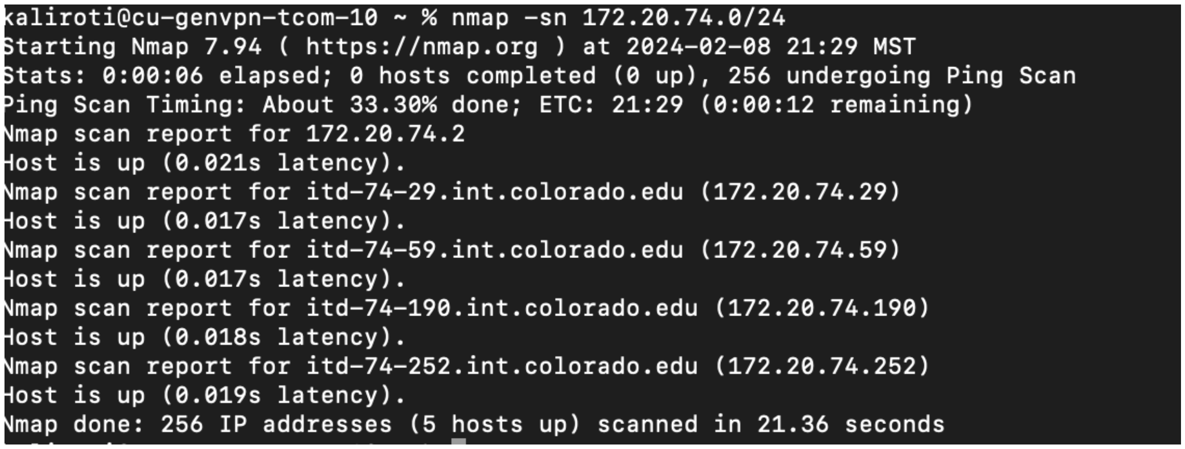


```
kaliroti@cu-genvpn-tcom-10 ~ % sudo nmap -sS 172.20.74.29
Password:
Starting Nmap 7.94 ( https://nmap.org ) at 2024-02-08 21:32 MST
Nmap scan report for itd-74-29.int.colorado.edu (172.20.74.29)
Host is up (0.19s latency).
All 1000 scanned ports on itd-74-29.int.colorado.edu (172.20.74.29) are in ignored states.
Not shown: 1000 closed tcp ports (reset)

Nmap done: 1 IP address (1 host up) scanned in 6.85 seconds
```

Objective 3: IP Spoofing and OS Detection

1. Perform a full network scan on the /24 network (optional: use a spoofed IP address (use target IP address from previous objective as the source))

   a. Provide the command used [**2.5 points**]



```
kaliroti@cu-genvpn-tcom-10 ~ % nmap -sn 172.20.74.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2024-02-08 21:29 MST
Stats: 0:00:06 elapsed; 0 hosts completed (0 up), 256 undergoing Ping Scan
Ping Scan Timing: About 33.30% done; ETC: 21:29 (0:00:12 remaining)
Nmap scan report for 172.20.74.2
Host is up (0.021s latency).
Nmap scan report for itd-74-29.int.colorado.edu (172.20.74.29)
Host is up (0.017s latency).
Nmap scan report for itd-74-59.int.colorado.edu (172.20.74.59)
Host is up (0.017s latency).
Nmap scan report for itd-74-190.int.colorado.edu (172.20.74.190)
Host is up (0.018s latency).
Nmap scan report for itd-74-252.int.colorado.edu (172.20.74.252)
Host is up (0.019s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 21.36 seconds
```

b. Explain the different "state" options for a Nmap port scan (i.e. open, filtered, closed, etc.) [**2.5 points**]

Open: This indicates that the scanned port actively responds to Nmap probes, indicating that a service is running on that port and is accessible.

Closed: A "closed" port means that no application is listening on that port.

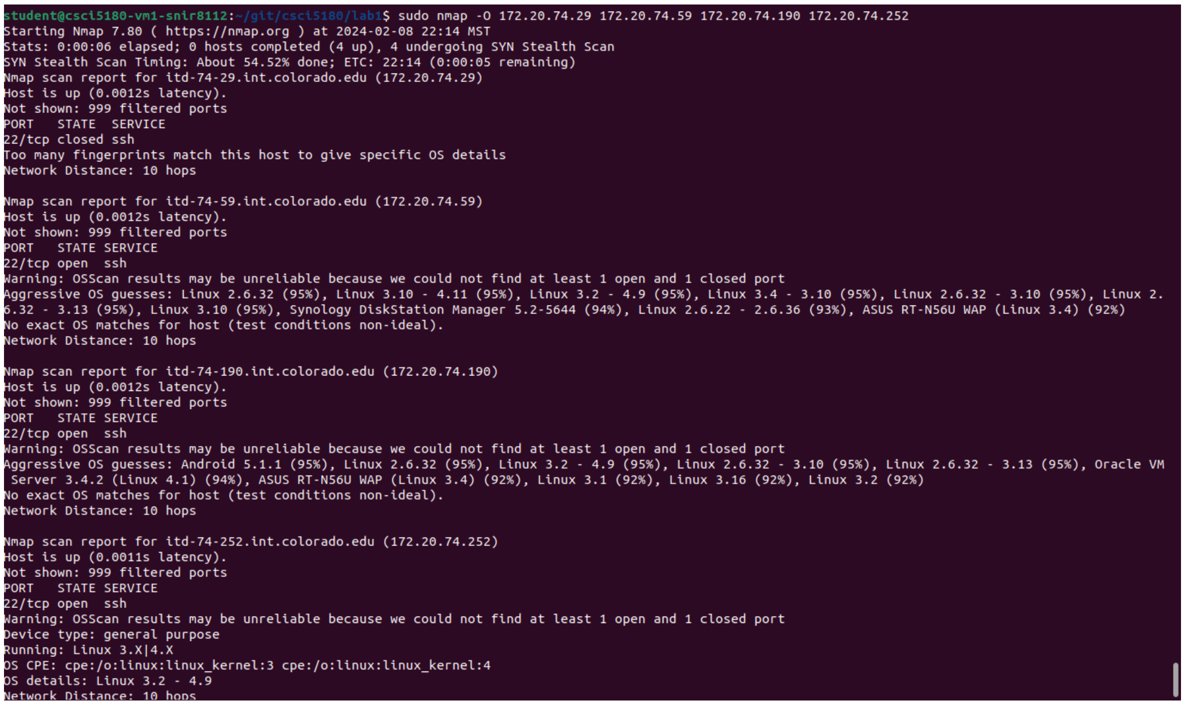
Filtered: When Nmap identifies a port as "filtered," it means that it was unable to determine whether the port is open or closed due to filtering mechanisms like firewalls, packet filtering, or other network devices.

Unfiltered: This state indicates that Nmap was able to determine that the port is neither open nor closed with certainty.

Open|Filtered: This state implies that Nmap is unable to determine whether the port is open or filtered.

Closed|Filtered: This state suggests that Nmap is unable to determine whether the port is closed or filtered.

2. Provide screenshots of the Operating Systems running on each of these machines [**2.5 points**]



```
student@csci5180-vm1-snir8112:~/git/csci5180/lab1$ sudo nmap -O 172.20.74.29 172.20.74.59 172.20.74.190 172.20.74.252
Starting Nmap 7.80 ( https://nmap.org ) at 2024-02-08 22:14 MST
Stats: 0:00:06 elapsed; 0 hosts completed (4 up), 4 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 54.52% done; ETC: 22:14 (0:00:05 remaining)
Nmap scan report for itd-74-29.int.colorado.edu (172.20.74.29)
Host is up (0.0012s latency).
Not shown: 999 filtered ports
PORT   STATE  SERVICE
22/tcp closed ssh
Too many fingerprints match this host to give specific OS details
Network Distance: 10 hops

Nmap scan report for itd-74-59.int.colorado.edu (172.20.74.59)
Host is up (0.0012s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 2.6.32 (95%), Linux 3.10 - 4.11 (95%), Linux 3.2 - 4.9 (95%), Linux 3.4 - 3.10 (95%), Linux 2.6.32 - 3.10 (95%), Linux 2.
6.32 - 3.13 (95%), Linux 3.10 (95%), Synology DiskStation Manager 5.2-5644 (94%), Linux 2.6.22 - 2.6.36 (93%), ASUS RT-N56U WAP (Linux 3.4) (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 10 hops

Nmap scan report for itd-74-190.int.colorado.edu (172.20.74.190)
Host is up (0.0012s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Android 5.1.1 (95%), Linux 2.6.32 (95%), Linux 3.2 - 4.9 (95%), Linux 2.6.32 - 3.10 (95%), Linux 2.6.32 - 3.13 (95%), Oracle VM
 Server 3.4.2 (Linux 4.1) (94%), ASUS RT-N56U WAP (Linux 3.4) (92%), Linux 3.1 (92%), Linux 3.16 (92%), Linux 3.2 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 10 hops

Nmap scan report for itd-74-252.int.colorado.edu (172.20.74.252)
Host is up (0.0011s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 10 hops
```

Objective 4: Scripting and Automation

1. IP Address Mapping

   a. If using the VM, install Nmap

      #sudo apt-get install nmap

   b. Run a ping sweep on the /24 network

   c. Using **Bash or Python**, record the IP addresses into a **text/CSV**

   file d. Repeat the ping sweep after some time (~10 min.)

   e. Compare the two files

      i. Were there any differences? If so, what is different? [**2 points**]

      There was no difference except the latency between devices.

ii. Submit the scripts, files, procedures or screenshots of how you accomplished this [**10 points**]

[Attached the script]

f. As a network manager, list one thing that is useful and one thing could be detrimental with this information [**5 points**]

With the IP address mapping, you can gain insights into the active devices on your network. This information can be valuable for network troubleshooting, inventory management, and security monitoring. By comparing the two files generated before and after the ping sweep, you can identify changes in the network topology and detect new or removed devices. However, constantly running ping sweeps and maintaining a record of IP addresses can potentially generate a large amount of data. Depending on the size of the network and the frequency of the sweeps, this could lead to storage issues and increased network traffic.

2. **Extra Credit:**
   Rogue Web Server (web servers ending with IP addresses .1-.10 are legitimate; outside of that range are rogue)

   a. Run a full network port scan to find open ports for **80, 443**, and **8080**

   b. Submit the file of all web servers that are not in the range (i.e. rogue web server)

      i. How did you accomplish this? [**5 points**]

Report Questions

1. How can you set a decoy, to hide your source IP address using Nmap? [**2.5 points**]

nmap -D <decoy1,decoy2,...,decoyN> <target>

This command would initiate a scan on the 192.168.1.0/24 network with decoy IP addresses 10.0.0.1 and 172.16.0.1 included in the scan.

2. List some ways Nmap can be used to trick a firewall. [**2.5 points**]

- Nmap can send fragmented packets to a target system. Fragmented packets may evade simple firewall rules that inspect only the header of packets. By fragmenting packets, Nmap can potentially bypass firewall rules designed to block certain types of traffic.
- Idle scanning is a stealthy scanning technique that uses a third-party system (a "zombie") to indirectly scan a target. This technique may bypass firewall logs or evade detection as the scan traffic appears to originate from the zombie system rather than the attacker's system.
- Nmap allows the use of decoy IP addresses to obfuscate the true source of the scan. By specifying multiple decoy IP addresses, Nmap can make it more challenging for the firewall to distinguish between legitimate and malicious traffic, potentially leading to a false sense of security.

Total Score = __________________/ 167 [+17 Bonus]