ARQUITETURAS DE COMUNICAÇÃO

SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

Objectives

- Simple Network Management Protocol (SNMP) – Configuration, usage and security

SNMP v2c

1. Create a VM (VirtualBox) with Linux as operating system. Connect it to the internet, configuration the interface of the VM as NAT. Within the Linux VM guarantee that the Network Manager is configured to aquired IPv4 address automatically. Run the VM, and:

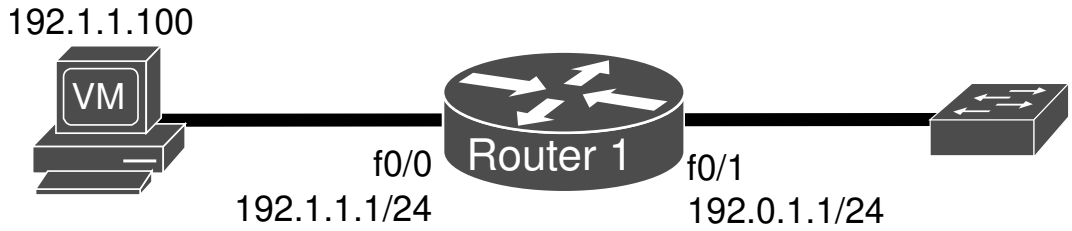
- Install the SNMP tools (package `snmp` or `net-snmp` in most distributions);

- Download, from elearning.ua.pt, the files mibs.zip and baseSNMP.py;

- Install the MIB Browser application. You may (i) install the Linux package `imibrowser` if available in our distribution, or (ii) download the JAVA version (mibbrowser.zip) from the website https://ireasoning.com/downloadmibbrowserlicense.shtml and unzip it. It requires the JAVA Runtime environment, install the respective package in your distribution (default-jre or jre8-openjdk).

Stop the VM and create a GNS3 template with it. Configure the VM network interface as **not attached** before starting ot in GNS3.

In GNS3, configure a network according to the following figure. It is recommended that Router1 should be a 7200 model with and IOS version superior to 15.0.



2. At the router, configure a SNMP community (using the default name "public") with Read-Only permissions:

```
Router(config)# snmp-server community public RO
```

Start a capture with Wireshark. Using the Linux SNMP tools (type "*man snmpwalk*" for more details), retrieve the Router's complete MIB information (starting from .1 base object ID):

```
snmpwalk -v2c -c public 192.1.1.1 .1
```

Analyze the information present in Router's MIB and the captured SNMP packets.

Note: if your VM does not have the MIBS placed in the default MIBS path, you should use the option "-m ALL -M ./mibs/ietf:./mibs/iana:./mibs/cisco" with commands snmpwalk, `snmpget` and `snmpset`. Where "./mibs" is the path of the unzipped contents of the MIBS.zip file.

3. Retrieve partial MIB information using a filter string:

```
snmpwalk -v2c -c public 192.1.1.1 <filter>
```

Retrieve the Router's general information:

```
snmpwalk -v2c -c public 192.1.1.1 sysDescr
snmpwalk -v2c -c public 192.1.1.1 .1.3.6.1.2.1.1.1
```

Retrieve the Router's ARP table:

```
snmpwalk -v2c -c public 192.1.1.1 at                                #IOS < 15.0
snmpwalk -v2c -c public 192.1.1.1 ipNetToMediaPhysAddress           #IOS >= 15.0
snmpwalk -v2c -c public 192.1.1.1 .1.3.6.1.2.1.3                    #IOS < 15.0
```

Retrieve the Router's IP routing table:

```
snmpwalk -v2c -c public 192.1.1.1 ipRoute            #IOS < 15.0
snmpwalk -v2c -c public 192.1.1.1 ipCidr             #IOS >= 15.0
snmpwalk -v2c -c public 192.1.1.1 .1.3.6.1.2.1.4.21  #IOS < 15.0
```

Retrieve the Router's interfaces information and identify the interfaces' names and status:

```
snmpwalk -v2c -c public 192.1.1.1 interfaces
snmpwalk -v2c -c public 192.1.1.1 .1.3.6.1.2.1.2
```

Try other filter strings.

4. Start a capture with Wireshark and try to obtain a specific MIB entry. For system description:

```
snmpget -v2c -c public 192.1.1.1 SNMPv2-MIB::sysDescr.0
snmpget -v2c -c public 192.1.1.1 .1.3.6.1.2.1.1.1.0
```

Try to obtain other MIB objects. Analyze the captured packets.

5. Start a capture with Wireshark and try to change the status of the Ethernet interface connected to 192.0.1.0/24 using the *snmpset* command (type "*man snmpset*" for more details):

```
snmpset -v2c -c public 192.1.1.1 IF-MIB::ifAdminStatus.2 i 2
```

Create a new community with Read-Write permission

```
Router(config)# snmp-server community myrouter1 RW
```

Retry the above *snmpset* command with the new community, verify at the router the correct change of the interface status and analyze the captured SNMP packets.

6. Remove the public community defined above and give the RO community another name:

```
Router(config)# no snmp-server community public RO
Router(config)# snmp-server community myrouter0 RO
```

Test the new community:

```
snmpwalk -v2c -c myrouter0 192.1.1.1 sysDescr
```

7. Restrict the access to myrouter0 community to PC VM (IP address 192.1.1.100):

```
Router(config)# access-list 10 permit 192.1.1.100
Router(config)# snmp-server community myrouter0 RO 10
```

Test the configuration by accessing the MIB from your PC:

```
snmpwalk -v2c -c myrouter0 192.1.1.1 sysDescr
```

8. Define new community with a MIB view restriction just to allow the RO access to the system objects:

```
Router(config)# snmp-server view myview system included
Router(config)# snmp-server community myrouter2 view myview RO 10
```

Test the configuration by trying to access the Router's interfaces information:

```
snmpwalk -v2c -c myrouter2 192.1.1.1 interfaces
```

Redefine the SNMP view to allow the access to the MIB's interfaces objects.

9. Start the MIB Browser. Run the command `imibrowser` (if install from repository) or run

```
$bash /ireasoning/mibbrowser/browser.sh
```

Add the provided MIB files, go to Go to Tools→ Options → MIB Files, and add the directory with the unzipped contents of the MIBS.zip file.

Create an agent to interact with the router's two SNMP communities (myrouter0 and myrouter1). Go to Tools→ Options → Agents, and add an agent.

10. In the MIB Browser, replicate the actions done in points 3, 4 and 5, by performing Walk, Get and Set operations.

SNMP traps

11. Routers can generate automatic SNMP messages to notify a specif event (SNMP traps). Perform the following configurations to generate a SNMP trap every time the Router's system log has a new entry:

```
Router(config)# snmp-server enable traps syslog
Router(config)# snmp-server host 192.1.1.100 version 2c myrouter
```

Start a capture with Wireshark. At the router, change (several times) the status of the Ethernet interface connected to 192.0.1.0/24. Analyze the captured packets.

(optional) SNMP Scripting (with Python)

12. Download and test the baseSNMP.py script, and understand how different MIB objects can be accessed.

```
python baseSNMP.py –r 192.1.1.1
```

Note that relevant MIBS should be place on the script's sub folder "./mibs/".

Use the following MIB objects to access relevant interface traffic statistics:

- *ifHCOutUcastPkts*, *ifHCInUcastPkts*, *ifHCOutOctets*, *ifHCInOctets* from IF-MIB.

13. Create a time loop, with periodicity given by argument, and retrieve interface statistics. Display, and store, byte and packet increments (in both directions) with timestamp.