NTNU
Kunnskap for en bedre verden

Department of Information Security and Communication Technology

TTM4240 - Advanced Network Control and Management

Fall 2021

Assignment 2

Group 15

Authors:
Sivert Lundli
Erlend Håkegård
Jakob Lund Johannessen

October, 2021

Contents

List of Figures

List of Tables

1 Introduction

This report is a mandatory delivery in the course TTM4240 - Advanced Network Control and Management at NTNU. The networking lab provides a hands on learning approach to networking for the students, through GNS3, a Network Software Emulator.

The topic for the lab was to monitor the network traffic in a small network. This was monitored using both *SNMP* and *NetFlow*, allowing us to see how they differ, their strengths, and weaknesses.

The network topology for the lab is shown in Figure 1 below.

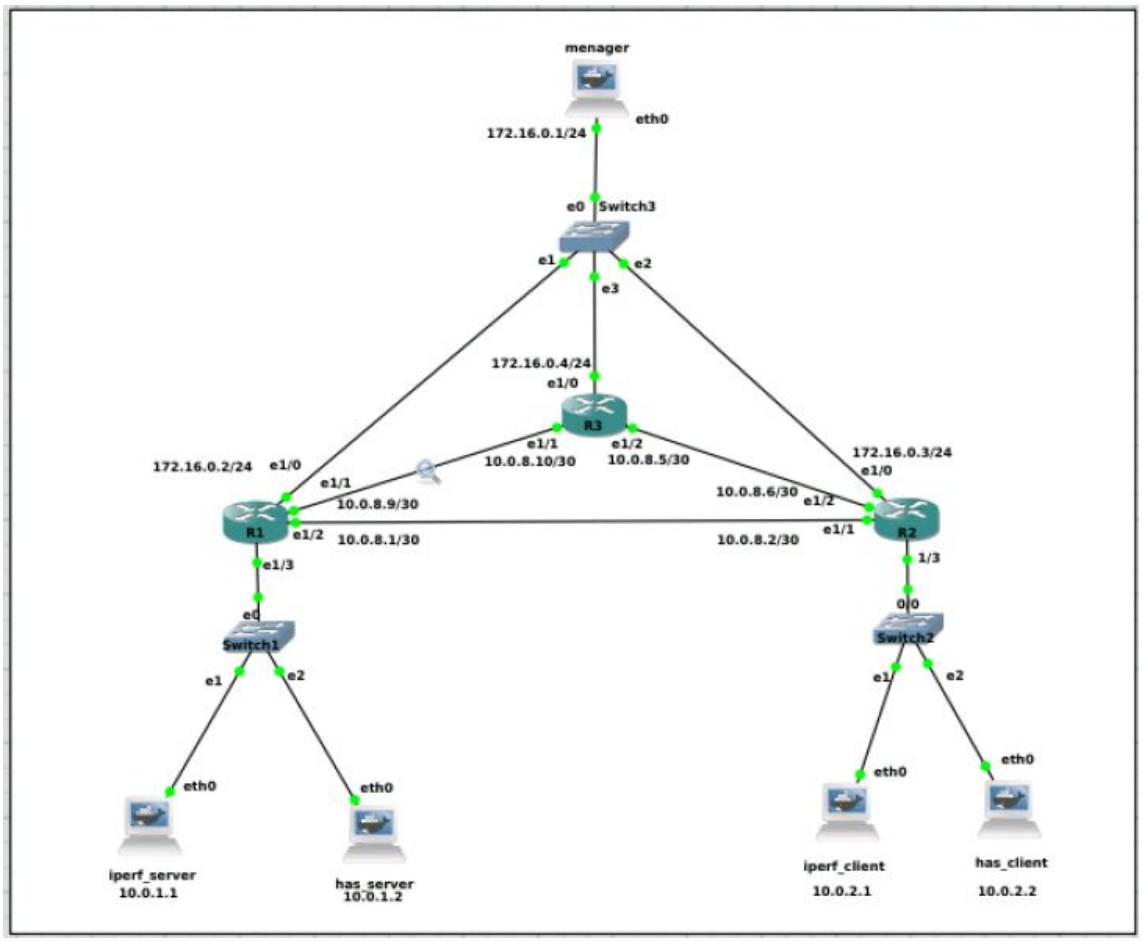

Figure 1: Network Topology

2 Task 1.1 - Configure SNMP

Task a)

In order to set up SNMP agents in each router, the following command was used:

```
snmp-server community ttm4240 RW
```

The community name was set to *ttm4240*, and the permissions to Read-Write, indicated by *RW*. This was configured on every router.

Task b)

No aditional configuration of the SNMP agents was needed. To query the SNMP agents from the

SNMP manager the IP addresses of the SNMP agents was needed. This was obtained during the configuration of the clients with the command:

```
show ip int br
```

Each router was queried with the following commands, with the router IP specified:

```
snmptable -c ttm4240 -v 2c <ip-address> ipAddressTable
snmptable -c ttm4240 -v 2c <ip-address> ifTable
```

The output of these queries can be found in Table 1, 2, and 3.

Table 1: Router 1

Interface	Interface	mac-address	ip-address	subnet mask
index-2	e1/0	ca:1:4a:e4:0:1c	172.16.0.0	/24
index-3	e1/1	ca:1:4a:e4:0:1d	10.0.8.8	/30
index-4	e1/2	ca:1:4a:e4:0:1e	10.0.8.0	/30
index-5	e1/3	ca:1:4a:e4:0:1f	10.0.1.0	/24

Table 2: Router 2

Interface	Interface	mac-address	ip-address	subnet mask
index-2	e1/0	ca:2:4a:f4:0:1c	172.16.0.0	/24
index-3	e1/1	ca:2:4a:f4:0:1d	10.0.8.0	/30
index-4	e1/2	ca:2:4a:f4:0:1e	10.0.8.4	/30
index-5	e1/3	ca:2:4a:f4:0:1f	10.0.2.0	/24

Table 3: Router 3

Interface	Interface	mac-address	ip-address	subnet mask
index-2	e1/0	ca:3:4b:4:0:1c	172.16.0.0	/24
index-3	e1/1	ca:3:4b:4:0:1d	10.0.8.8	/30
index-4	e1/2	ca:3:4b:4:0:1e	10.0.8.4	/30

Task c)

Consequence for error situations: SNMP runs over UDP which is a connectionless protocol. When a packet do not reach its destination the sender is not notified if it is received or not. The SNMP clients sends responses to get and set operations, meaning the manager can detect errors with an absence of a response. There are no mandatory trap responses in SNMP meaning the manager may not know if a trap was sent from a client and then lost.

What you observe if no agent is running:

```
root@menager:/# snmptable -c ttm4240 -v 2c 172.16.0.3 ifTable
Timeout: No Response from 172.16.0.3
```

What happens in case of packets loss:

```
root@menager:/# snmptable -c ttm4240 -v 2c 172.16.0.3 ifTable
Timeout: No Response from 172.16.0.3
```

What you observe if wrong community string is used:

```
root@menager:/# snmptable -c wrong -v 2c 172.16.0.3 ifTable
Timeout: No Response from 172.16.0.3
```

The protocols found in the packet capture is: CDP - Cisco Discovery Protocol, Configuration Test Protocol (LOOP), SNMP and ARP. As discussed above SNMP runs over UDP.

The following snmpget command [1] was used:

```
snmpget -c ttm4240 -v 2c 172.16.0.2 system.sysDescr.0
```

In the console, the OID are specified with the name of the group followed by the object name and a 0 since it is the only leaf object. In the packet capture, the whole ISO OID tree path is specified: 1.3.6.1.2.1.1.1.0, see Figure 2.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	ca:01:4a:e4:00:1c	ca:01:4a:e4:00:1c	LOOP	60	Reply
2	1.811793	ca:03:4b:04:00:1c	CDP/VTP/DTP/PAgP/UD…	CDP	359	Device ID: R3 Port ID: Ethernet1/0
3	7.523863	ca:02:4a:f4:00:1c	CDP/VTP/DTP/PAgP/UD…	CDP	364	Device ID: R2 Port ID: Ethernet1/0
4	10.004560	ca:01:4a:e4:00:1c	ca:01:4a:e4:00:1c	LOOP	60	Reply
5	11.279927	172.16.0.1	172.16.0.2	SNMP	86	get-request 1.3.6.1.2.1.1.1.0
6	11.287984	172.16.0.2	172.16.0.1	SNMP	345	get-response 1.3.6.1.2.1.1.1.0
7	16.510079	b6:23:2f:f9:bb:c9	ca:01:4a:e4:00:1c	ARP	42	Who has 172.16.0.2? Tell 172.16.0.1
8	16.514382	ca:01:4a:e4:00:1c	b6:23:2f:f9:bb:c9	ARP	60	172.16.0.2 is at ca:01:4a:e4:00:1c

Figure 2: Packet capture when running a snmpget command.

For each router the system contact and system location was set to ‘guttaboys@ttm4240.kp’ and ‘10.212.142.126’ respectively using the following commands [2]:

```
snmpset -c ttm4240 -v 2c <IP address> system.sysContact.0 s guttaboys@ttm4240.kp
snmpset -c ttm4240 -v 2c <IP address> system.sysLocation.0 s 10.212.142.126
```

The SNMP manager takes the ISO value after mib-2 as input. The snmpwalk on ‘iso.org.dod.internet.mgmt.mib-2.system is 1.3.6.1.2.1’ was achieved with the following command [3]:

```
snmpwalk -c ttm4240 -v 2c 172.16.0.2 system
```

With the output:

```
SNMPv2-MIB::sysDescr.0 = STRING: Cisco IOS Software, 7200 Software (C7200-ADVENTERPRI
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2011 by Cisco Systems, Inc.
Compiled Mon 26-Sep-11 21:34 by prod_rel_team
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.9.1.222
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (26382050) 3 days, 1:17:00.50
SNMPv2-MIB::sysContact.0 = STRING: guttaboys@ttm4240.kp
SNMPv2-MIB::sysName.0 = STRING: R1
SNMPv2-MIB::sysLocation.0 = STRING: 10.212.142.126
SNMPv2-MIB::sysServices.0 = INTEGER: 78
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (0) 0:00:00.00
```

No.	Time	Source	Destination	Protocol	Length	Info
3	11.615093	172.16.0.1	172.16.0.2	SNMP	84	get-next-request 1.3.6.1.2.1.1
4	11.648104	172.16.0.2	172.16.0.1	SNMP	345	get-response 1.3.6.1.2.1.1.1.0
5	11.648737	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.1.0
6	11.668270	172.16.0.2	172.16.0.1	SNMP	95	get-response 1.3.6.1.2.1.1.2.0
7	11.668745	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.2.0
8	11.678356	172.16.0.2	172.16.0.1	SNMP	90	get-response 1.3.6.1.2.1.1.3.0
9	11.678826	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.3.0
10	11.688580	ca:01:4a:e4:00:1c	ca:01:4a:e4:00:1c	LOOP	60	Reply
11	11.698575	172.16.0.2	172.16.0.1	SNMP	106	get-response 1.3.6.1.2.1.1.4.0
12	11.698994	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.4.0
13	11.708687	172.16.0.2	172.16.0.1	SNMP	88	get-response 1.3.6.1.2.1.1.5.0
14	11.709086	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.5.0
15	11.718784	172.16.0.2	172.16.0.1	SNMP	100	get-response 1.3.6.1.2.1.1.6.0
16	11.719169	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.6.0
17	11.728892	172.16.0.2	172.16.0.1	SNMP	87	get-response 1.3.6.1.2.1.1.7.0
18	11.730014	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.7.0
19	11.779289	172.16.0.2	172.16.0.1	SNMP	87	get-response 1.3.6.1.2.1.1.8.0
20	11.779705	172.16.0.1	172.16.0.2	SNMP	86	get-next-request 1.3.6.1.2.1.1.8.0
21	11.799462	172.16.0.2	172.16.0.1	SNMP	87	get-response 1.3.6.1.2.1.2.1.0

Figure 3: Packet capture of the snmpwalk on system.

Figure 3 shows the packet capture on the link while running this command. Snmpwalk uses the get-next-request message, which is responded on with the value of next OID according to lexicographical order [4]. Snmpwalk continues to request the next OID value until the agent responds with a OID that is outside the group specified in the command.

3 Task 1.2 - Traffic monitoring with SNMP: Detecting traffic patterns

Discussion
It was decided to monitor the objects 'ifInUcastPkts' and 'ifOutUcastPkts' for relevant interfaces. 'ifInUcastPkts' is the number of unicast packages delivered to a higher layer protocol, and 'ifOutUcastPkts' is the number of packets that was requested by a higher layer protocol, not addressed to a multicast or broadcast address. The monitoring data was found with the snmpdelta command, which outputs the difference in the object values for a given interval. The following command [5] was used:

```
snpmdelta -v 2c -c ttm4240 -Cp 5 -Cl -Cs 172.16.0.2 ifInUcastPkts.5
ifOutUcastPkts.5
```

The traffic was monitored in R1 on interface 5, the e1/3 interface. This is because all Iperf and HAS packages traverse this router and interface. The time between each pulling was set to 5 second, and the monitoring period to 20 minutes.

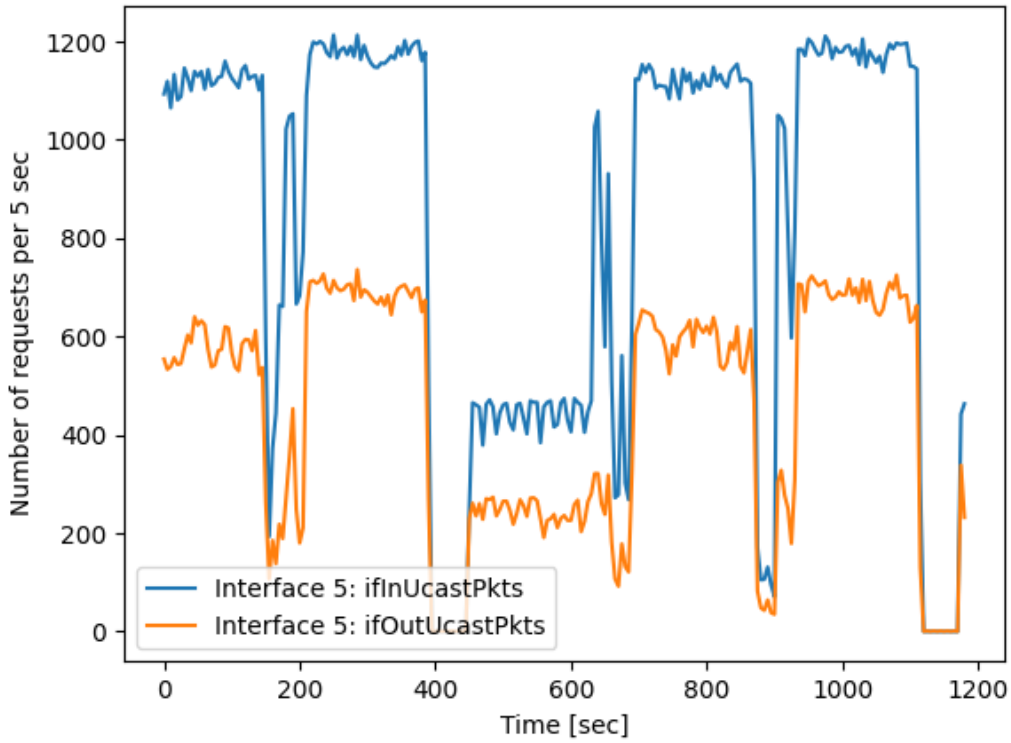

Figure 4: Plot of the amount of packets aggregated every 5 seconds with 'ifInUcastPkts' and 'ifOutUcastPkts' on e1/3 in R1.

The result is shown in Figure 4. The traffic pattern seems to repeat every 700 seconds. Interestingly, about half of the packages are monitored with 'ifOutUcastPkts' compared to with 'ifOutUcastPkts'. This shows there is a lot more packages delivered than requested.

a) Frequent measurements
The advantage of monitoring every 5 seconds compared to every 5 minutes is the higher resolution of the data. The data received through SNMP is the average of packets received by the agent between each monitoring. With a longer time between monitoring there may be spikes and valleys in the traffic that is not seen in the monitoring.

The disadvantage with a short time between each monitoring is that it introduces traffic in the network, which may result in packet loss and delay. However, in the lab setup, the links between the SNMP agents and SNMP manager is not shared with the user data, meaning a high monitoring rate will have a low impact on the network traffic.

b) Identify which protocol?
In SNMP there is only possible to monitor traffic by requesting specific OID. Since there is no application specific OIDs specified in SNMP it is not possible to differentiate traffic from different applications. There is possible to different traffic based on the interfaces. Since both iperf and has traffic comes from interface e1/3 there is not possible to differentiate the applications in this task.

4 Task 1.3 - Network Reconfiguration

a)
To change the topology Anna have to reconfigure `R1` and `R3`. In both routers the mask of interface `e1/3` need to be split between the new interface `e2/0` and `e2/1`. The servers, clients and new interfaces in `R1` and `R3` need to be assigned an IP within its mask. The `HAS` and `Iperf` servers and clients also needs to be reconfigured to reach each other in the new network.

Discussion
Before monitoring could start we needed to re-enable the SNMP routers as in Task 1.1. The following commands were used:

```
snmp-server community ttm4240 RW
write memory
```

It was decided to monitor R1 on both the interfaces towards the `HAS` server and `iperf` server. To find the index of these interfaces the following command was used:

```
snmptable -c ttm4240 -v 2c 172.16.0.2 ifTable
```

The output showed that interface e2/0, towards the `iperf` server, have index 6, and interface `e2/1`, towards the `HAS` server, have index 7. The objects 'ifInUcastPkts' and 'ifOutUcastPkts' was monitored on both theses interfaces, every 5 seconds, for 20 minutes with the following command:

```
snpmdelta -v 2c -c ttm4240 -Cp 5 -Cl -Cs 172.16.0.2
ifInUcastPkts.6 ifInUcastPkts.7 ifOutUcastPkts.6 ifOutUcastPkts.7
```

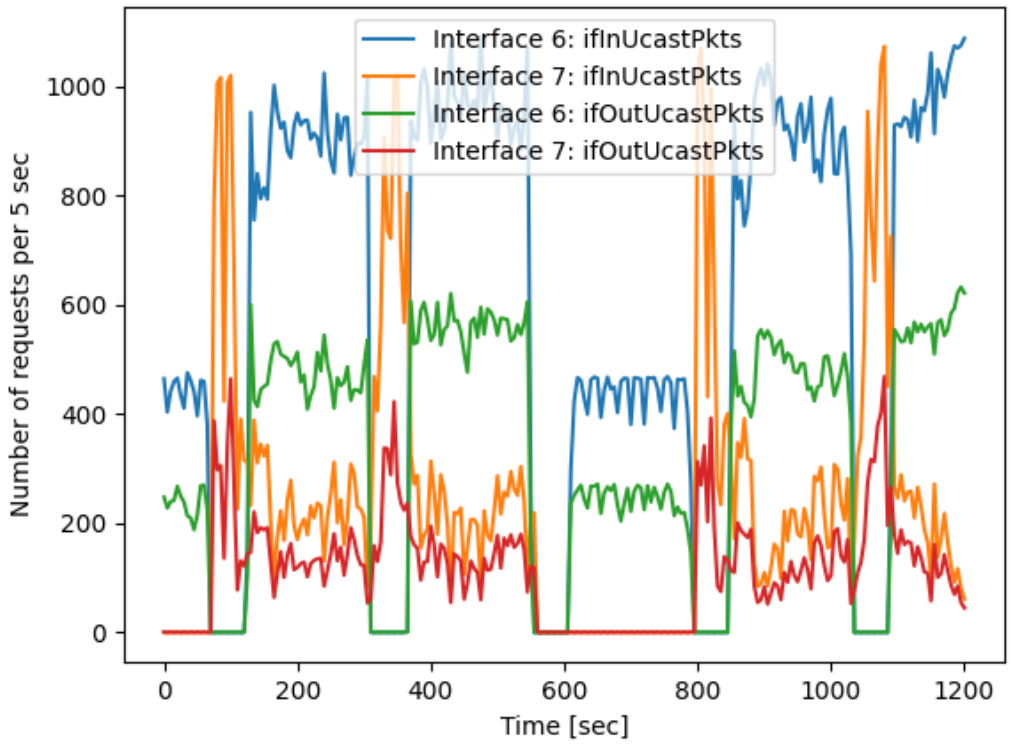

Figure 5: Plot of the amount of packets aggregated every 5 seconds on interface e2/0 and e2/1 in R1.

The result is shown in Figure 5. This also shows that traffic is periodical about every 700 seconds

from both the has server and the `iperf` server. For each period the `iperf` traffic in interface 6 have three periods of 200 seconds with high traffic. One period with about 500 packets delivered every 5 seconds, and two with about 1000 packets delivered every 5 second. The HAS traffic, in interface 7, have two short spikes per period with about 1000 packets per second. These spikes happen between the `iperf` periods with traffic. Also in this task the packets delivered in ifInUcastPkts is much higher than packets requested in ifOutUcastPkts.

5 Task 1.4 - Traffic Monitoring with NetFlow

a)

Commands used in R1: [6]

```
ip flow-export source e1/0
ip flow-export destination 172.16.0.1 7777
ip flow-cache timeout active 1
ip flow-export version 5
sh ip flow export
```

Commands used in manager: [7]

```
flow-capture -w /tmp/flows 0/0/7777
```

Flow files could then be found in the manager's /tmp directory.

Output from `sh ip flow export`:

```
R1#sh ip flow export
Flow export v5 is enabled for main cache
  Export source and destination details :
  VRF ID : Default
    Source(1)       172.16.0.2 (Ethernet1/0)
    Destination(1)  172.16.0.1 (7777)
  Version 5 flow records
  6799 flows exported in 1869 udp datagrams
```

It was decided to export both the egress and ingress flows of interface e1/3, towards switch 1. This means that most of the traffic is sent from the servers to the client is the ingress flow. The egress flow is the acknowledgements from the clients, to the servers. This was done with the following commands:

```
int e1/3
ip flow ingress
ip flow egress
```

b)

We used NetFlow Version 5 to monitor the traffic for both *HAS* and *iperf3*. Figure 6 to Figure 9 shows the plots for this.

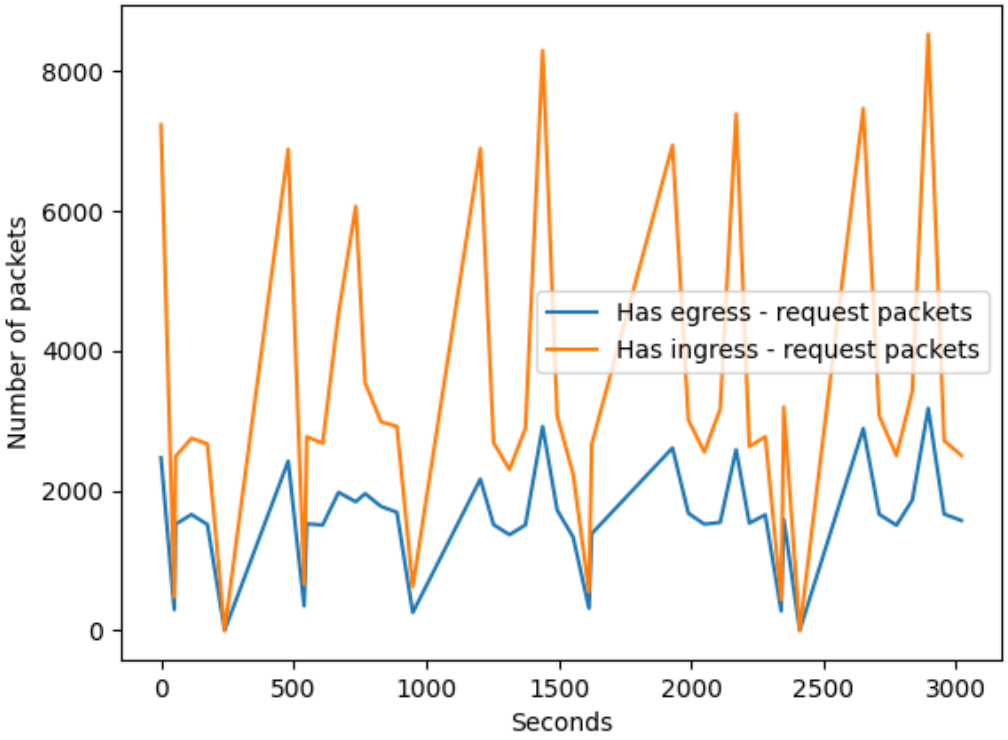

Figure 6: HAS ingress and egress requests group per every 5 seconds

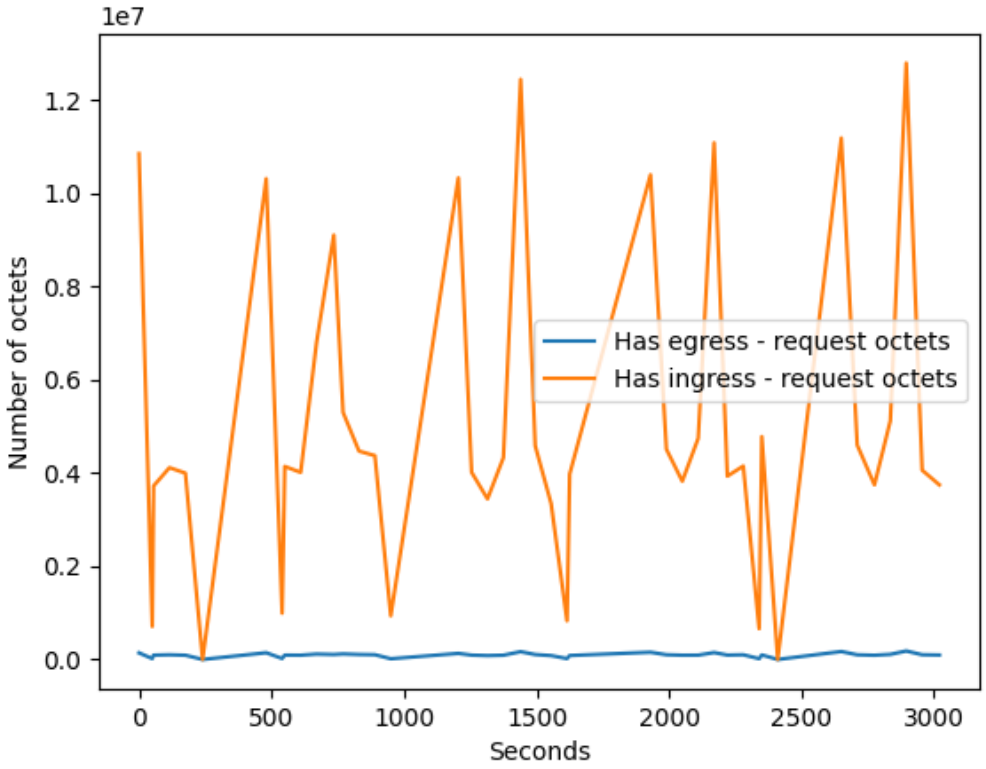

Figure 7: HAS ingress and egress octets group by every 5 seconds

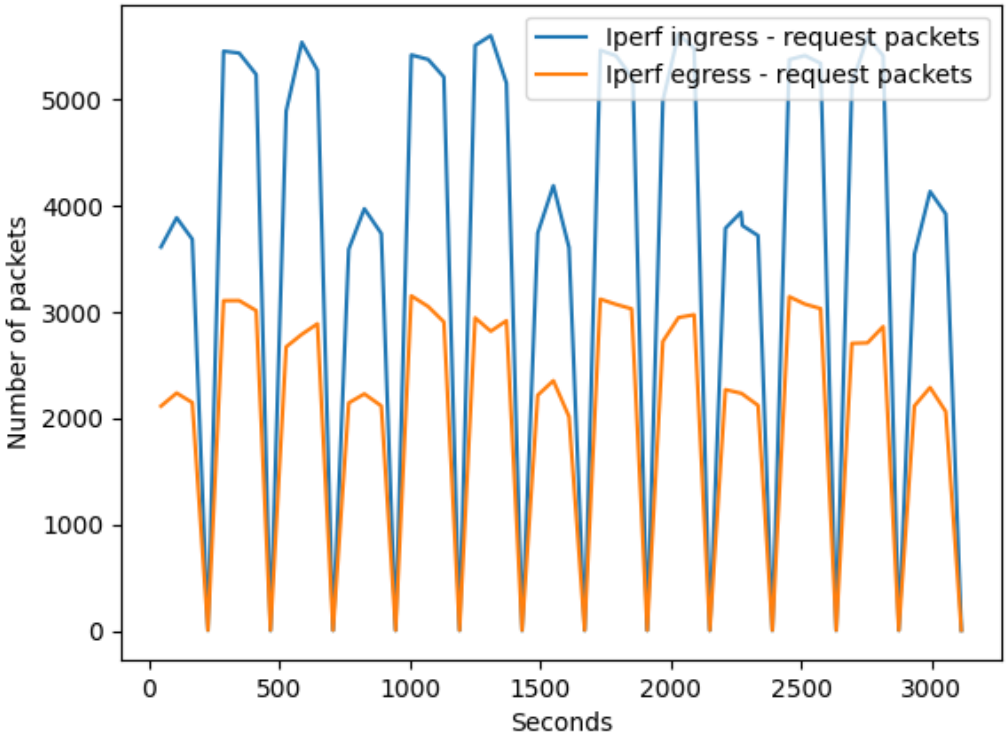

Figure 8: Iperf ingress and egress requests group by every 5 seconds

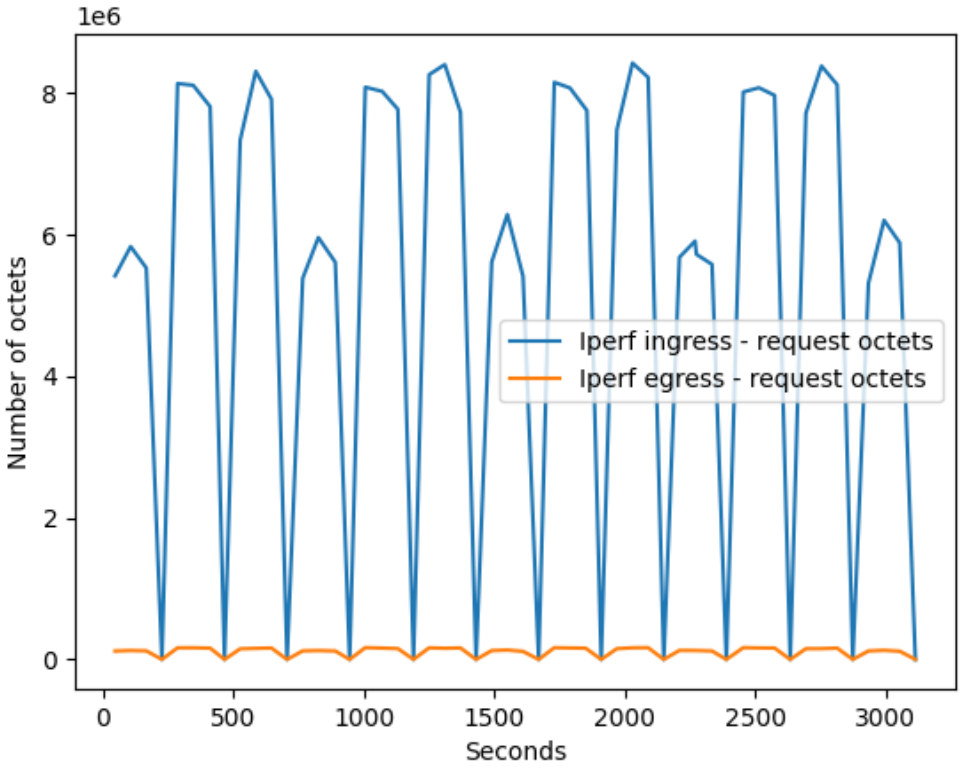

Figure 9: Iperf ingress and egress octets group by every 5 seconds

Figure 10: The average request time for a packet per 5 seconds

As seen in Figure 6 and Figure 8, the number of packets seem to match what was seen with SNMP quite well for Iperf, however the HAS traffic does not seem to fit that good with the earlier plots. As seen, the Has has higher spikes for the number of requests seen with Netflow. This is probably due to our aggregation of the data, and the SNMP delta commands might be more intelligent when aggregating traffic flow than our algorithm. However, the spikes seem to be in the right periodic position, and to match the pattern seen in Figure 5 from 1.3.

What is interesting is that the number of egress packets, that means packets that are sent from the client to the server, follows the number of ingress packets quite well, as seen in Figure 6 and Figure 8. However, the Octets for egress data, both for HAS and iperf is minimal compared to the octets from ingress data. This is probably due to the clients sending mostly retransmitting messages or acks. We found this observation very interesting. This can be seen in Figure 7 and Figure 9

Also worth to mention is that the request times does not have that much variation and not that high spikes, as seen in 10

c)
SNMP and NetFlow are both used for network management and monitoring. They are often used simultaneously, as they can complement each other, by providing different insight of the traffic. SNMP is primarily used for querying information about network devices, measure bandwidth usage, and can be used for fault management by setting up triggers that will send notifications on certain events.

NetFlow on the other hand is used to measure the network traffic more accurately. SNMP only shows how much data is passing, NetFlow on the other hand shows what type of data, giving the information about what protocol, and which ports. This gives the network administrator a more in depth view of the network, and can help detecting problems with the network, that SNPMP sometimes can not.

SNMP - advantages:

- Open standard - widely supported among hardware vendors [8]
- Easy to implement [8]
- Lightweight - low resource consumption [8]
- Monitor CPU utilization, temperature, memory [8]
- Fault Management[8]

SNMP - disadvantages:

- Scalability
- Cannot differentiate between types of traffic

NetFlow - advantages:

- In depth traffic analysis
- Individual packet level monitoring
- Can be used to identify network congestions

NetFlow - disadvantages:

- High bandwidth usage
- Consumes a lot more disk space if data is being stored

6 Task 1.5 - Rerouting HAS traffic

To reroute HAS traffic from R1 to R2 through R3, we made the following configuration in R1:

```
no ip route 10.0.2.0 255.255.255.0 e1/2
ip route 10.0.2.1 255.255.255.255 e1/2
ip route 10.0.2.2 255.255.255.255 e1/1
```

With these configurations in place, HAS traffic from R1 to R2 was being rerouted through R3.

Using NetFlow, we monitored the network traffic with the rerouting in place. The plots are shown in Figures 11, 12, 13, and 14.

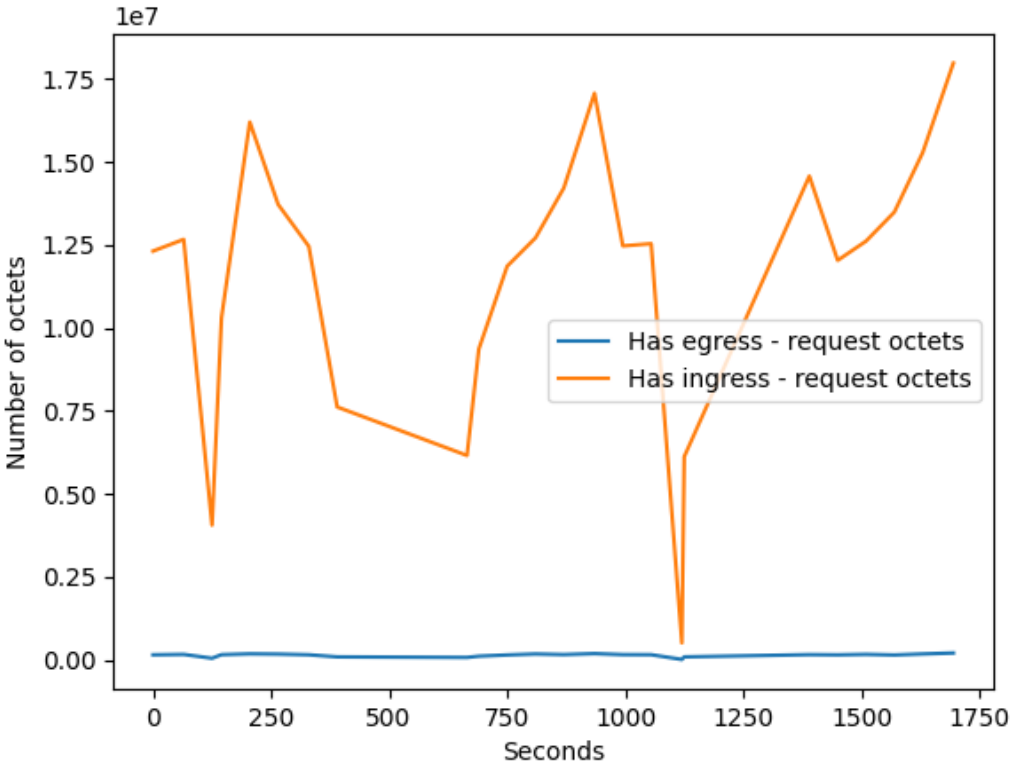

Figure 11: HAS octets after traffic reroute

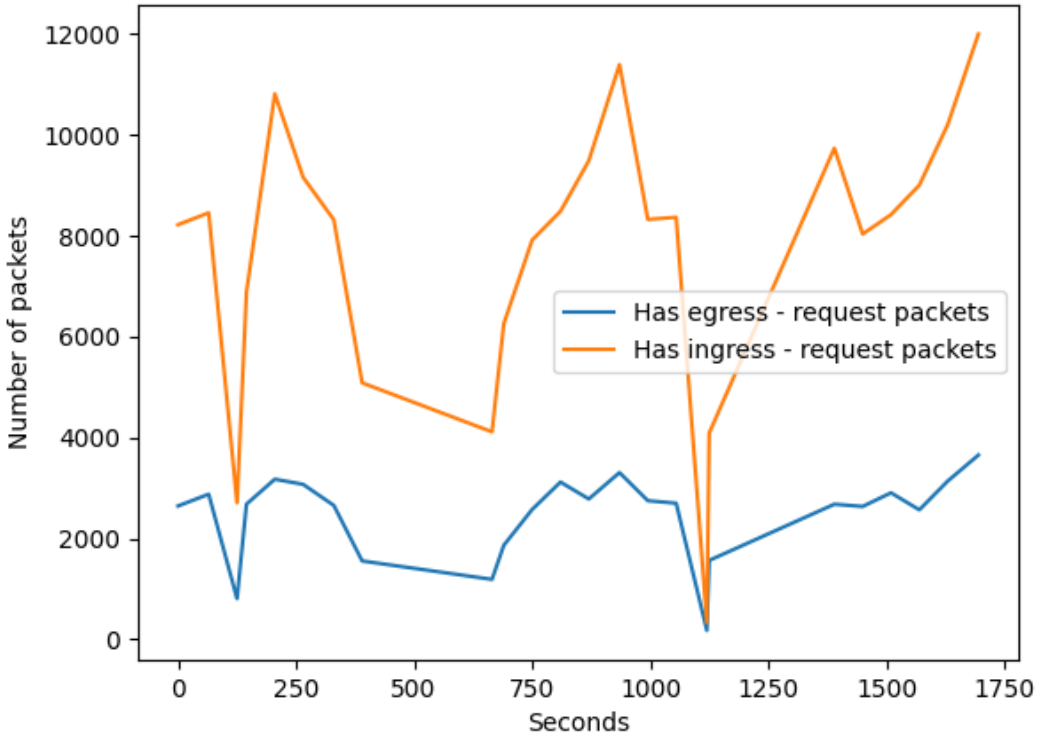

Figure 12: HAS packets after traffic reroute

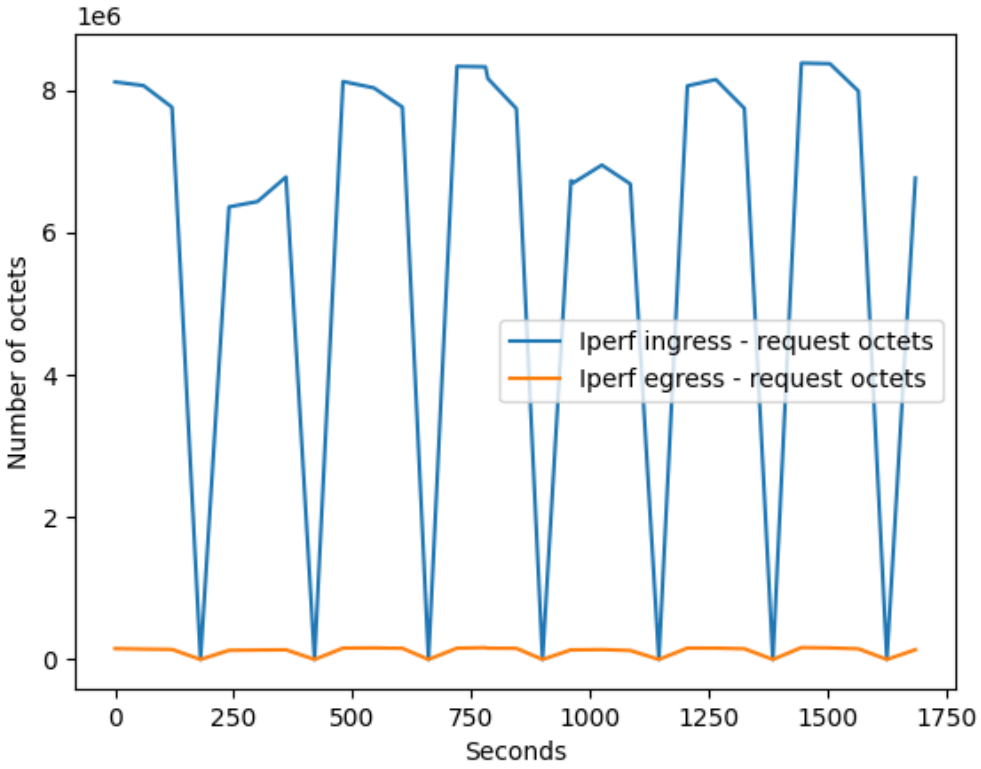

Figure 13: Iperf octets after Has has been rerouted

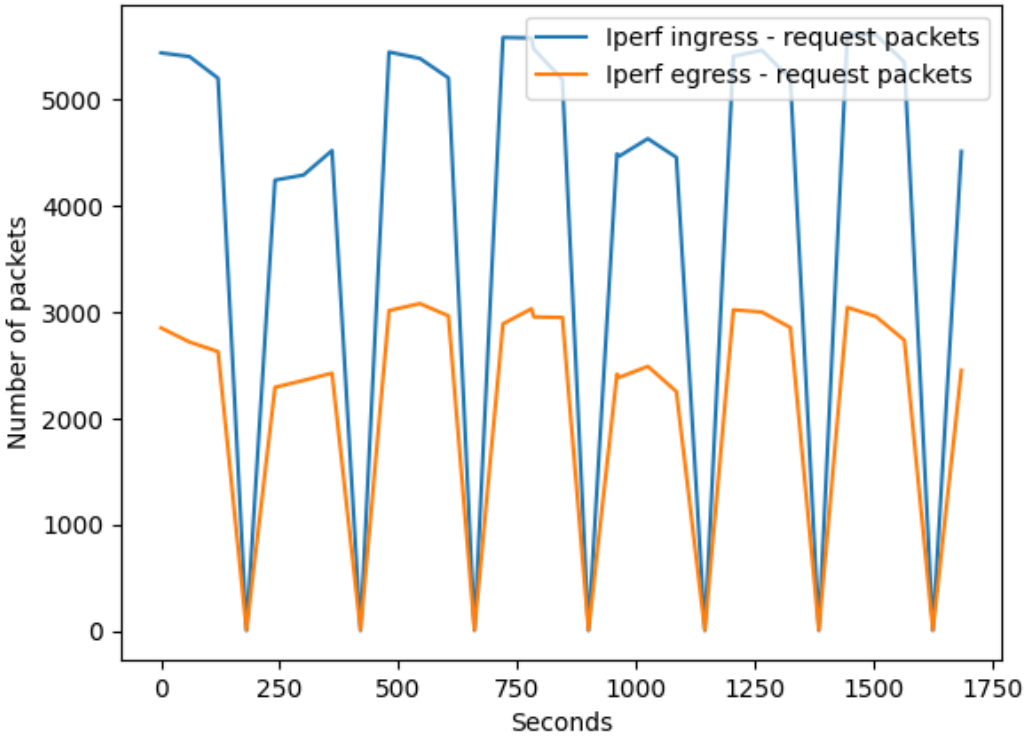

Figure 14: Iperf packets after Has has been rerouted

From the plots, we observe that the amount of octets sent has increased, in comparison to Task 1.4. This is expected, there are now more bandwidth available, since a different link is being used for traffic from R1 to R2 than from R2 to R1. The extra bandwidth allows the server to deliver a video stream of higher quality.

References

1. *snmget docs* http://www.net-snmp.org/docs/man/snmpget.html/. [Online; accessed 07-Nov-2021].
2. *snmp set docs* http://www.net-snmp.org/docs/man/snmpset.html. [Online; accessed 07-Nov-2021].
3. *snmpwalk docs* http://www.net-snmp.org/docs/man/snmpwalk.html. [Online; accessed 07-Nov-2021].
4. Subramanian, M. *Network Management* (2010).
5. *snmdelta docs* http://www.net-snmp.org/docs/man/snmpdelta.html/. [Online; accessed 07-Nov-2021].
6. *Cisco NetFlow Configuration* https://www.cisco.com/c/dam/en/us/td/docs/security/stealthwatch/netflow/Cisco_NetFlow_Configuration.pdf/. [Online; accessed 10-Nov-2021].
7. *Linux.die flow-capture* https://linux.die.net/man/1/flow-capture/. [Online; accessed 10-Nov-2021].
8. *itprc.com SNMP advantages* https://www.itprc.com/what-is-snmp/#What_are_the_Advantages_of_Using_SNMP/. [Online; accessed 10-Nov-2021].