iu-ne-lab-03-Iskander_Nafikov

- **Name**: Iskander Nafikov
- **E-mail**: i.nafikov@innopolis.university
- **Username**: iskanred
- **Hostname**: `lenovo`

---

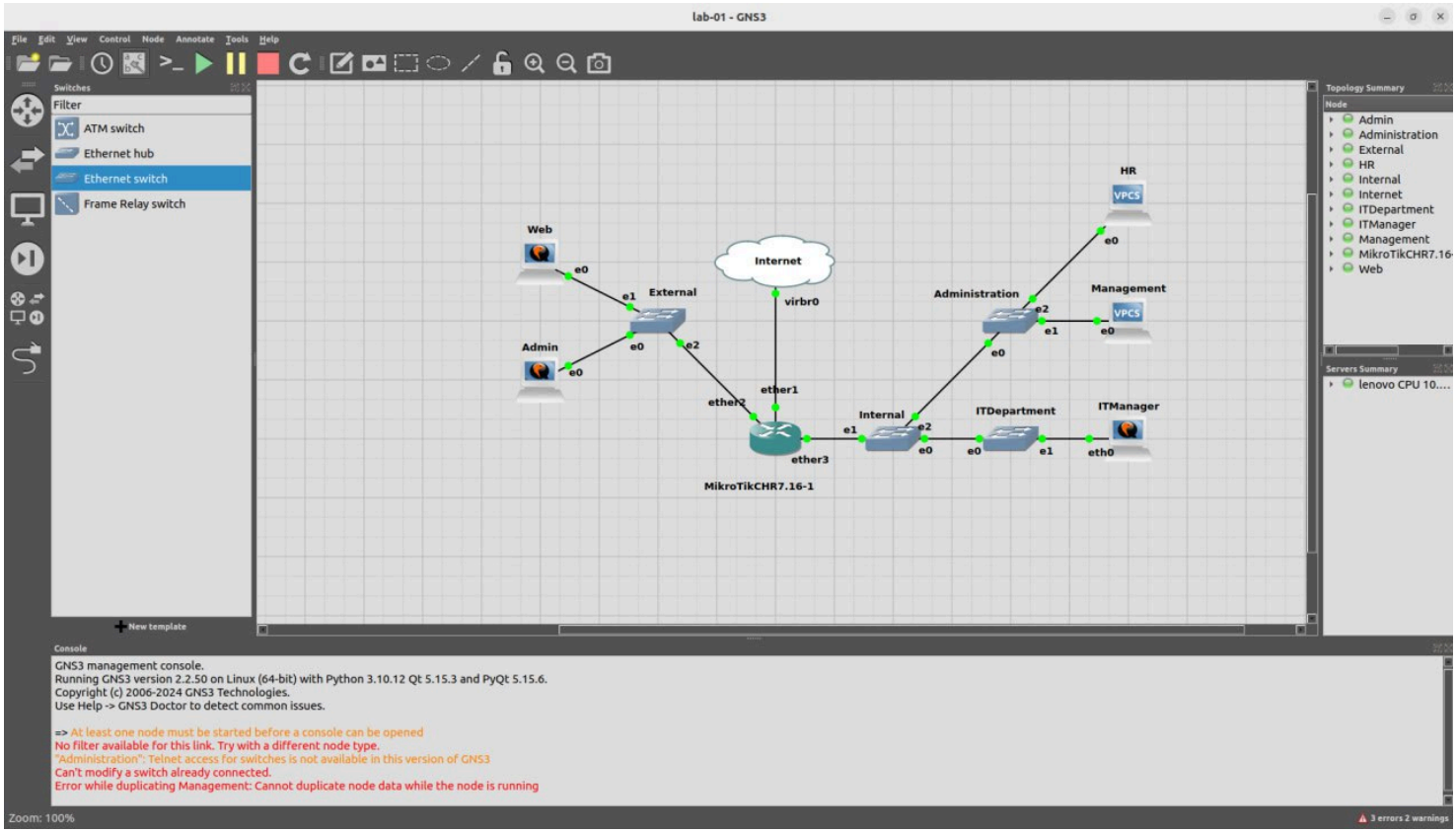
1. Preparation

The preparations were done. I used Mikrotik router.

2. VLANs

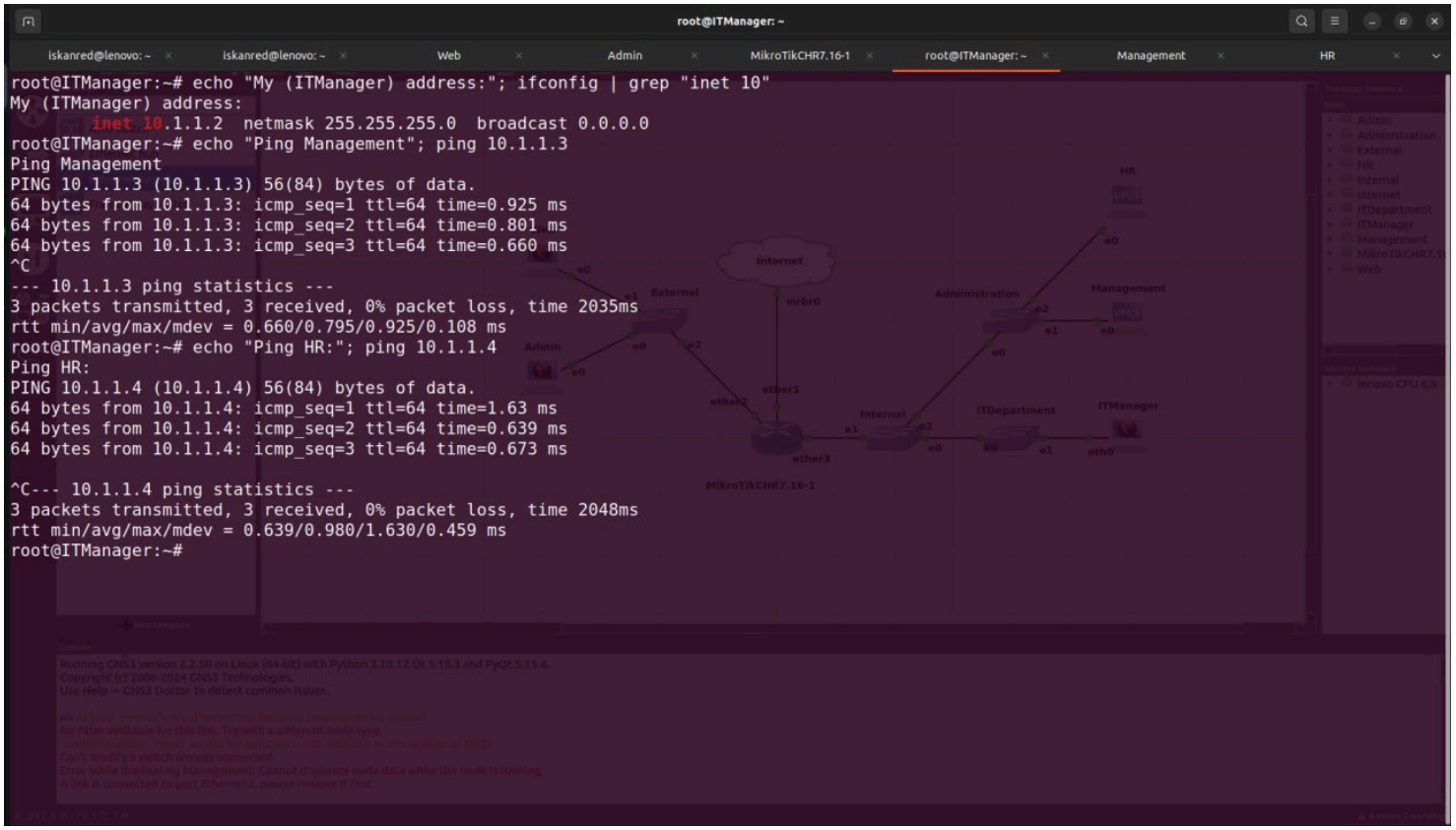
> **a.** `Change the topology of your network to as follows, and make the necessary configurations.`

- I made a topology as follows also added changes to the topology from the previous labs (i.e. changed name of "Worker" node to "ITManager"). The results are the following:

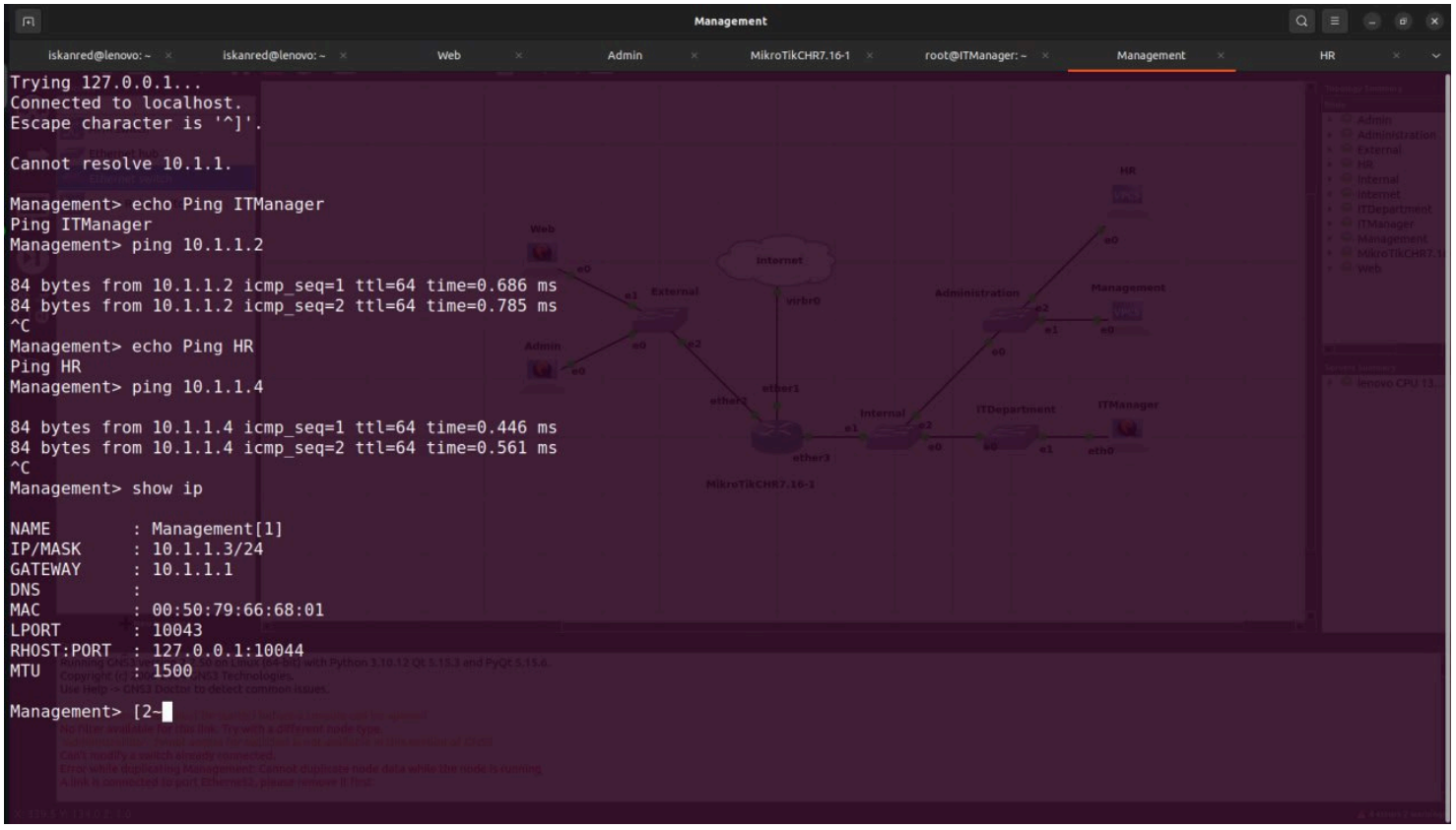


> **b.** `Configure the switches and make sure you have connectivity between the hosts.`

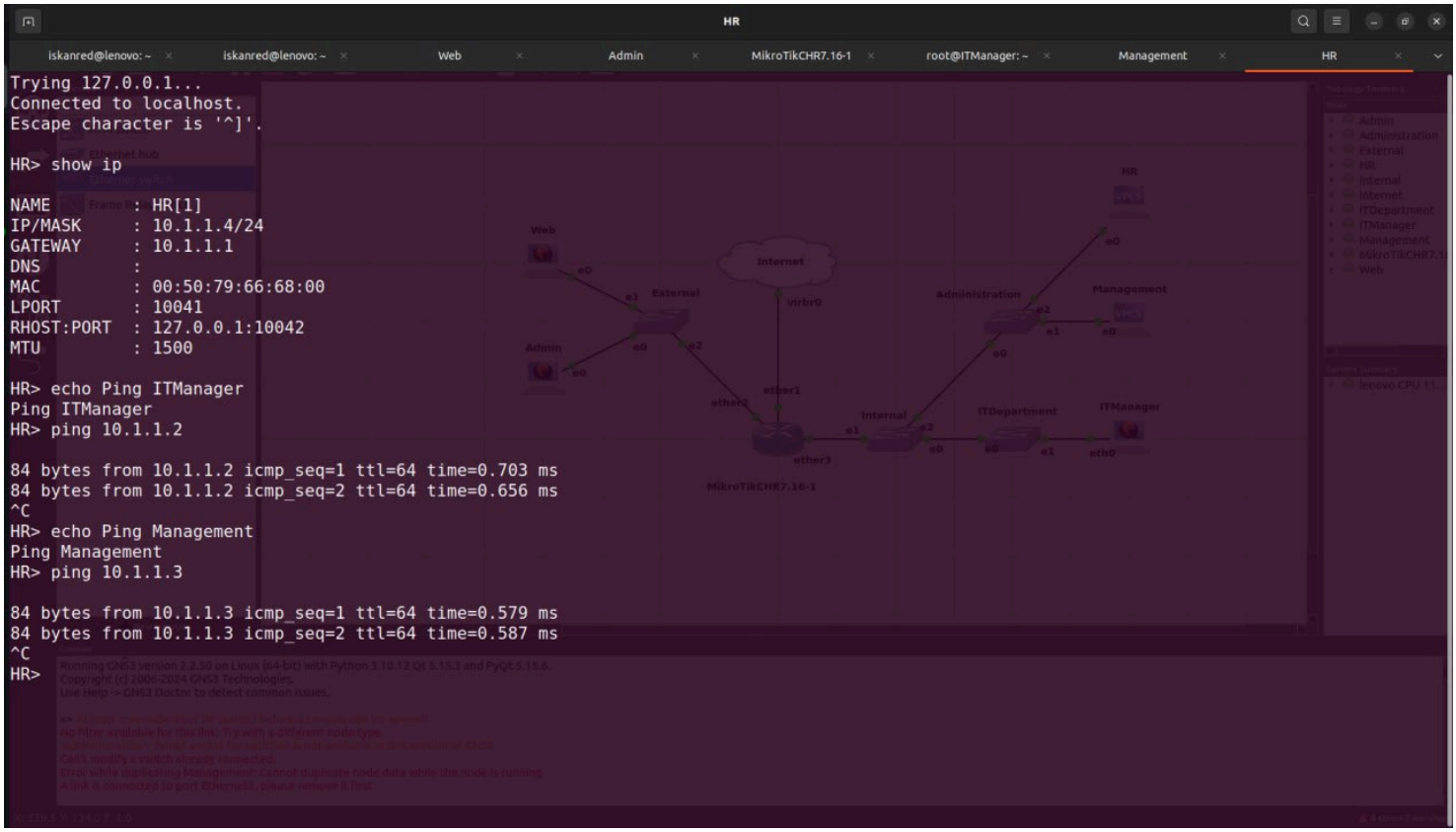
- I checked the connectivity between **ITManager** and Management & HR



- Between **Management** and ITManager and HR



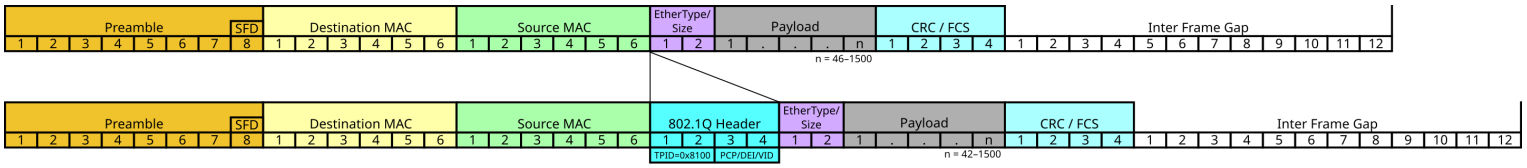
- Between **HR** and ITManager and Management



> **c.** `How do VLANs work at a packet level? What are the two main protocols used for this?`

VLAN is a technology that exists only in 2nd OSI-layer (Link layer). Therefore, it is not correct to mention working at a "packet" level, since packets are entities of the 3rd OSI-layer (Network layer). The more convenient would be to use word "frames". **Note**: Actually it can exist above the 2nd layer, but it is not related to `IEEE 802.1Q`, so I will not consider it

The frames contains special IEEE 802.1Q header



This header has the following structure:

802.1Q tag format

16 bits	3 bits	1 bit	12 bits
TPID	TCI		
	PCP	DEI	VID

Where:

- **TPID** = Tag protocol identifier: A 16-bit field set to a value of `0x8100` in order to identify the frame as an IEEE 802.1Q-tagged frame.

- **TCI** = Tag control information: A 16-bit field containing several sub-fields. The most interesting sub-field for us is:
  - **VID** = VLAN identifier: A 12-bit field specifying the VLAN to which the frame belongs.

Marking a frame with different **VID** is the same as making them belonging to different VLANs.

The main protocols used:

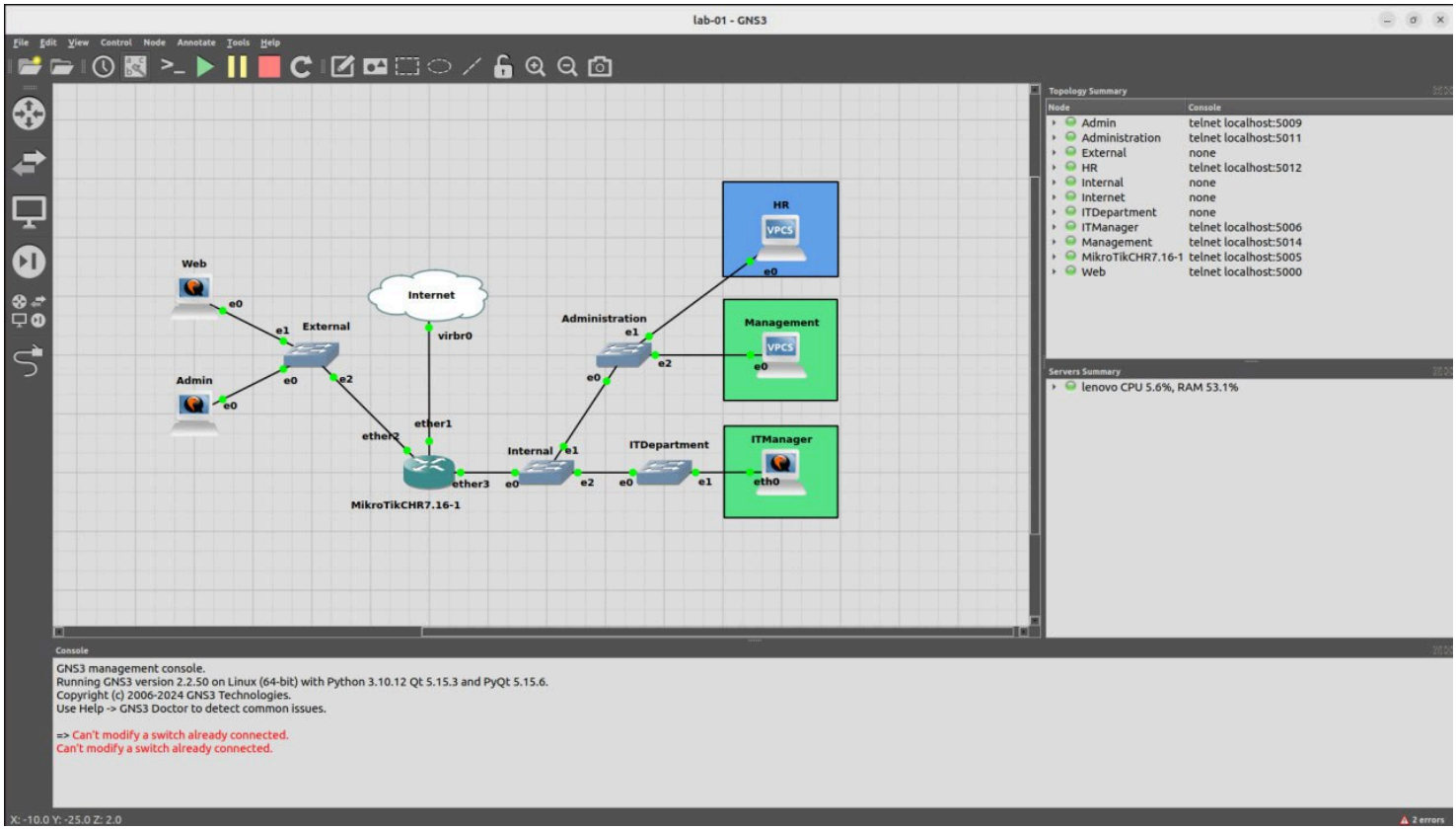
- **IEEE 802.1Q (or Dot1Q)**
- **Q-in-Q** (or **IEEE 802.1ad**) which is an extension to 802.1Q and it allows two or more VLAN headers in the same frame

> **d.** `What is the Native VLAN?`

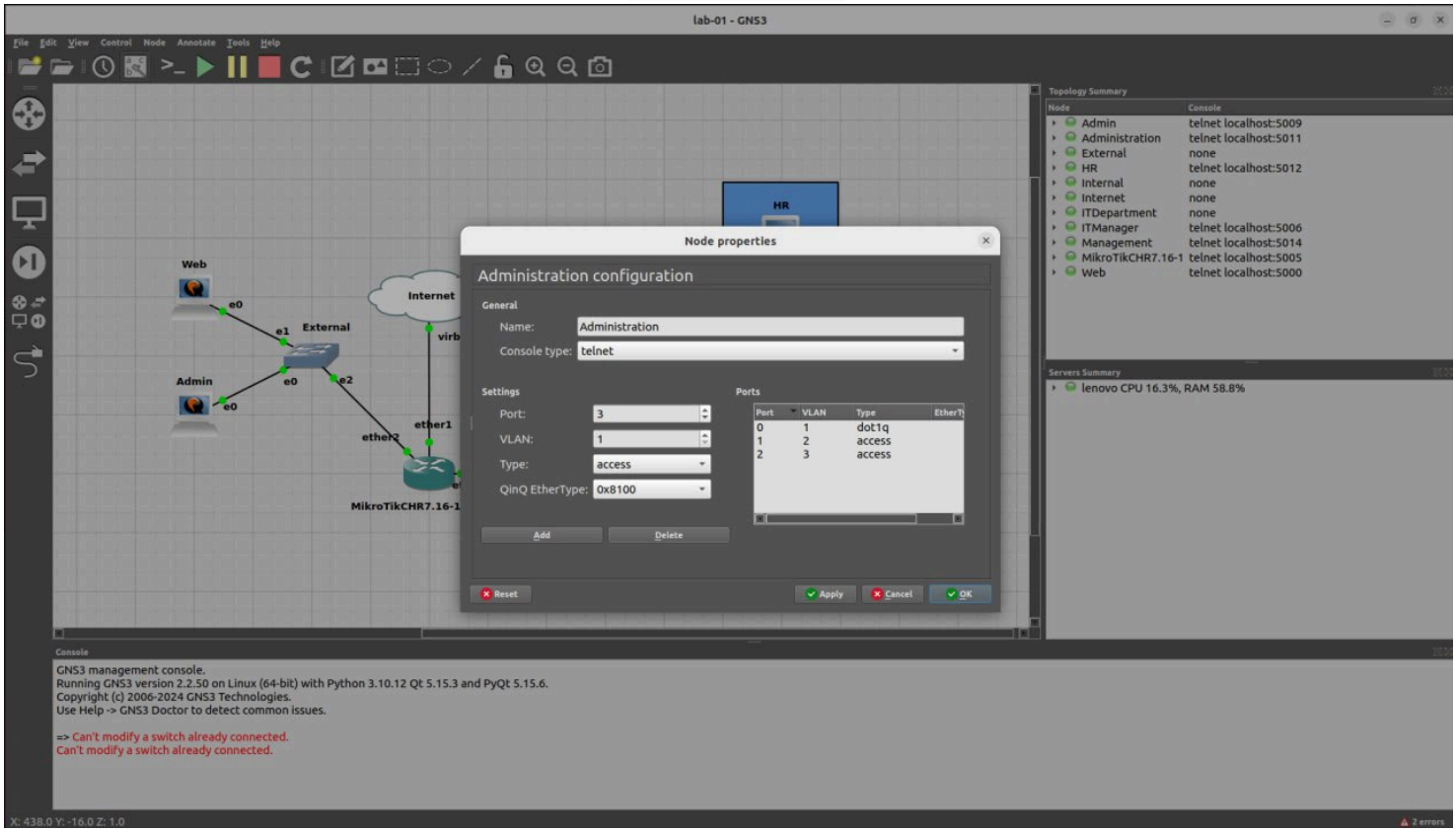
Native VLAN is a concept in the 802.1Q standard that designates a VLAN on a switch where all frames go without a tag, i.e. traffic is transmitted untagged. By default, this is `VLAN 1`. In some switch models, such as Cisco, this can be changed by specifying another VLAN as native. The network switch automatically adds the labels of this VLAN to all received frames that do not have any labels. VLANs based on ports have some limitations. Only one VLAN can receive all the untagged frames!

> **e.** `Configure the VLANs on the switches to isolate the two virtual networks as follow.`

- I configured the topolgy as follows (some changes in port's names because of switches reconfiguration):



- Here is an example of configuration of the switch `Administration`

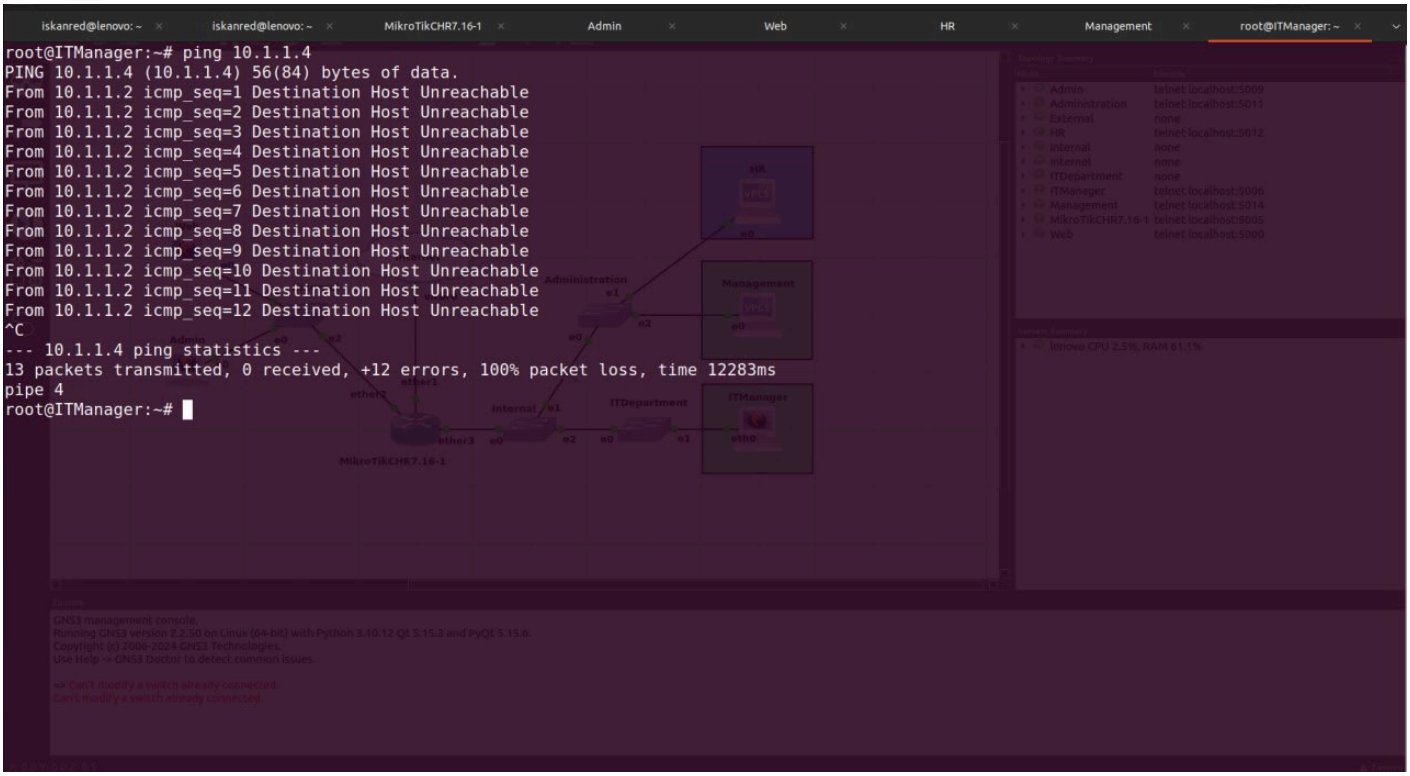


> **f.** `Ping between ITManager and HR, do you have replies? Ping between ITManager and Management, do you have replies? Can you see the VLAN ID in Wireshark?`

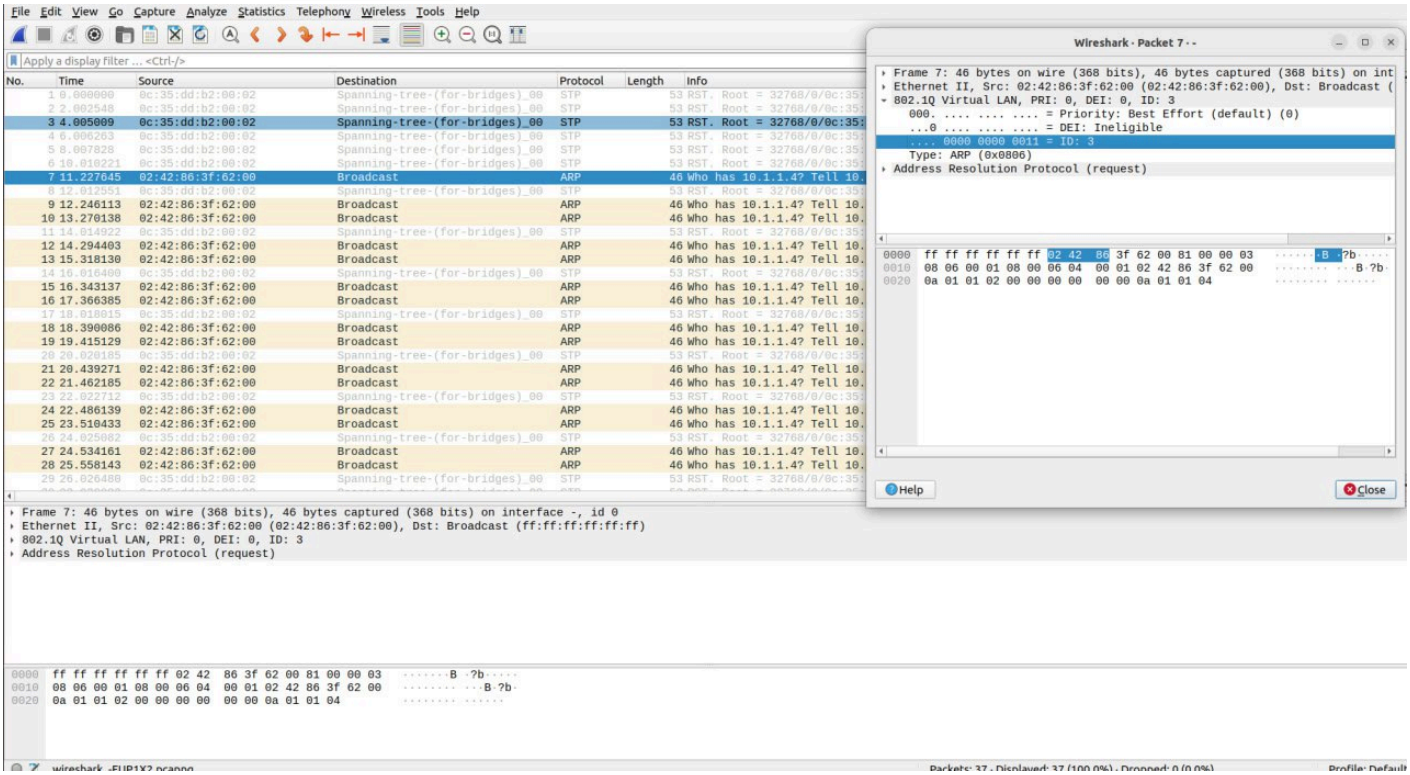
- Ping between `ITManager` and `HR` :
  - The address of `HR` remained the same

```
HR> show ip

NAME        : HR[1]
IP/MASK     : 10.1.1.4/24
GATEWAY     : 10.1.1.1
DNS         :
MAC         : 00:50:79:66:68:00
LPORT       : 10041
RHOST:PORT  : 127.0.0.1:10042
MTU         : 1500
```

- The `HR` did not reply



- I decided to capture the traffic of the link between switch `Internal` and `Administration`. There we can see that ARP frames had `IEEE 802.1Q` ID $= 3$ of VLAN. However, there were no reply for these requests.

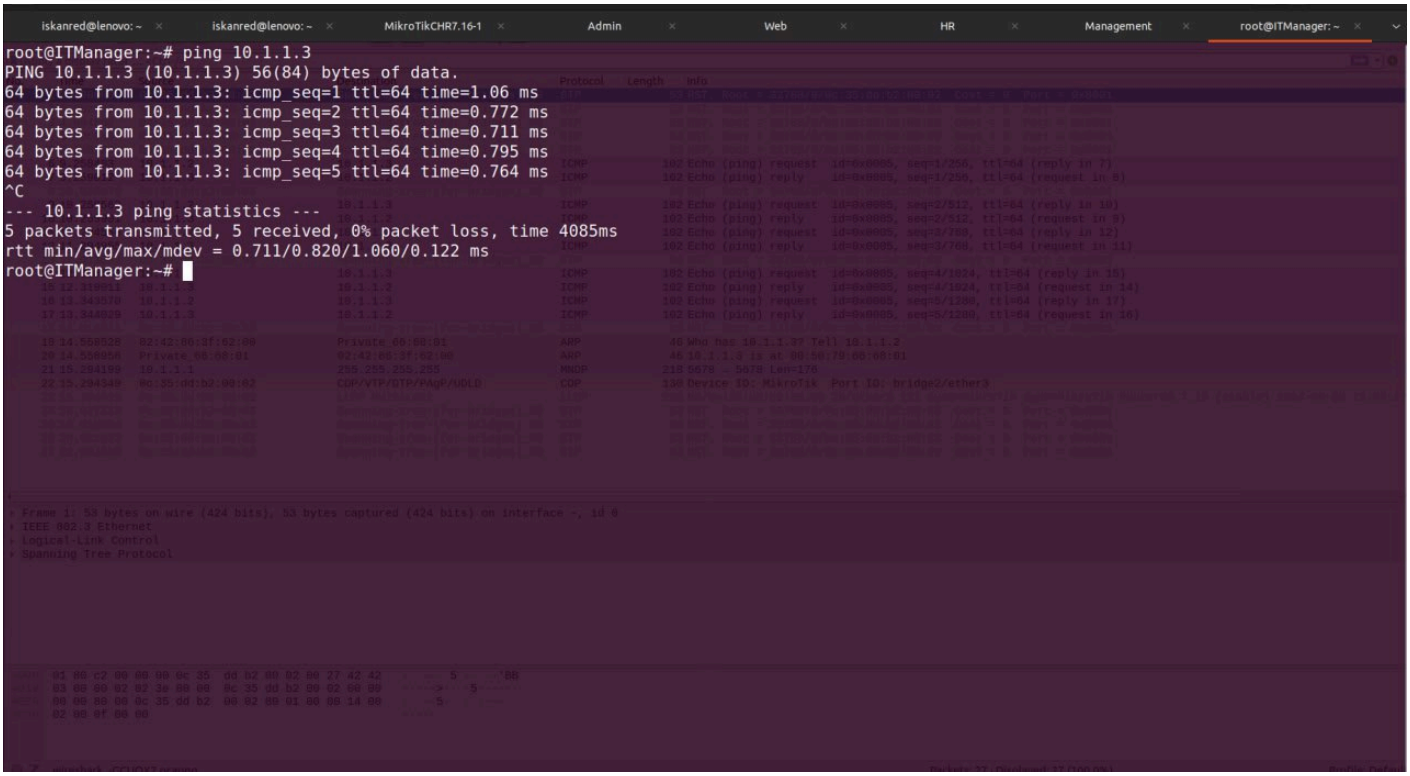


- Ping between `ITManager` and `Management`. The same is applicable here the result is the opposite though:

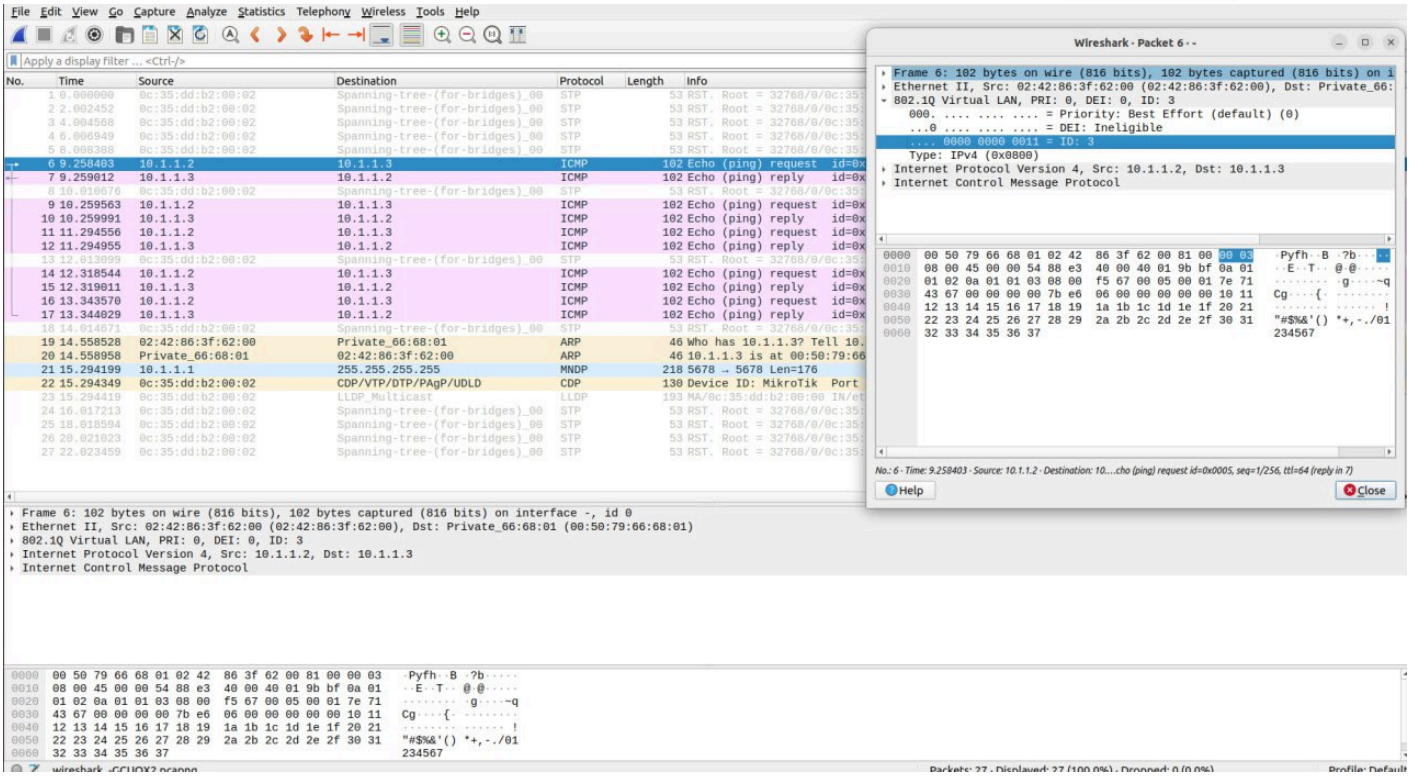
- The address of `Management` remained the same

```
Management> show ip

NAME          : Management[1]
IP/MASK       : 10.1.1.3/24
GATEWAY       : 10.1.1.1
DNS           :
MAC           : 00:50:79:66:68:01
LPORT         : 10043
RHOST:PORT    : 127.0.0.1:10044
MTU           : 1500
```

- The `Management` did reply!



- ICMP requests were replied successfully. Inside their frames we can see the same `IEEE 802.1Q` ID $= 3$



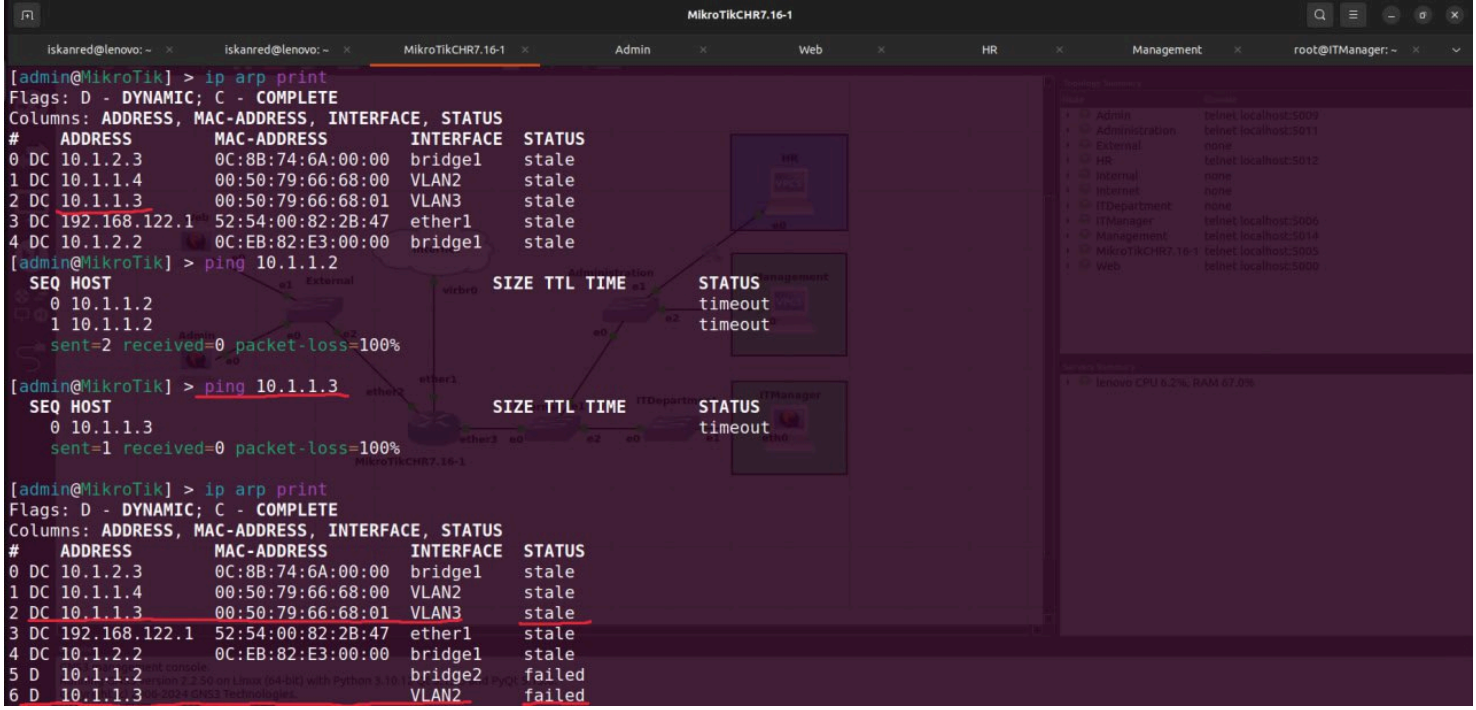
> **g.** `Configure Inter-VLAN Routing between Management VLAN and HR VLAN and Show that you can now ping between them.`

I faced a **problem** with my configuration after configuring Inter-VLAN communication. The problems was that the router always tried to make `ARP` broadcast requests to the **wrong VLAN**:

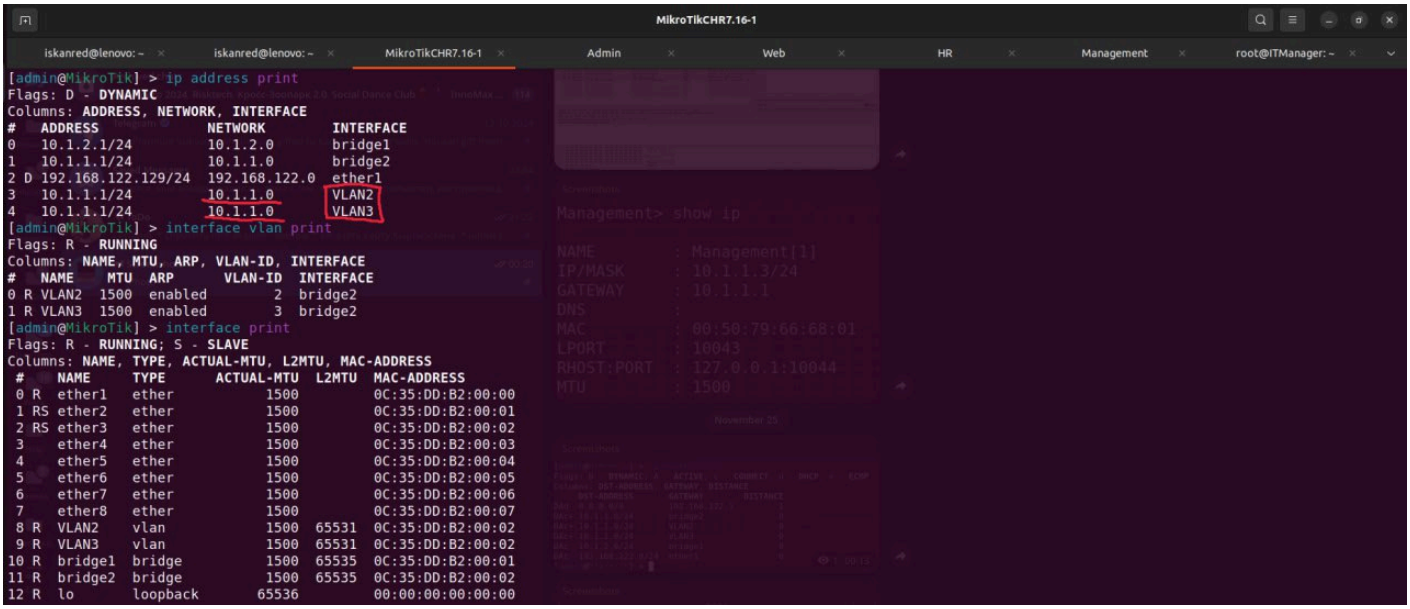
- For example, I tried `10.1.1.2` which was on VLAN $= 3$. However, the router continued searching for this IP with ARP broadcast requests tagging its frame with ID$= 2$ and vice a versa!
- I realized that the problem had been in that router had the same address for three interfaces: simple `bridge2`, `VLAN2`, `VLAN3`

```
[admin@MikroTik] > ip route print
Flags: D - DYNAMIC; A - ACTIVE; c - CONNECT, d - DHCP; + - ECMP
Columns: DST-ADDRESS, GATEWAY, DISTANCE
     DST-ADDRESS         GATEWAY         DISTANCE
DAd  0.0.0.0/0           192.168.122.1          1
DAc+ 10.1.1.0/24         bridge2                0
DAc+ 10.1.1.0/24         VLAN2                  0
DAc+ 10.1.1.0/24         VLAN3                  0
DAc  10.1.2.0/24         bridge1                0
DAc  192.168.122.0/24    ether1                 0
[admin@MikroTik] >
```

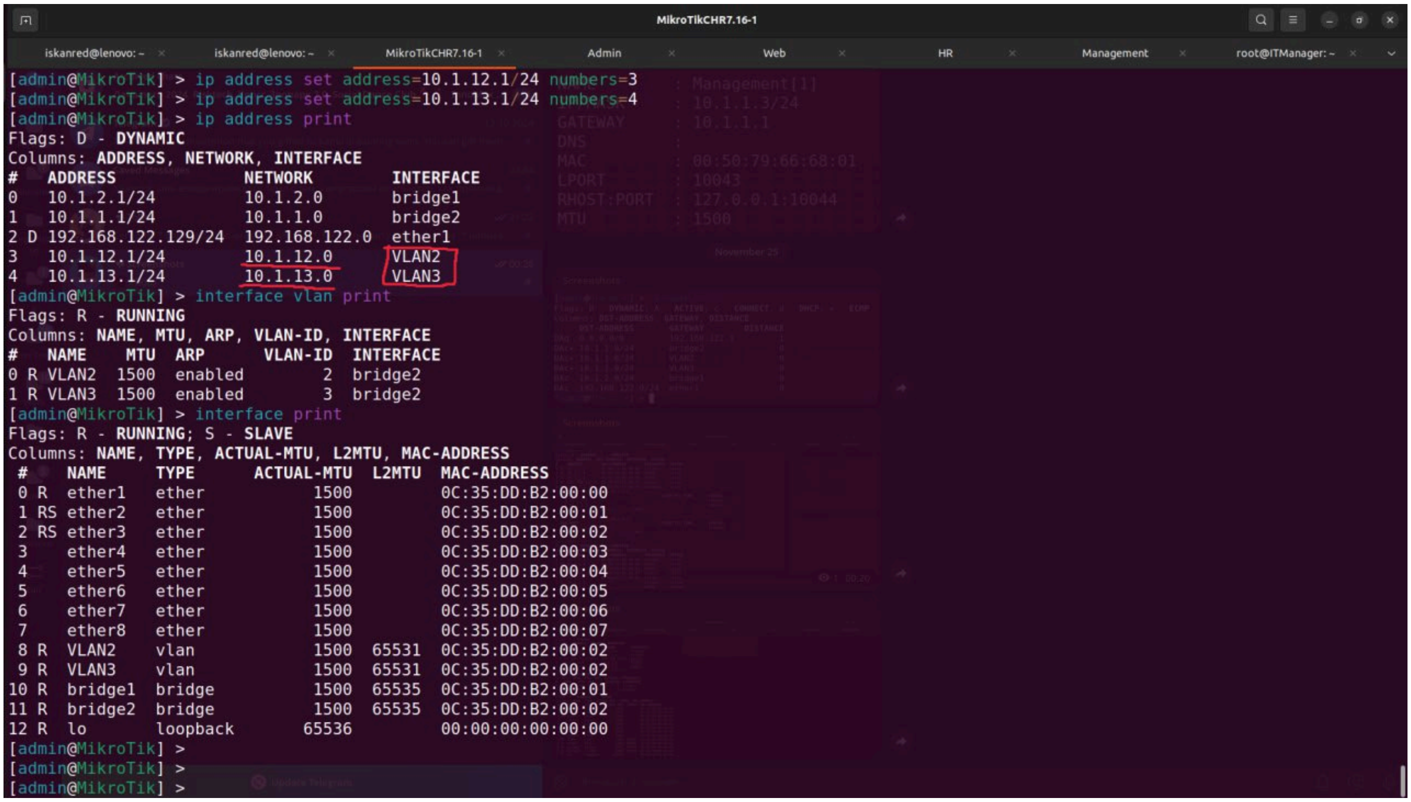
- Of course, there always been a confusion since the router using its internal algorithm chose some of the available interfaces and the choice had always been wrong 😢



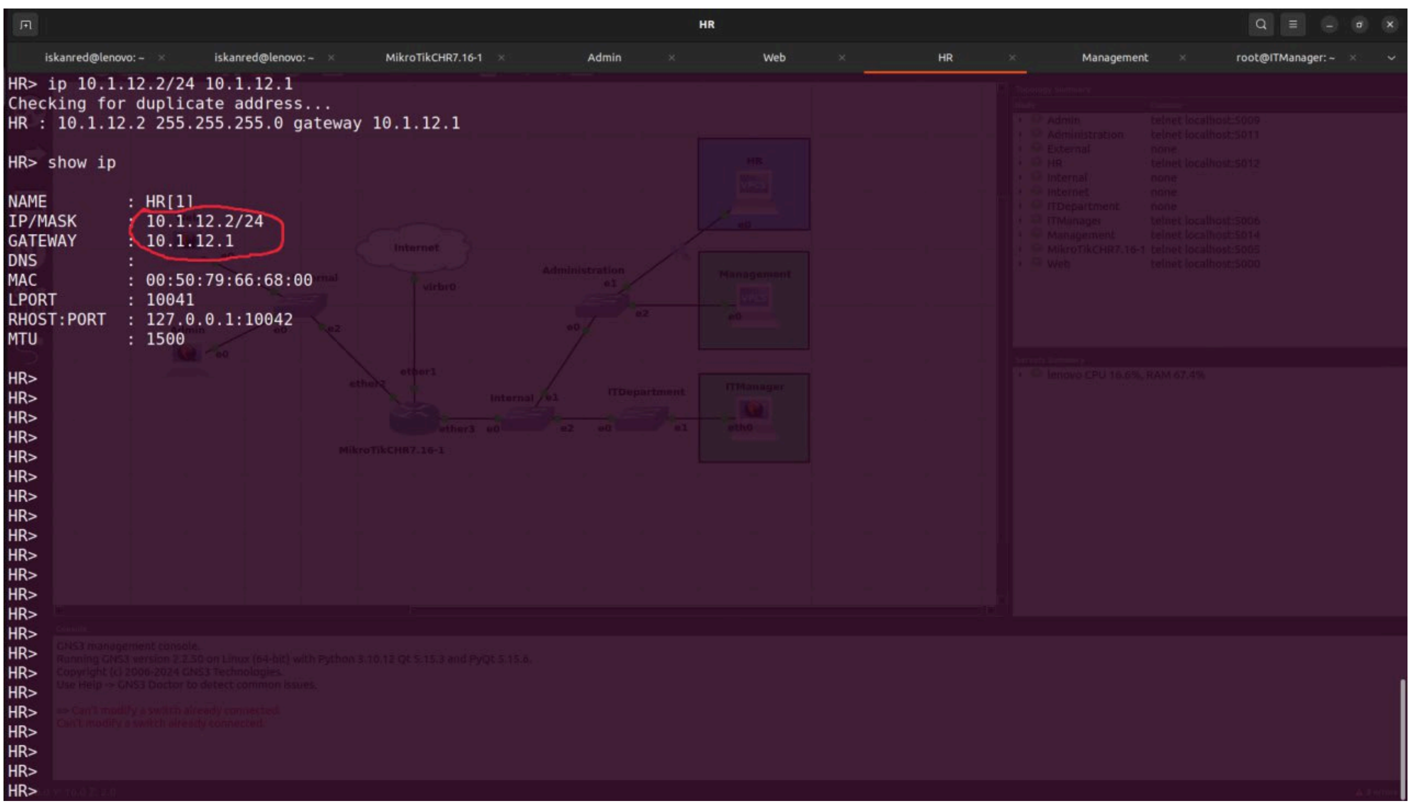
- Therefore, I decided to **change the subnets** for each of the VLANs to: `10.1.12.0/24` for `VLAN2` and `10.1.13.0/24` for `VLAN3`
  - Below is my **old** configuration



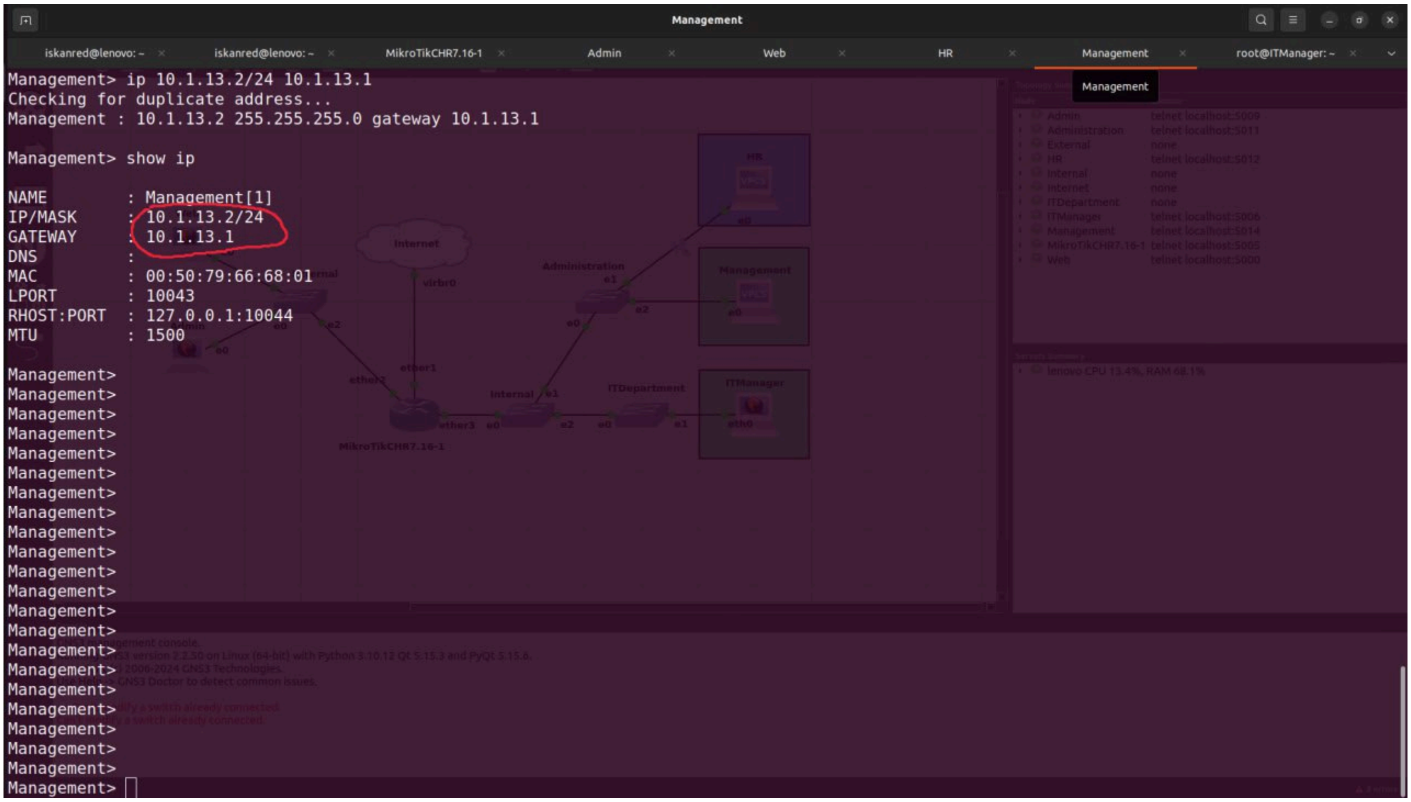
- So now, the configuration had been changed to:



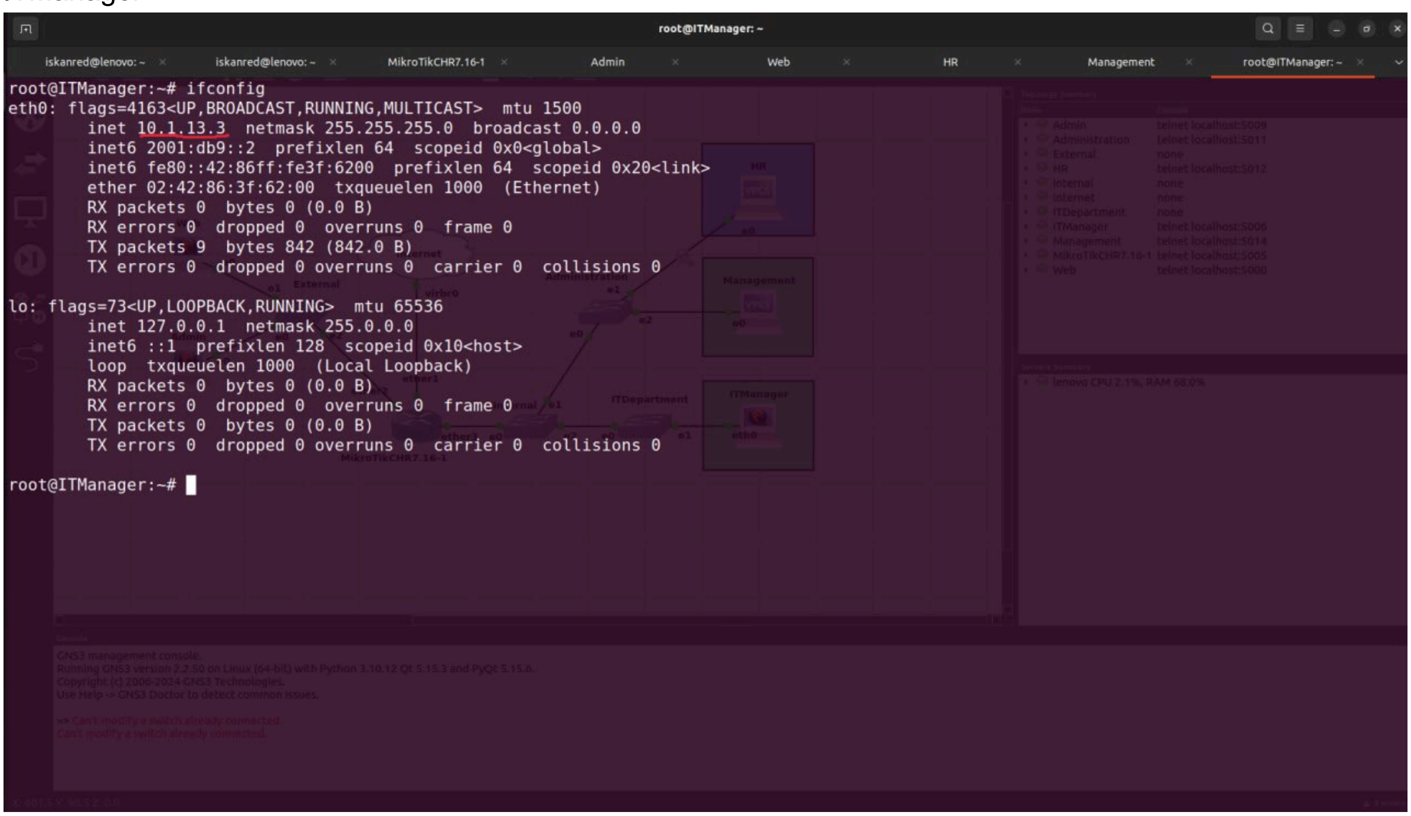
- Moreover, after that I had to **change the IP** addresses on my nodes `Management`, `HR` and `ITManager`
    - HR



- Management

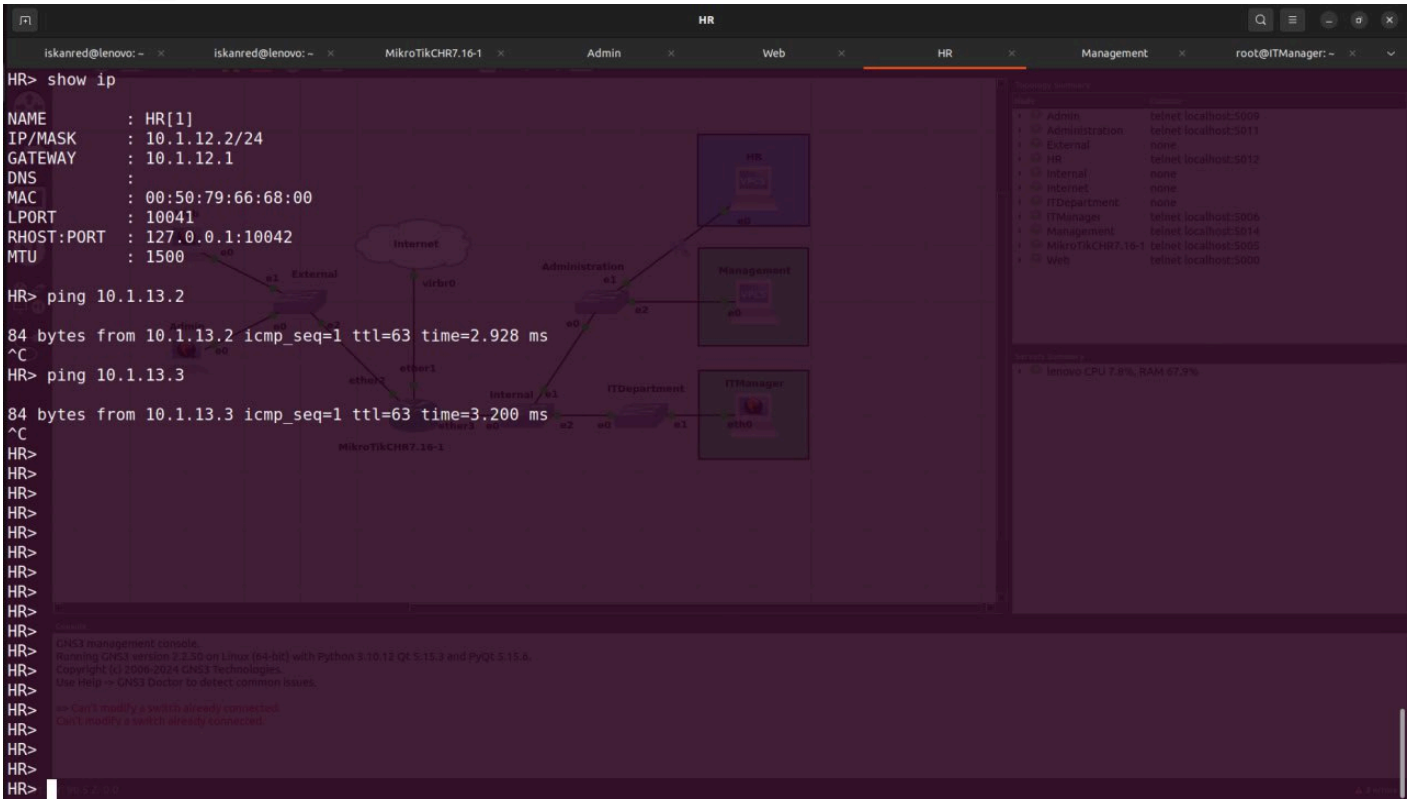


- ITManager

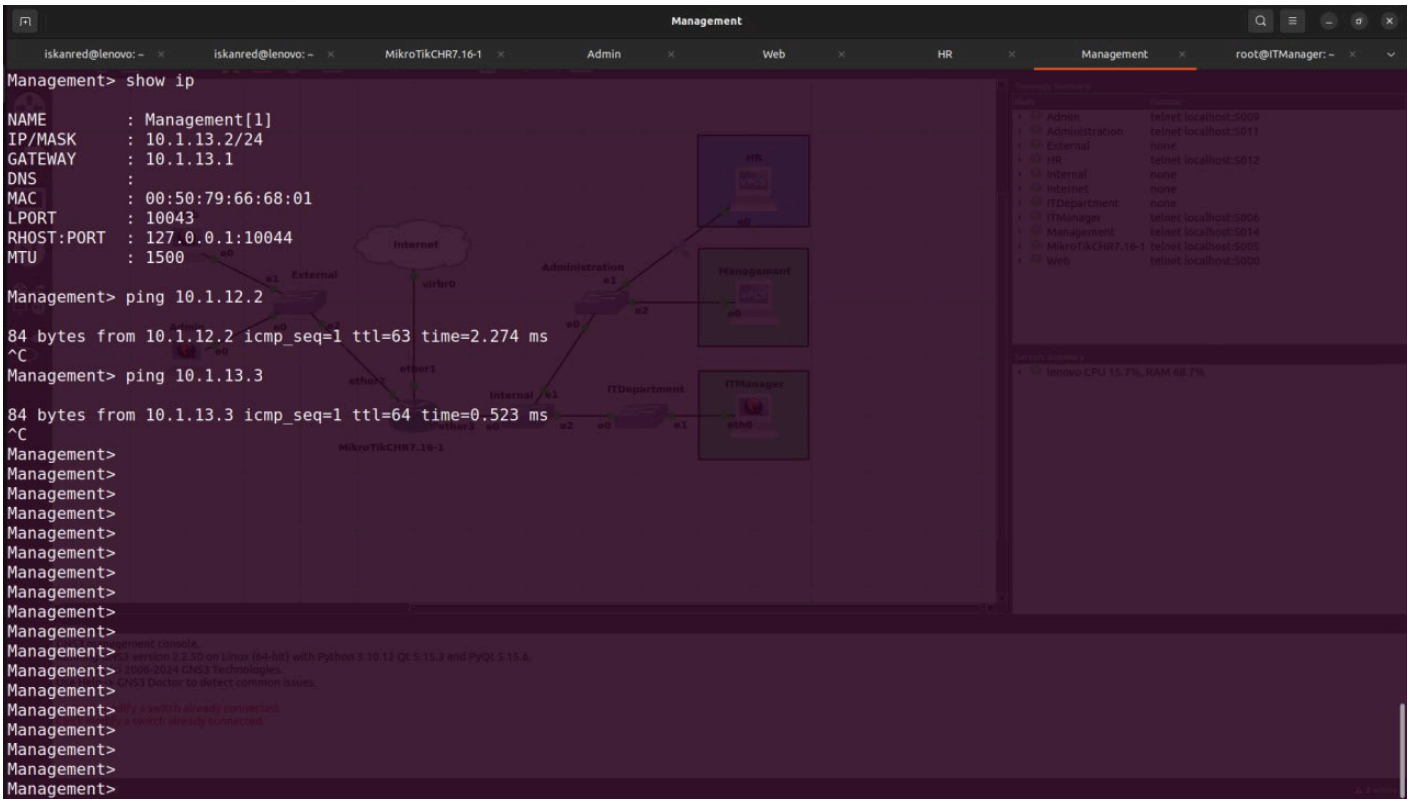


- Finally, I was able to perform ping from Management to HR and vice a versa:

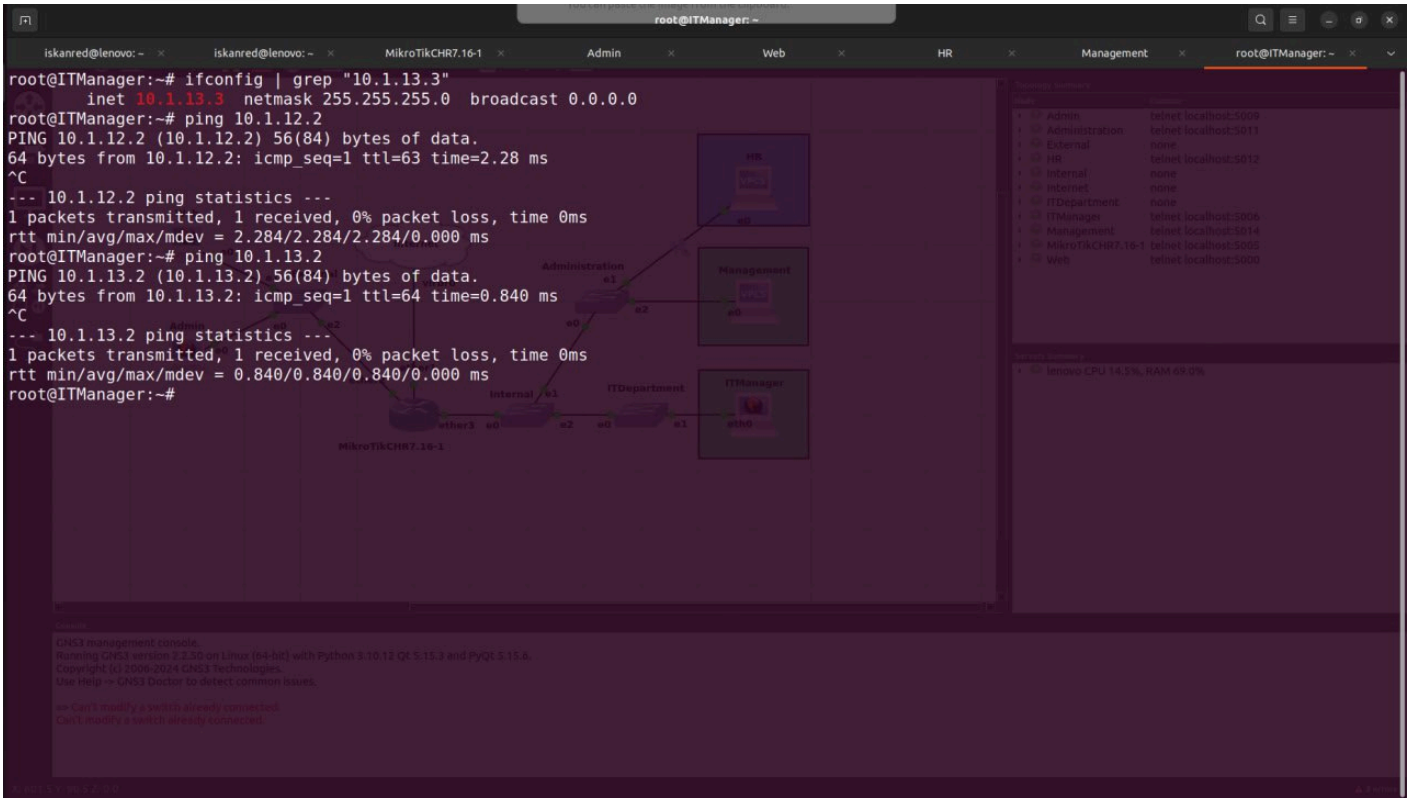
- Ping from `HR`



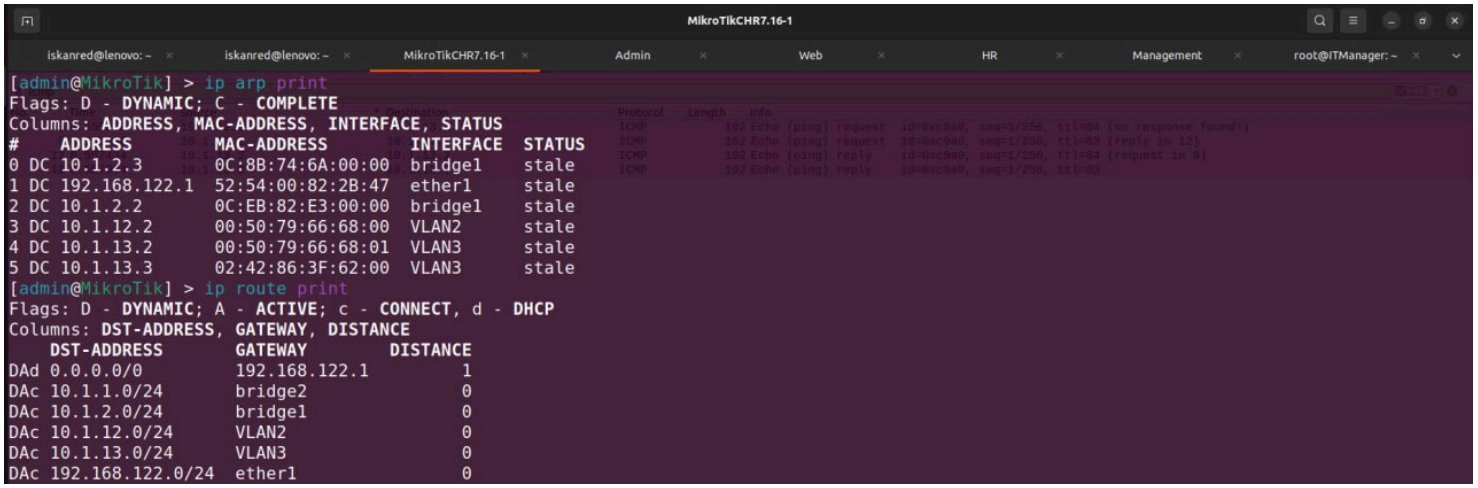
- Ping from `Management`



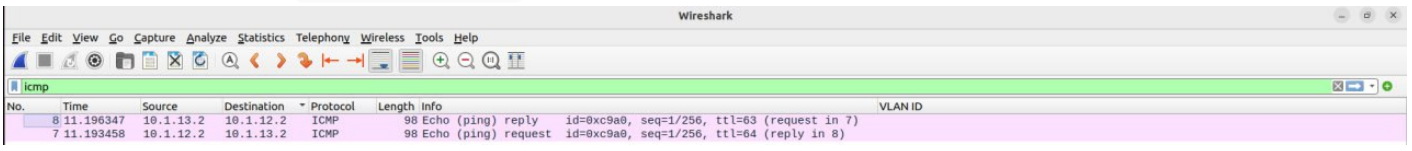
- Ping from `ITManager`



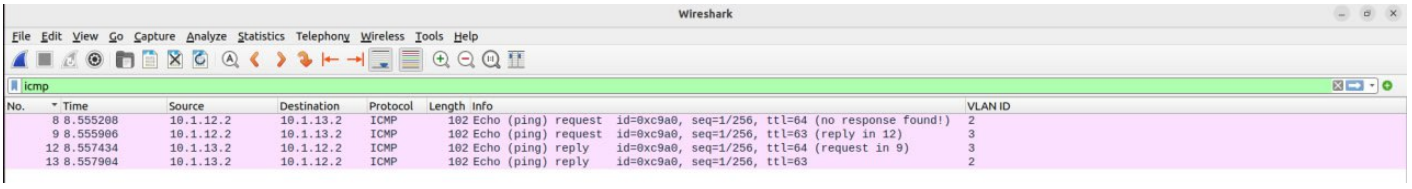
- And we can see that ARP table in the router was updated correctly



- What is interesting is that now in Wireshark we can see the ICMP request from the `HR` to `Management` twice
  - Below is capturing of the link `HR – Administration`. Here we have no VLAN ID in the header since it is added and stripped by `Administration` switch.



  - However, if we look at the link `Administration – Internal` we will notice that there are two ICMP requests and replies within the different VLANs



  - And that is an evidence that both the request and response were transferred across the `Administration` switch twice: before and after being routed into different VLAN
    - The request's path

```
HR -> Administration (VID = 2) -> Internal (VID = 2) -> Router (VID = 2) -> Router
(VID = 3) -> Internal (VID = 3) -> Administration (VID = 2) -> Management
```

- The reply's path

```
Management -> Administration (VID = 2) -> Internal (VID = 3) -> Router (VID = 3) ->
Router (VID = 2) -> Internal (VID = 2) -> Administration (VID = 2) -> HR
```

3. Fault Tolerance

> **a.** `What is Link Aggregation? How does it work (briefly)? What are the possible configuration modes?`

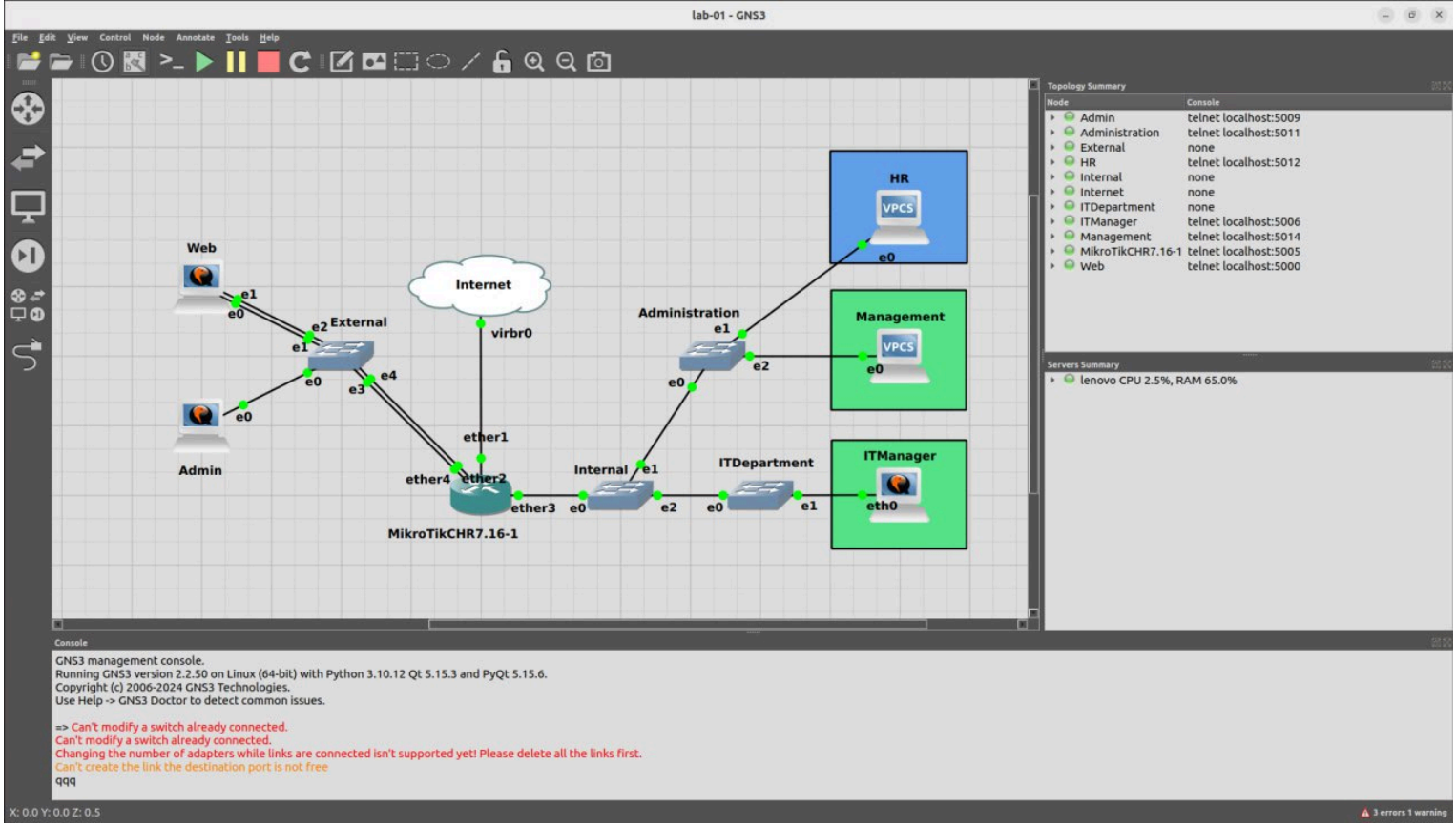
Definition
Link aggregation is a concept from **IEEE 802.1AX** (IEEE 802.3ad in the past, but no longer related **only** to Ethernet) that provides a way of using a bunch of links to increase performance of transmission and/or to ensure fault tolerance. In this context such links may exist as real links between the same two devices using different ports. On the other hand, these links could be two different devices acting as one (to achieve it clients must use Cisco VSS or Nexus VCC proprietary protocols). A group of ports combined together is called a **link aggregation group**, or LAG (bond, team, or port-channel in different implementations) with maximum of 8 links in a group. The rule that defines which packets are sent along which link is called the **scheduling algorithm**. The active monitoring protocol that allows devices to include or remove individual links from the LAG is called **Link Aggregation Control Protocol** (LACP).

How it works?
LACP works by sending frames (LACPDUs) down all links that have the protocol enabled. If it finds a device on the other end of a link that also has LACP enabled, that device will independently send frames along the same links in the opposite direction enabling the two units to detect multiple links between themselves and then combine them into a single logical link. LACP can be configured in one of two modes: active or passive. In active mode, LACPDUs are sent 1 per second along the configured links. In passive mode, LACPDUs are not sent until one is received from the other side, a speak-when-spoken-to protocol.

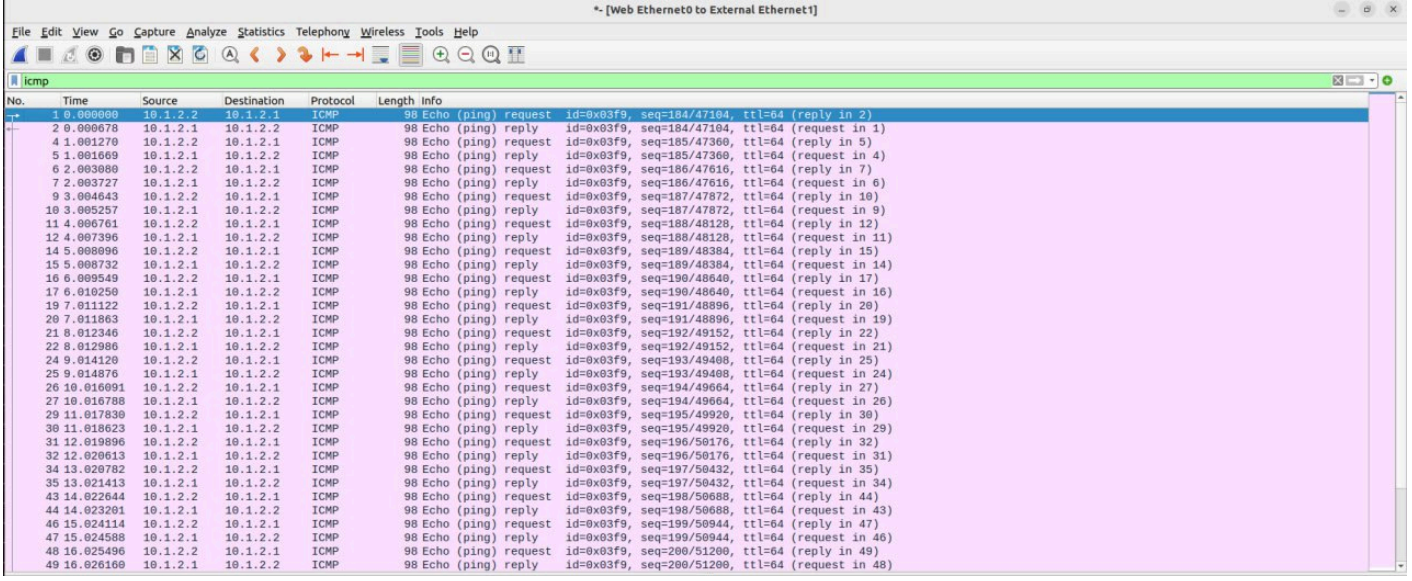
Configuration modes
The configuration modes may include:

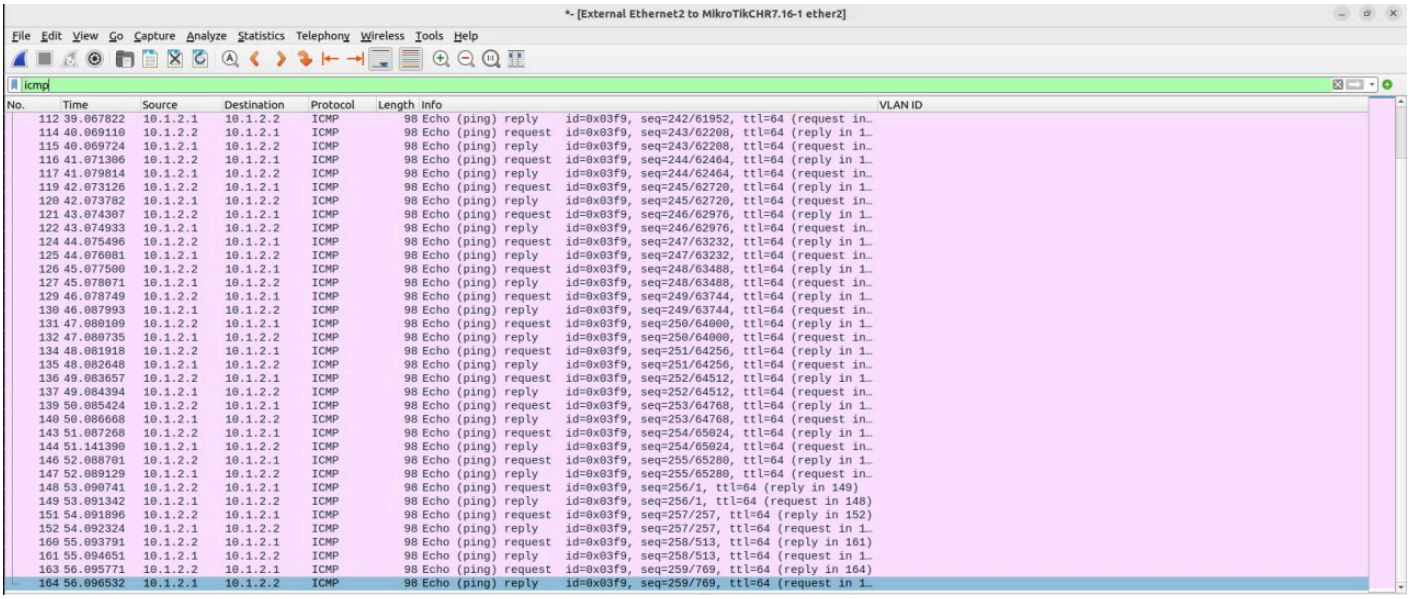
- **Scheduling** (load-balancing) algorithm. ***Example***: MAC-address hashing (L2), IP-address hashing (L3), socket hashing (L4), round-robin, other. It must be selected according utilisation.
- **Active** or **passive** (or merely on), equivalently, to use LACP or not to use (e.g. to use PAgP in Cisco). The **active** option means the device will actively monitor the state of the link and automatically remove any failed links from the bundle. However, the other device may not support LACP, so it can lead to the situation when one device will keep dumping packets down a link that the other device isn't watching.

> **b.** `Use link aggregation between the Web and the Gateway to have Load Balancing and Fault Tolerance as follows.`

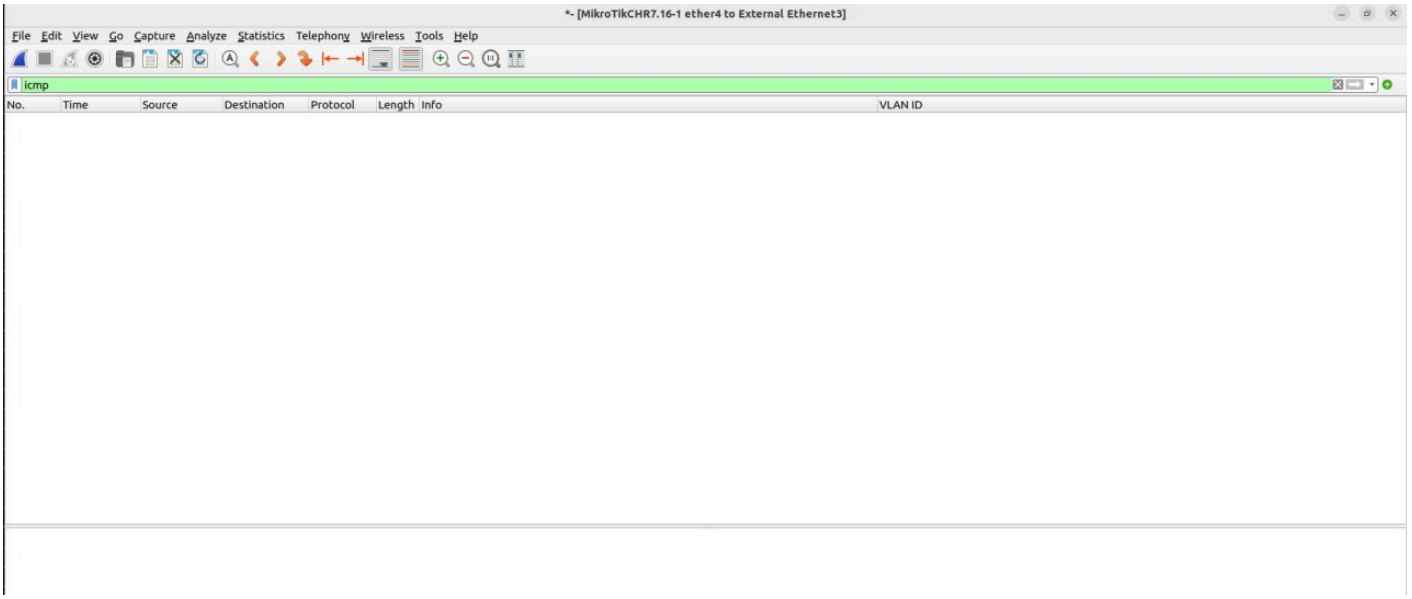


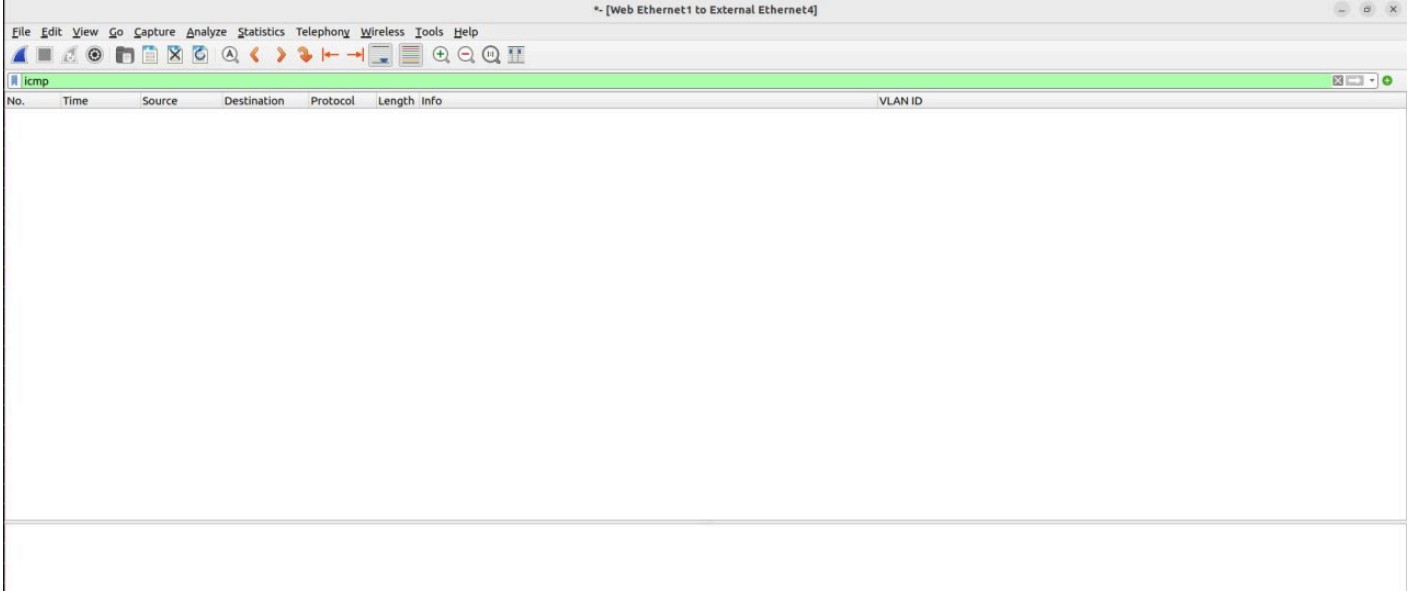
- If we put just one more link for each of the parts of the path from the Web to the Gateway we will get such a picture:
  - All the traffic goes through one of the links constantly for both `Web – External (switch)` and `External (switch) – Mikrotik (router)`





- While other links do not see any traffic





- So, let's firstly put a new `bonding` interface to our router which will combine `ether2` and `ether4` as slaves.
  - Let's have a look at the configuration of the router (`10.1.2.1/24`) with added `bonding` interface:

```
[admin@MikroTik] > interface bond print
Flags: X - disabled; R - running
 0  R name="bond2" mtu=1500 mac-address=0C:35:DD:B2:00:01 arp=enabled arp-timeout=auto slaves=ether2,ether4 mode=balance-rr primary=none
      link-monitoring=mii arp-interval=100ms arp-ip-targets="" mii-interval=100ms down-delay=0ms up-delay=0ms lacp-rate=30secs
      transmit-hash-policy=layer-2 min-links=0
[admin@MikroTik] > interface print
Flags: R - RUNNING; S - SLAVE
Columns: NAME, TYPE, ACTUAL-MTU, L2MTU, MAC-ADDRESS
 #    NAME     TYPE      ACTUAL-MTU  L2MTU  MAC-ADDRESS
 0 R  ether1   ether            1500         0C:35:DD:B2:00:00
 1 RS ether2   ether            1500         0C:35:DD:B2:00:01
 2 RS ether3   ether            1500         0C:35:DD:B2:00:02
 3 RS ether4   ether            1500         0C:35:DD:B2:00:03
 4    ether5   ether            1500         0C:35:DD:B2:00:04
 5    ether6   ether            1500         0C:35:DD:B2:00:05
 6    ether7   ether            1500         0C:35:DD:B2:00:06
 7    ether8   ether            1500         0C:35:DD:B2:00:07
 8 R  VLAN2    vlan             1500  65531  0C:35:DD:B2:00:02
 9 R  VLAN3    vlan             1500  65531  0C:35:DD:B2:00:02
10 R  bond2    bond             1500   1500  0C:35:DD:B2:00:01
11 R  bridge3  bridge           1500  65535  0C:35:DD:B2:00:02
12 R  lo       loopback        65536         00:00:00:00:00:00
[admin@MikroTik] > ip address print
Flags: D - DYNAMIC
Columns: ADDRESS, NETWORK, INTERFACE
#   ADDRESS            NETWORK         INTERFACE
0   10.1.13.1/24       10.1.13.0       VLAN3
1 D 192.168.122.129/24 192.168.122.0   ether1
2   10.1.12.1/24       10.1.12.0       VLAN2
3   10.1.2.1/24        10.1.2.0        bond2
[admin@MikroTik] > ip route print
Flags: D - DYNAMIC; A - ACTIVE; c - CONNECT, d - DHCP
Columns: DST-ADDRESS, GATEWAY, DISTANCE
    DST-ADDRESS       GATEWAY       DISTANCE
DAd 0.0.0.0/0         192.168.122.1        1
DAc 10.1.2.0/24       bond2                0
DAc 10.1.12.0/24      VLAN2                0
DAc 10.1.13.0/24      VLAN3                0
DAc 192.168.122.0/24  ether1               0
```

  - The default algorithm of load balancing for a `bonding` interface in Mikrotik is `balance-rr` which is a simple Round Robin algorithm.

```
[admin@MikroTik] > interface bonding print
Flags: X - disabled; R - running
 0  R name="bond2" mtu=1500 mac-address=0C:35:DD:B2:00:01 arp=enabled arp-timeout=auto slaves=ether2,ether4 mode=balance-rr primary=none
      link-monitoring=mii arp-interval=100ms arp-ip-targets="" mii-interval=100ms down-delay=0ms up-delay=0ms lacp-rate=30secs
      transmit-hash-policy=layer-2 min-links=0
[admin@MikroTik] > interface bonding monitor 0
          mode: balance-rr
    active-ports: ether2,ether4
  inactive-ports:

[admin@MikroTik] > interface bonding monitor-slaves 0
input does not match any value of bond
[admin@MikroTik] > interface bonding monitor-slaves bond2
Flags: A - active; P - partner
 A  port=ether2

 A  port=ether4
```

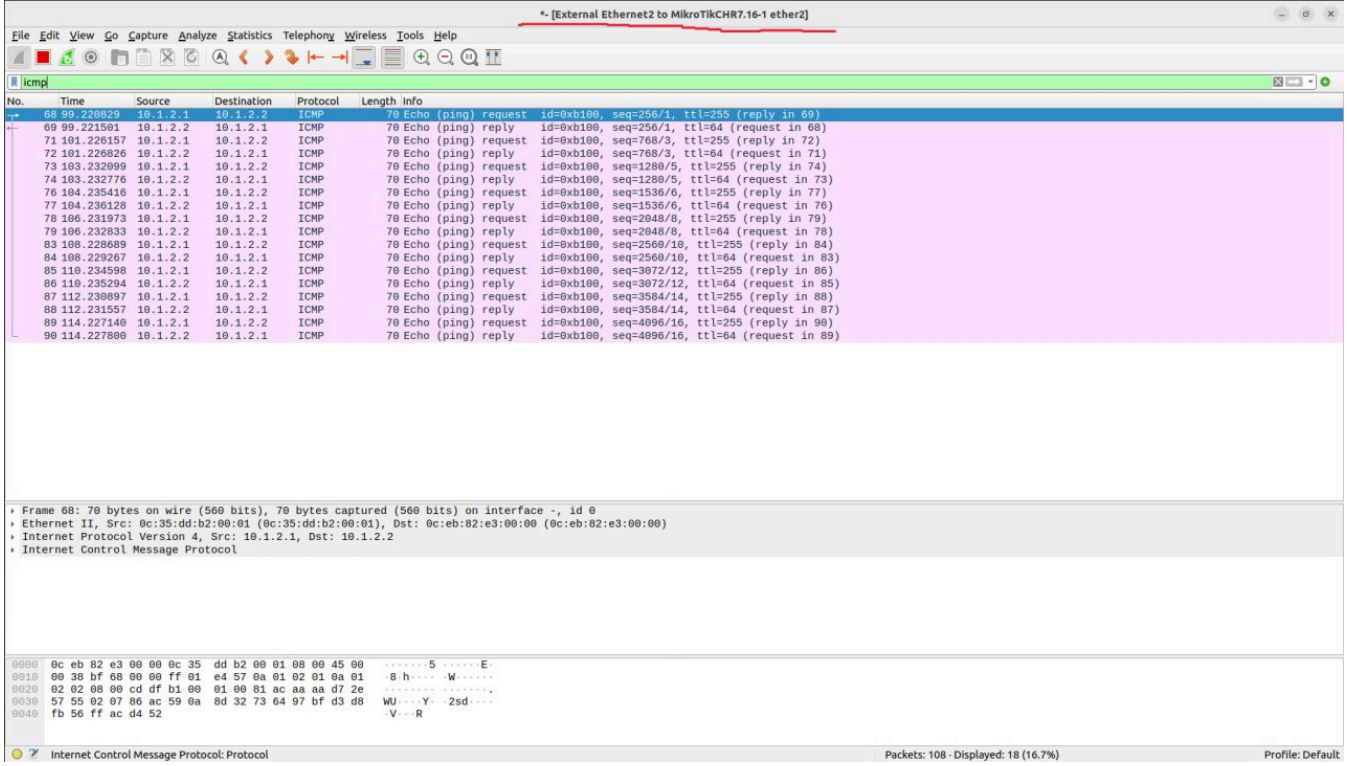
  - Also, let's take a look at the configuration of the `Web` node (`10.1.2.2/24`). For now it's important just to check it, the reasons will be considered after.

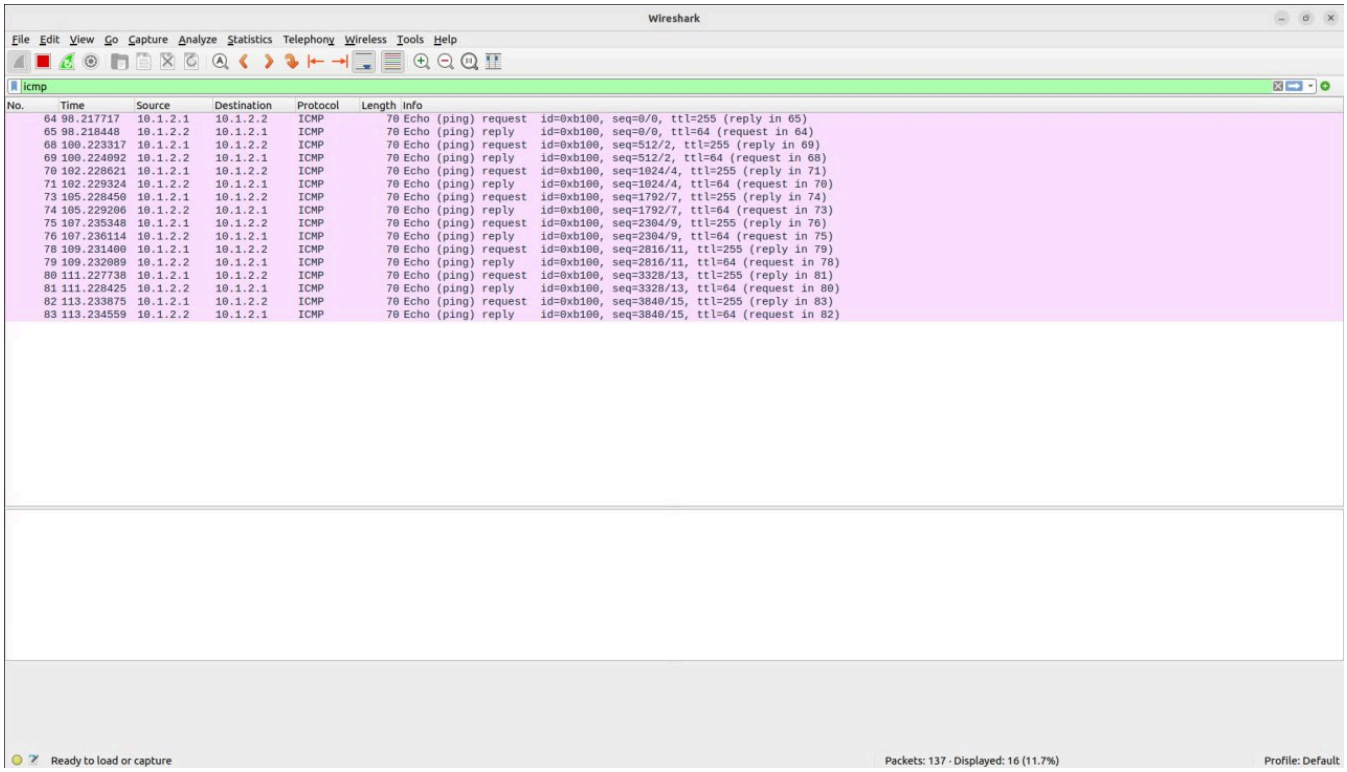
```
ubuntu@ubuntu-cloud:~$ ifconfig | grep 10.1
        inet 10.1.2.2  netmask 255.255.255.0  broadcast 10.1.2.255
        inet 10.1.2.2  netmask 255.255.255.0  broadcast 10.1.2.255
ubuntu@ubuntu-cloud:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2
  ethernets:
    e0:
      match:
        macaddress: "0c:eb:82:e3:00:00"
      dhcp4: no
      dhcp6: no
      addresses:
        - 10.1.2.2/24
        - "2001:db8::b/64"
      routes:
        - to: default
          via: 10.1.2.1
        - to: "::/0"
          via: "2001:db8::1"
          on-link: yes
      set-name: "e0"
      nameservers:
        addresses:
          - 1.1.1.1
          - 8.8.8.8
    e1:
      match:
        macaddress: "0c:eb:82:e3:00:01"
      dhcp4: no
      dhcp6: no
      addresses:
        - 10.1.2.2/24
        - "2001:db8::b/64"
      routes:
        - to: default
          via: 10.1.2.1
        - to: "::/0"
          via: "2001:db8::1"
          on-link: yes
      set-name: "e1"
      nameservers:
        addresses:
```

- Let's check if it works
  - Let's perform ping from the `Router` to the `Web`

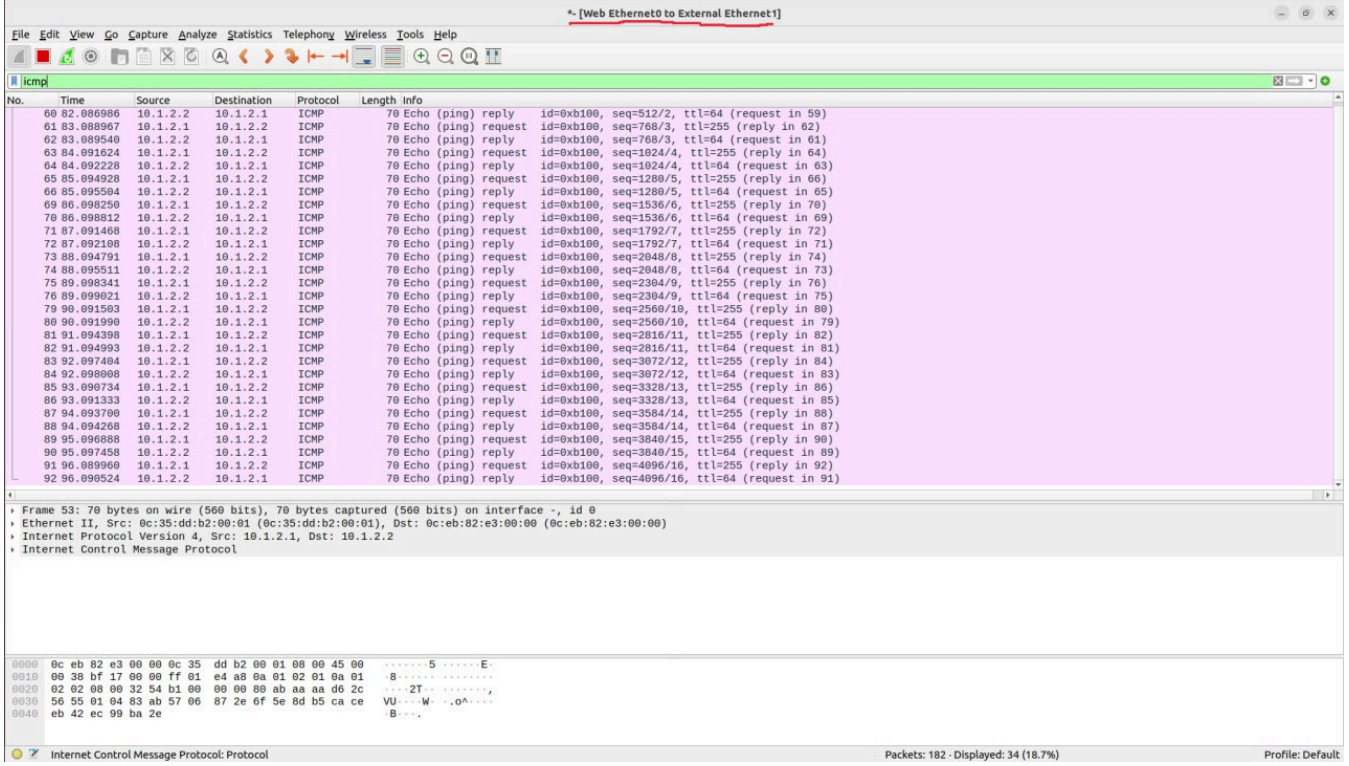
```
[admin@MikroTik] > interface bond print
Flags: X - disabled; R - running
 0  R name="bond2" mtu=1500 mac-address=0C:35:DD:B2:00:01 arp=enabled arp-timeout=auto slaves=ether2,ether4 mode=balance-rr primary=none
     link-monitoring=mii arp-interval=100ms arp-ip-targets="" mii-interval=100ms down-delay=0ms up-delay=0ms lacp-rate=30secs
     transmit-hash-policy=layer-2 min-links=0
[admin@MikroTik] > ip address print
Flags: D - DYNAMIC
Columns: ADDRESS, NETWORK, INTERFACE
#   ADDRESS            NETWORK        INTERFACE
0   10.1.13.1/24       10.1.13.0      VLAN3
1 D 192.168.122.129/24 192.168.122.0  ether1
2   10.1.12.1/24       10.1.12.0      VLAN2
3   10.1.2.1/24        10.1.2.0       bond2
[admin@MikroTik] > ping 10.1.2.2
  SEQ HOST                                     SIZE TTL TIME       STATUS
    0 10.1.2.2                                   56  64 1ms607us
    1 10.1.2.2                                   56  64 1ms405us
    2 10.1.2.2                                   56  64 1ms608us
    3 10.1.2.2                                   56  64 1ms529us
    4 10.1.2.2                                   56  64 1ms511us
    5 10.1.2.2                                   56  64 1ms555us
    6 10.1.2.2                                   56  64 1ms561us
    7 10.1.2.2                                   56  64 1ms566us
    8 10.1.2.2                                   56  64 1ms761us
    9 10.1.2.2                                   56  64 1ms500us
   10 10.1.2.2                                   56  64 1ms282us
   11 10.1.2.2                                   56  64 1ms348us
   12 10.1.2.2                                   56  64 1ms593us
   13 10.1.2.2                                   56  64 1ms362us
   14 10.1.2.2                                   56  64 1ms351us
   15 10.1.2.2                                   56  64 1ms439us
   16 10.1.2.2                                   56  64 1ms395us
    sent=17 received=17 packet-loss=0% min-rtt=1ms282us avg-rtt=1ms492us max-rtt=1ms761us

[admin@MikroTik] >
```

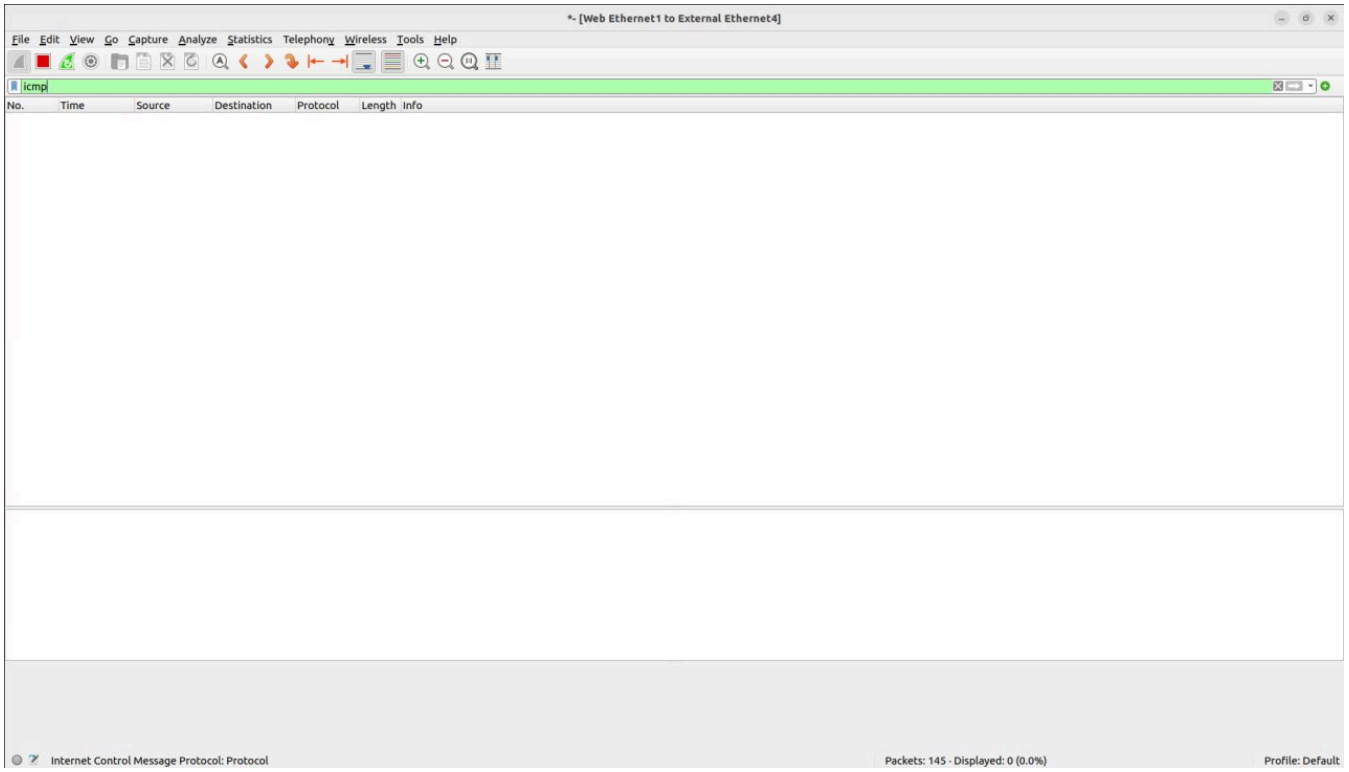
  - We can notice that all ICMP traffic was equally divided between two links that are located between the nodes `Router` - `Switch`.





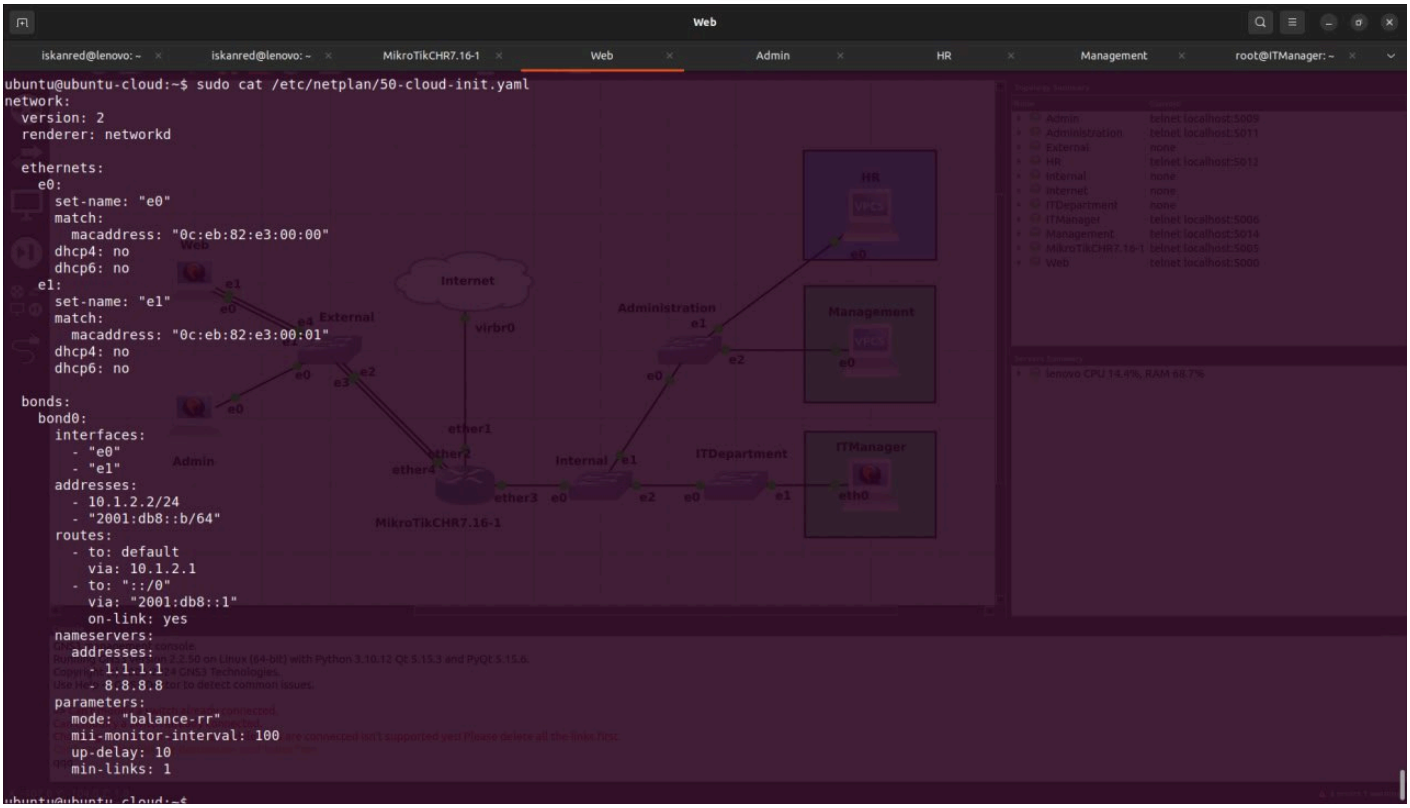
- However, the distribution between the links `Switch` - `Web` remained the same (only one link transmitted all the frames).



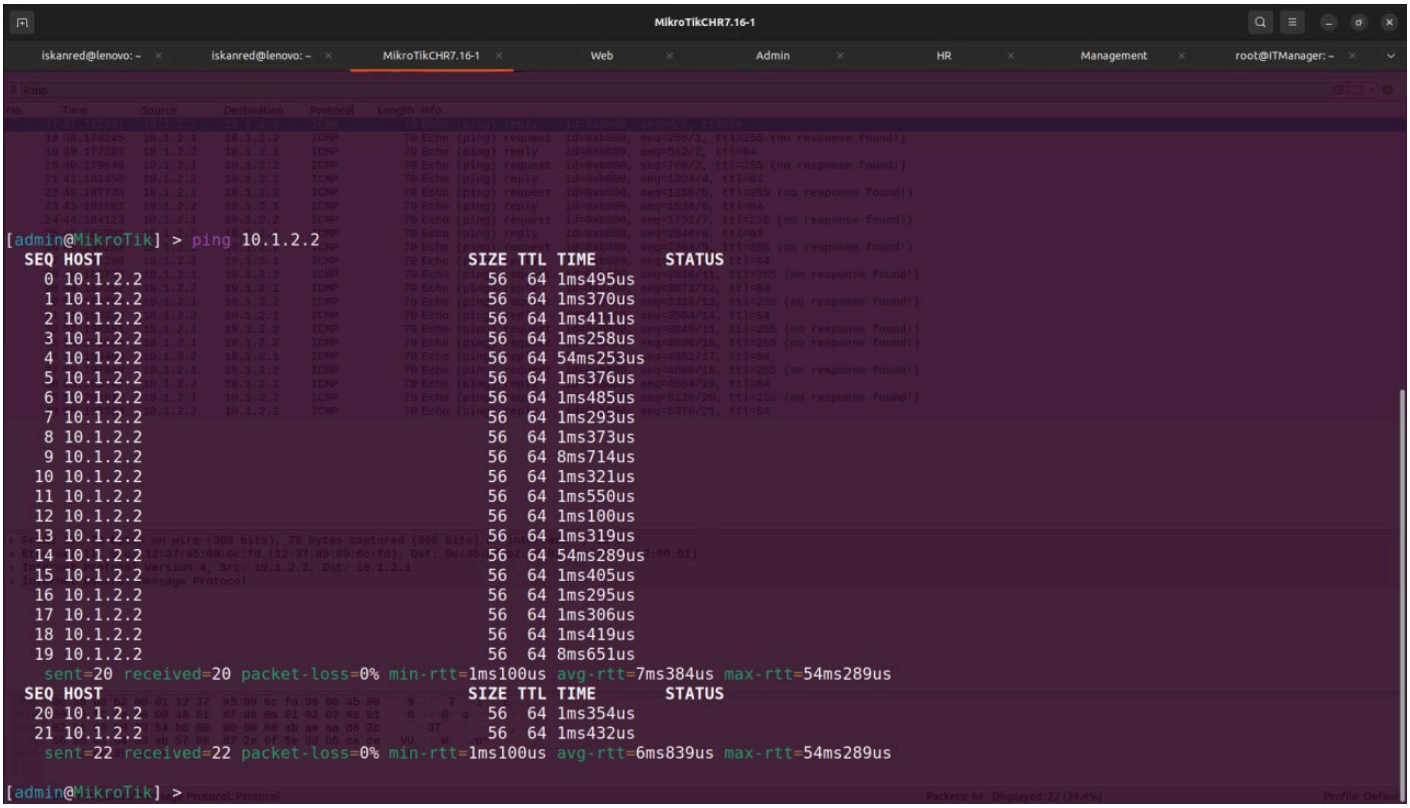


  - This happened because I didn't configured anything on the `Switch` side or `Web` side, so we had an impact only on links that are connect to the router directly. Since my switch is really limited in its configuration (because of version of GNS3 used), let's configure only `Web` side.
- So, to configure the `Web` side for link aggregation let's use `netplan`

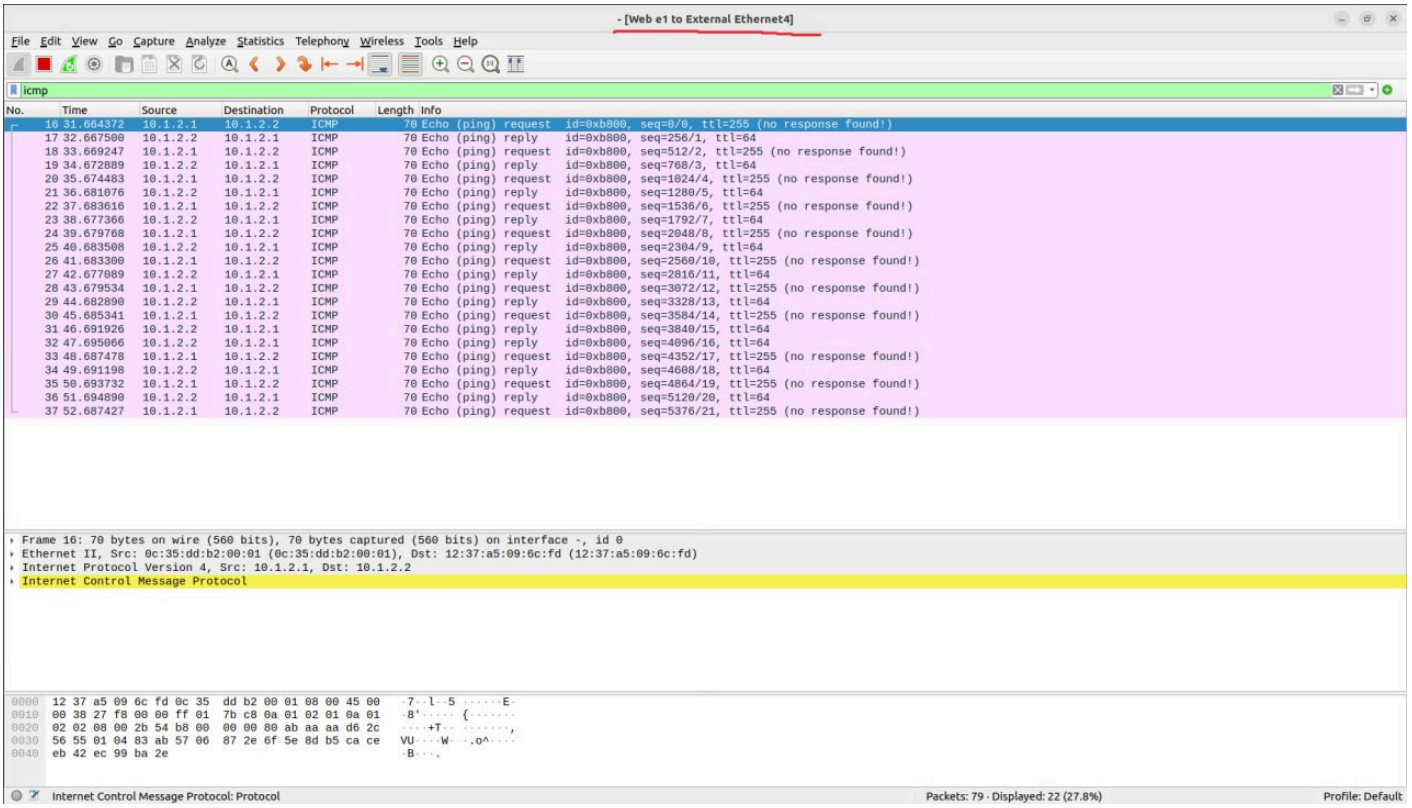
- First let's configure bonding interface on the `Web` using `netplan`

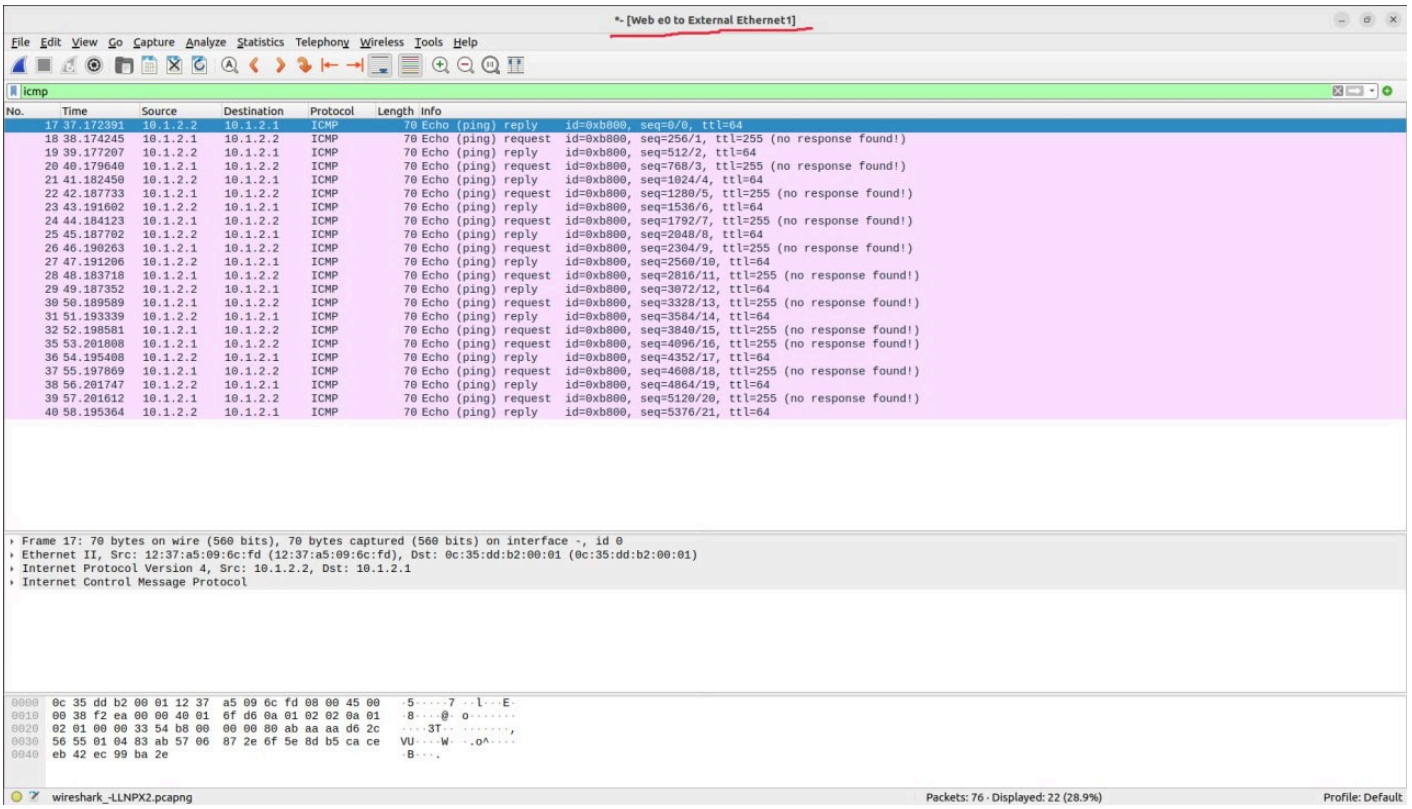


- So, let's again perform ping from the `Router` to the `Web`



- Now it's noticeable that both links transmitted ICMP traffic



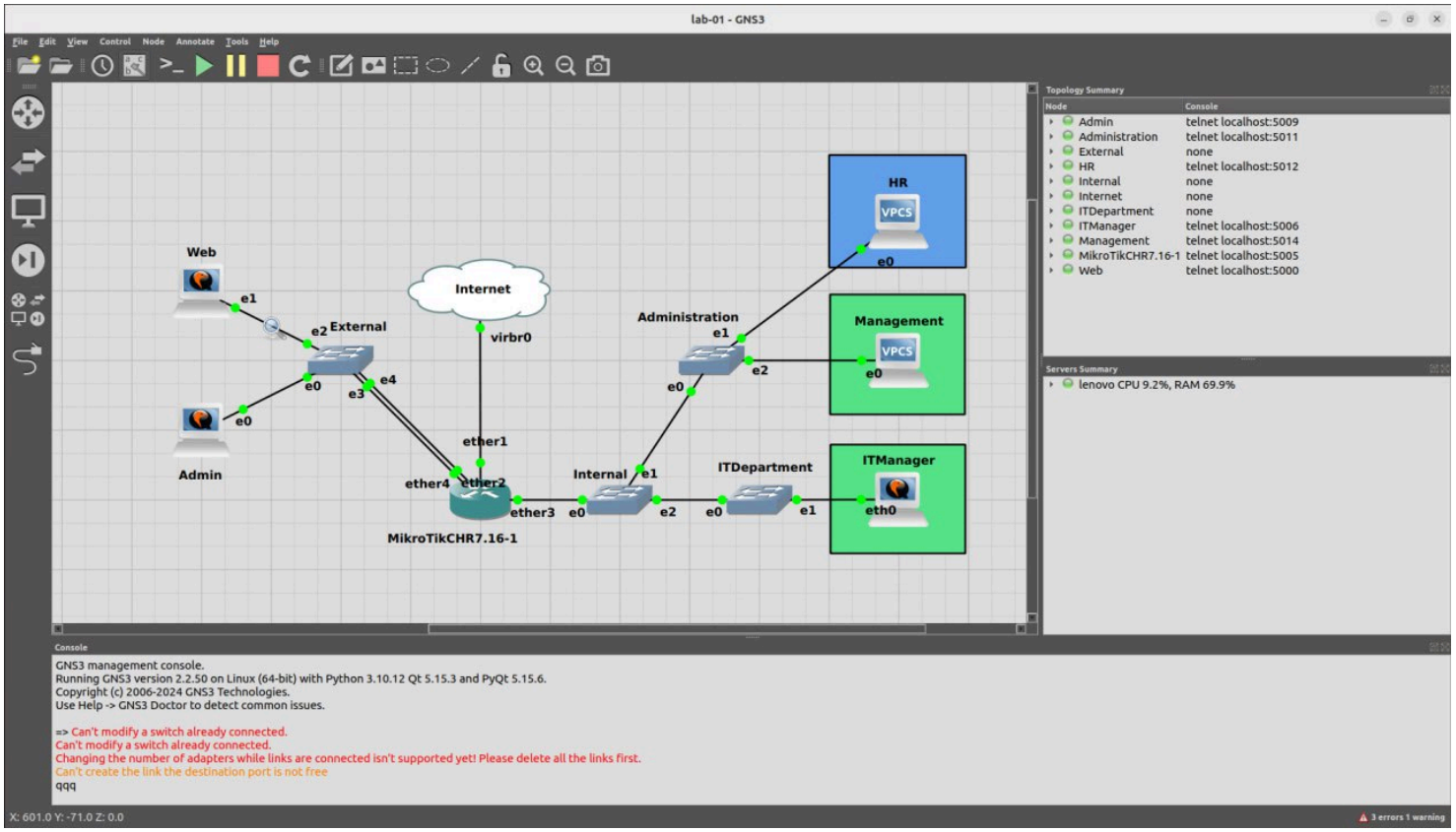


- What is interesting here, is that all ICMP requests always go through the same link. Not surprisingly, all replies also always go through the other link.

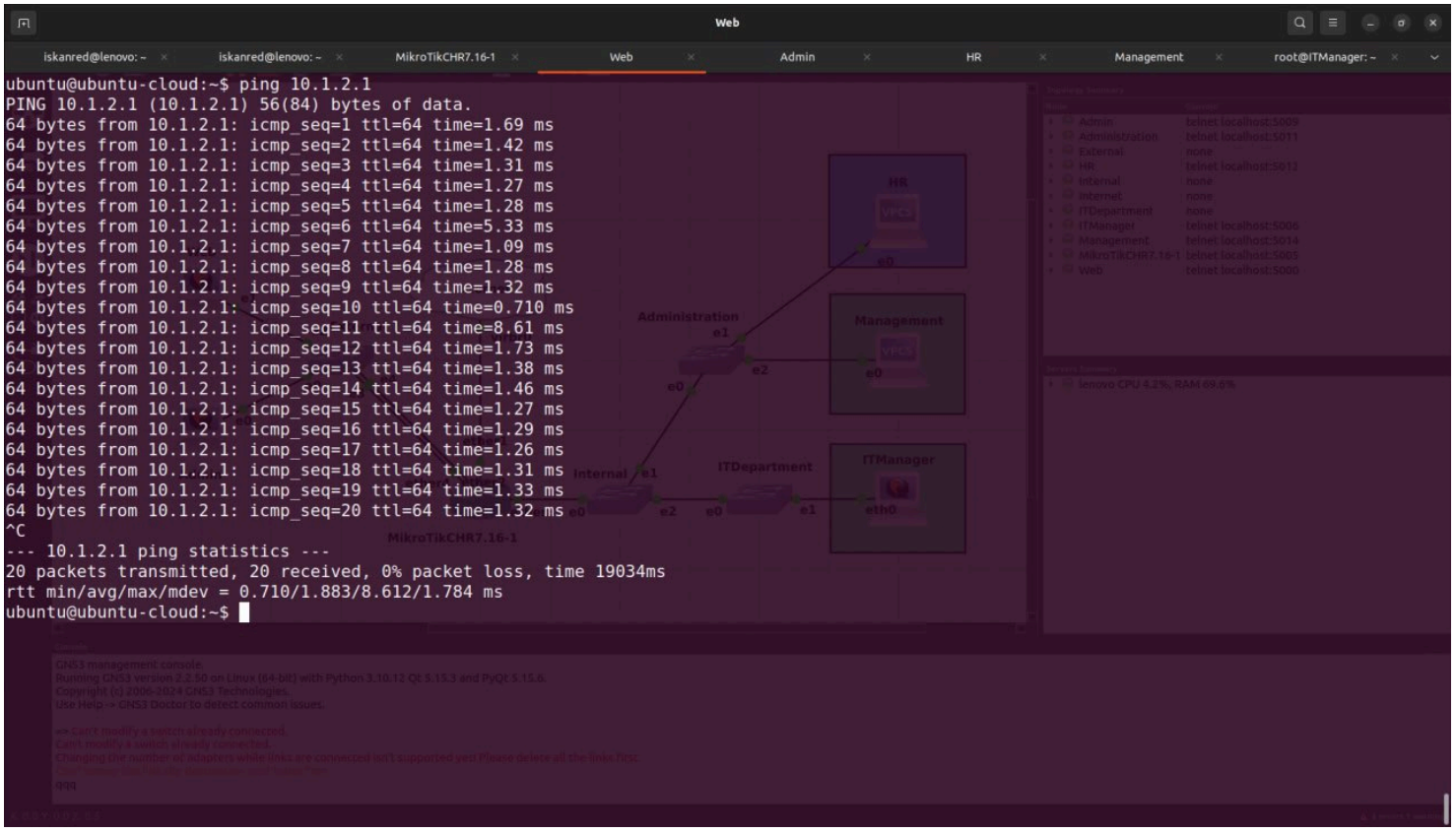
> **c.** `Test the Fault Tolerance by stopping one of the cables and see if you have any downtime`

- I decided to remove the link from the `Web` to `External`. But before it I started pinging from the `Web` to the `Router`

- Then, I removed the link

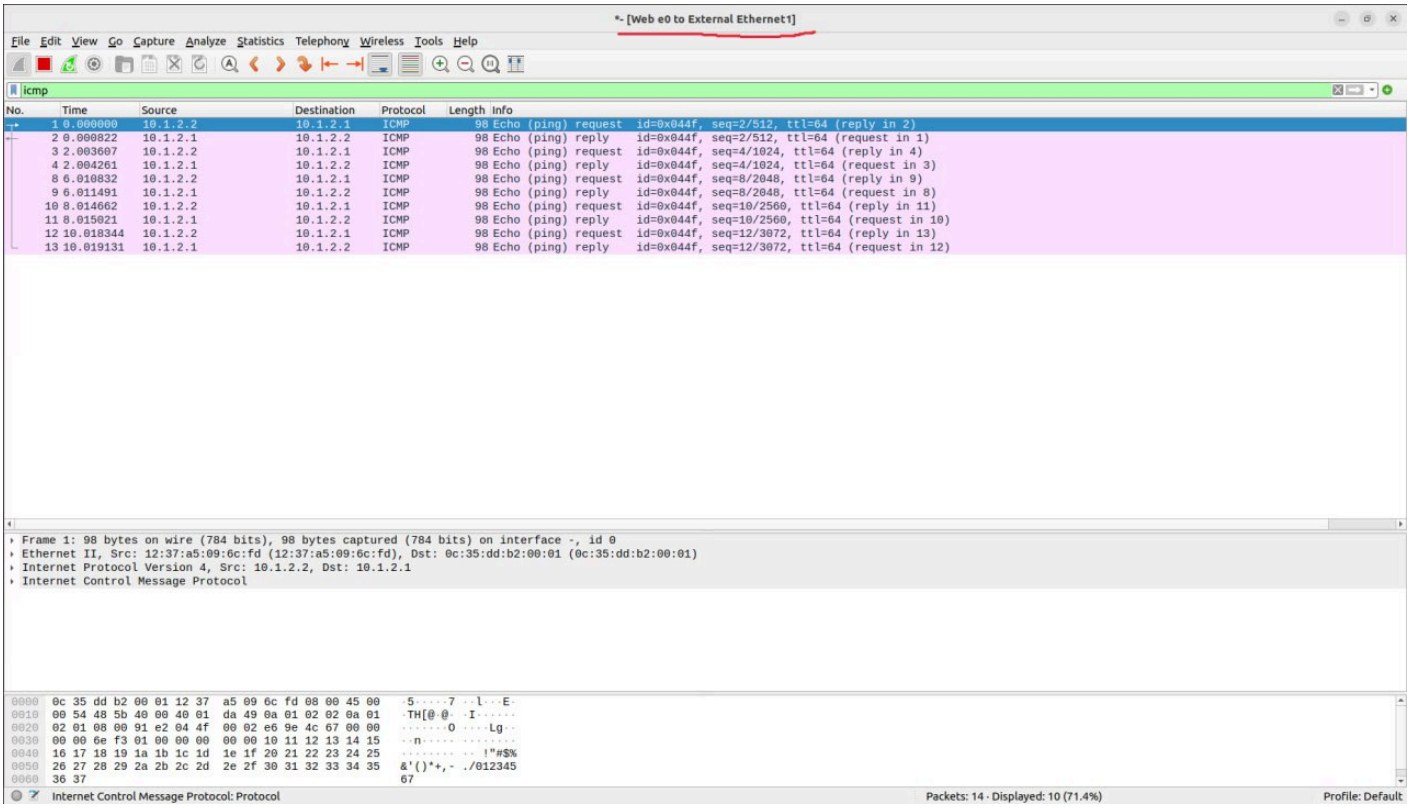


- Looking at ping further we can see there was no downtime at all! 0% packet loss is the proof

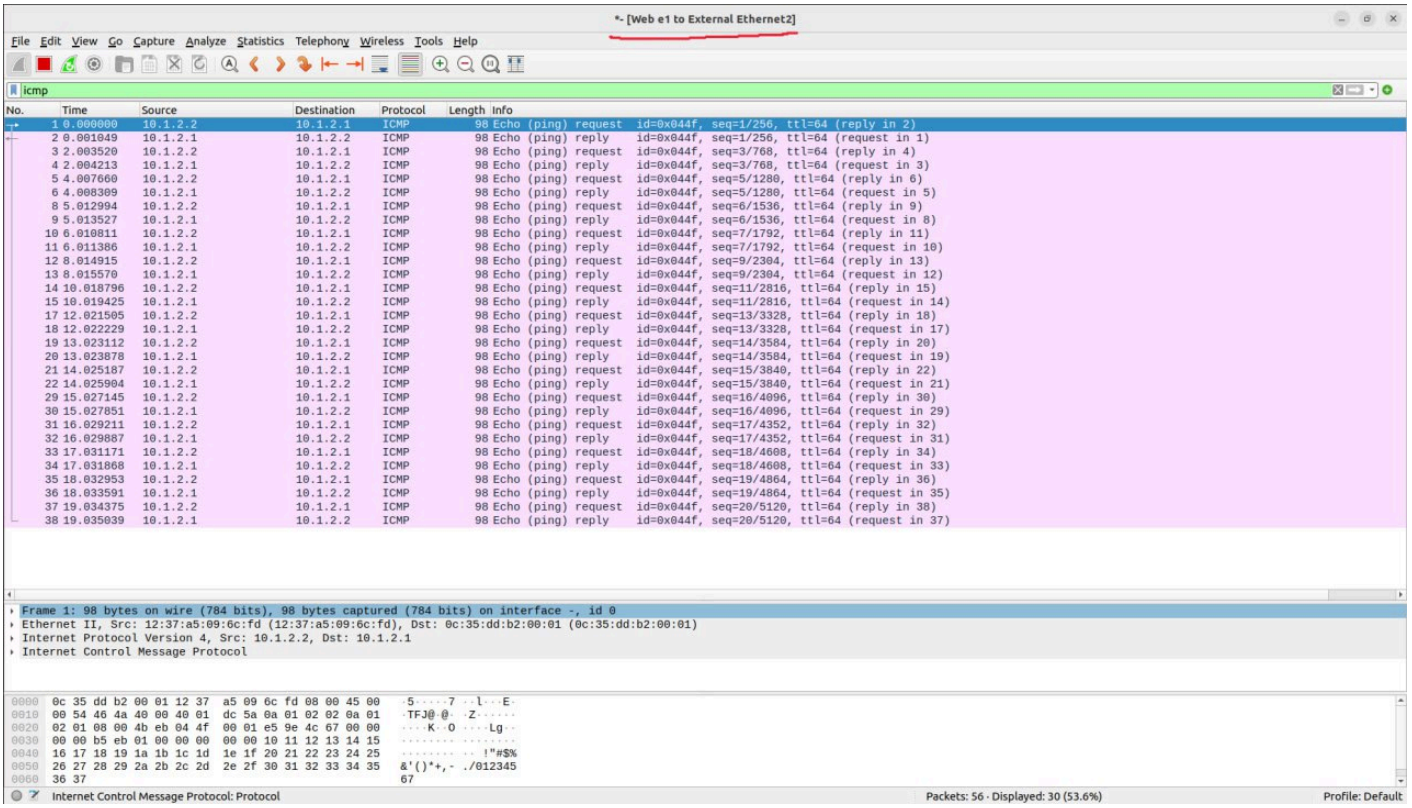


- Let's check Wireshark capturing frames

- Here we can see the link that was removed



- Here is the link that was NOT removed



- Nothing was duplicated and there was no downtime. Probably, such a fast forwarding was caused of using failure detection algorithm that is called **media-independent interface** (MII) which is fast and it depends on the device driver.

References

- https://en.wikipedia.org/wiki/IEEE_802.1Q
- https://en.wikipedia.org/wiki/VLAN
- http://xgu.ru/wiki/Native_VLAN
- https://www.smart-soft.ru/blog/tehnologija_vlan/

- https://www.auvik.com/franklyit/blog/network-basics-link-aggregation/