Lab 02: Network Segmentation part 1 - Subnets

Network Infrastructure Security (CSP)  
 2025-2026

© Henk Brouckxon, Nico Declerck, Chris Roets

# Introduction

## Lab concept

In the previous lab, we started configuring the CBROPS network. However, it should be clear that this network is far from ideal with regard to security. During this lab, we will start improving upon its originally flat (and insecure) layout. As a first step, we’ll segment the network into logical subnets and check what impact these have on network communications and visibility. Also, we’ll look closer into the “network.cbrops.net” device and its security impact.

## Learning goals

* Monitoring network traffic with wireshark
* Network segmentation using subnets + its impact on security
* Identifying layer 2 networking devices (switches etc.)
* Network monitoring and SPAN ports

## Practicalities and prerequisites

You will need the following:

* A laptop/desktop
* Access to the CBROPS lab environment

A diagram of a computer system

Description automatically generated

Figure 1 Equivalent physical layout for the CBROPS network

A computer icons on a black background

Description automatically generated

Figure 2 Initial logical layout for the CBROPS network

# Baseline – A Flat Network

For this part of the lab, the following VMs should be started[[1]](#footnote-2):

* **firewall**
* **network**
* **workstation01**
* **workstation02**
* **server01**
* **adminstation**

The **switch**, **server02** and **adminserver** VMs should not be started yet.

1. Log in to **Workstation-01** with your personal account and perform the following:
2. Ping to workstation-02, server-01 and adminstation. If this is not possible, try to solve any problems that prohibit the communication.

The ip addresses of adminserver and adminstation switched, so I switched them back.

1. Install the net-tools package
2. Display the ARP cache of **Workstation01**

$ arp -a

workstation02-guust.cbrops.net (10.0.0.11) at bc:24:11:6f:41:33 [ether] on eth0

adminstation-guust.cbrops.net (10.0.0.31) at bc:24:11:b2:b2:09 [ether] on eth0

adminserver-guust.cbrops.net (10.0.0.30) at <incomplete> on eth0

server01-guust.cbrops.net (10.0.0.20) at bc:24:11:88:46:02 [ether] on eth0

firewall-guust.cbrops.net (10.0.0.5) at bc:24:11:48:3e:28 [ether] on eth0

1. Clear the arp cache of **Workstation-01** (hint: ip or arp command)

$ arp -d workstation01-guust

1. On your Kali VM (**adminstation**), run Wireshark and start capturing all the communication that passes by on its network interface.

1. On **Workstation01**, perform the following:
2. Ping to the other VMs
3. Access a website (e.g. [www.howest.be](http://www.howest.be/)) on the internet (hint: wget command)
4. Check again the ARP table of workstation01.

1. Go back to the adminstation VM and inspect the logged traffic in wireshark. What (types of) traffic between the different hosts does the adminstation VM see as a result of these pings?

ICMP ; TCP ; ARP

1. What does this tell you about the network device network.nis.net , i.e. like what type of network device does this network.nis.net VM act?

A hub

1. What impact does this have for a hacker who can connect his laptop to any port of the network device?

The hacker can see all trafic on the network.

# Monitoring the network with arpwatch

As a network administrator, it can be very useful to know what devices are connected to your network. One of the tools that can help with this, is called “arpwatch”.

1. Install arpwatch on server01 and perform the necessary steps to enable it.

Sudo apt update

Sudo apt install arpwatch

sudo systemctl enable arpwatch@eth0

sudo systemctl start arpwatch@eth0

1. Now monitor the arpwatch log files to see if it detects some of the network hosts. Start the server2 and adminserver VMs, and connect to them through SSH from adminstation. When does arpwatch detect network hosts?

Only the first time it communicates with a host it will be captured by arpwatch as for example: “new station 10.0.0.20 bc:24:11:88:46:02 eth0” by systemd and it will be written to the journal. So read the arp logs by running: sudo journalctl -u [arpwatch@eth0](mailto:arpwatch@eth0) where eth0 is the interface it is capturing

# Network Segmentation - Subnets

Up to now, we have a single flat network, where the IP addresses of all devices are in the same subnet, and no restrictions are imposed on communication. This of course is a non-ideal situation when we consider the security of the network, since a hacker who gets access to the network can immediately see all its devices. Similarly, a virus or worm that infects one of the webservers could immediately try to infect all the other devices on the network. Therefore, a crucial practice in network administration is to split (or segment) the network into different sub-networks that group the different hosts according to their application or security level. Communication between the subnets should be strictly controlled, e.g. by firewall policies (which we will see in later labs) and/or ACL (Access Control List) rules.

In this part of the lab, **your VMs will lose access to the internet (and to some degree eachother) and this will stay so for the remainder of the lab**. It will also be necessary to use IP addresses instead of hostnames when you try to connect between VMs. **This is expected behaviour, and will be corrected during the next lab**.

1. The following network range is assigned to you: 10.0.0.0/24.   
   You are asked to **efficiently** design the following three subnets and assign new IP addresses to the VMs, taking into account the below requirements (subnets in order of low-to-high IP ranges):

* 1. **Workstations** subnet
* In this subnet we will place all the workstations that are used by employees. Depending on the level of central management and user rights these machines will pose a medium to high level of risk, and will therefore later be separated from crucial network infrastructure. We expect a maximum of 50 machines in this subnet.
* First useable IP address: reserved for the Gateway
* Second useable IP address: **workstation01**
* third useable IP address: **workstation02**

* 1. **Management** subnet
* This subnet is dedicated to important network management devices such as switches, routers, and administrative servers. Access will be restricted to authorized personnel (administrators) only and strong authentication/encryption methods will be used.
* We expect a maximum of 10 machines in this subnet
* First useable IP address: reserved for the Gateway
* Second useable IP address:  **network (or switch)**
* third useable IP address: **adminserver**
* fourth useable IP address: **adminstation**
  1. **DMZ** subnet (**D**e**M**ilitarized **Z**one)
* This subnet is mainly meant for servers that will be accessible from the internet, and in this way pose a large security risk. Examples of machines that are in the DMZ network are typically webservers and mail servers. We expect a maximum of 5 machines in this subnet.
* First useable IP address: reserved for the Gateway
* Second useable IP address: **server01**
* third useable IP address: **server02**

Fill out the subnet table and IP address table for the different hosts

|  |  |  |  |
| --- | --- | --- | --- |
| **Subnet name** | **Network address** | **prefix** | **netmask** |
| **Workstations** | 10.0.0.0 | /26 | 255.255.255.192 |
| **Management** | 10.0.0.64 | /28 | 255.255.255.240 |
| **DMZ** | 10.0.0.80 | /29 | 255.255.255.248 |

Table 1 Subnet documentation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **hostname** | **IP address** | **Subnet mask** | **Default Gateway** | **DNS** | **Operating System** |
| **workstation01** | 10.0.0.2 | 255.255.255.192 | 10.0.0.1 | 10.0.0.1 | Ubuntu |
| **workstation02** | 10.0.0.3 | 255.255.255.192 | 10.0.0.1 | 10.0.0.1 | Debian |
| **network** | 10.0.0.66 | 255.255.255.240 | 10.0.0.65 | 10.0.0.65 | Alpine |
| **adminserver** | 10.0.0.67 | 255.255.255.240 | 10.0.0.65 | 10.0.0.65 | Ubuntu |
| **adminstation** | 10.0.0.68 | 255.255.255.240 | 10.0.0.65 | 10.0.0.65 | Kali |
| **server01** | 10.0.0.82 | 255.255.255.248 | 10.0.0.81 | 10.0.0.81 | Debian |
| **server02** | 10.0.0.83 | 255.255.255.248 | 10.0.0.81 | 10.0.0.81 | Fedora |
|  |  |  |  |  |  |

Table 2 Host documentation

1. Implement the segmented network you designed by applying the above IP addresses to the different hosts in the network (ignore the **firewall** and **switch** for now). Make sure that these IP address changes are persistent and survive a reboot of the machines!  
   Remember that - for devices with a fixed IP – you should always also configure the IP address in a second file, and that the IP addresses should be corrected in the DNS server!
2. Now log in to your Kali VM (adminstation), and start capturing the communication on its network interface with wireshark.

1. On **workstation01**, again perform the following:
2. Clear the arp cache of Workstation-01 (hint: ip or arp command)
3. Ping to the other VMs

Can you still connect to all VMs? No only the ones in the subnet, so workstation02

What (types of) traffic between the different hosts does the **adminstation** VM see in wireshark as a result of these pings? Only the ICMP request and repies of the pings between workstations and all ARP traffic

What impact does this have for a hacker who can connect his laptop to any port of the **network** device? They would be able to see all trafic between the hosts

1. Now change the IP address of the **adminstation** VM in such a way that it becomes part of the Workstations subnet. What changes do you notice if you perform the tests from assignment 9?

I also see DNS traffic

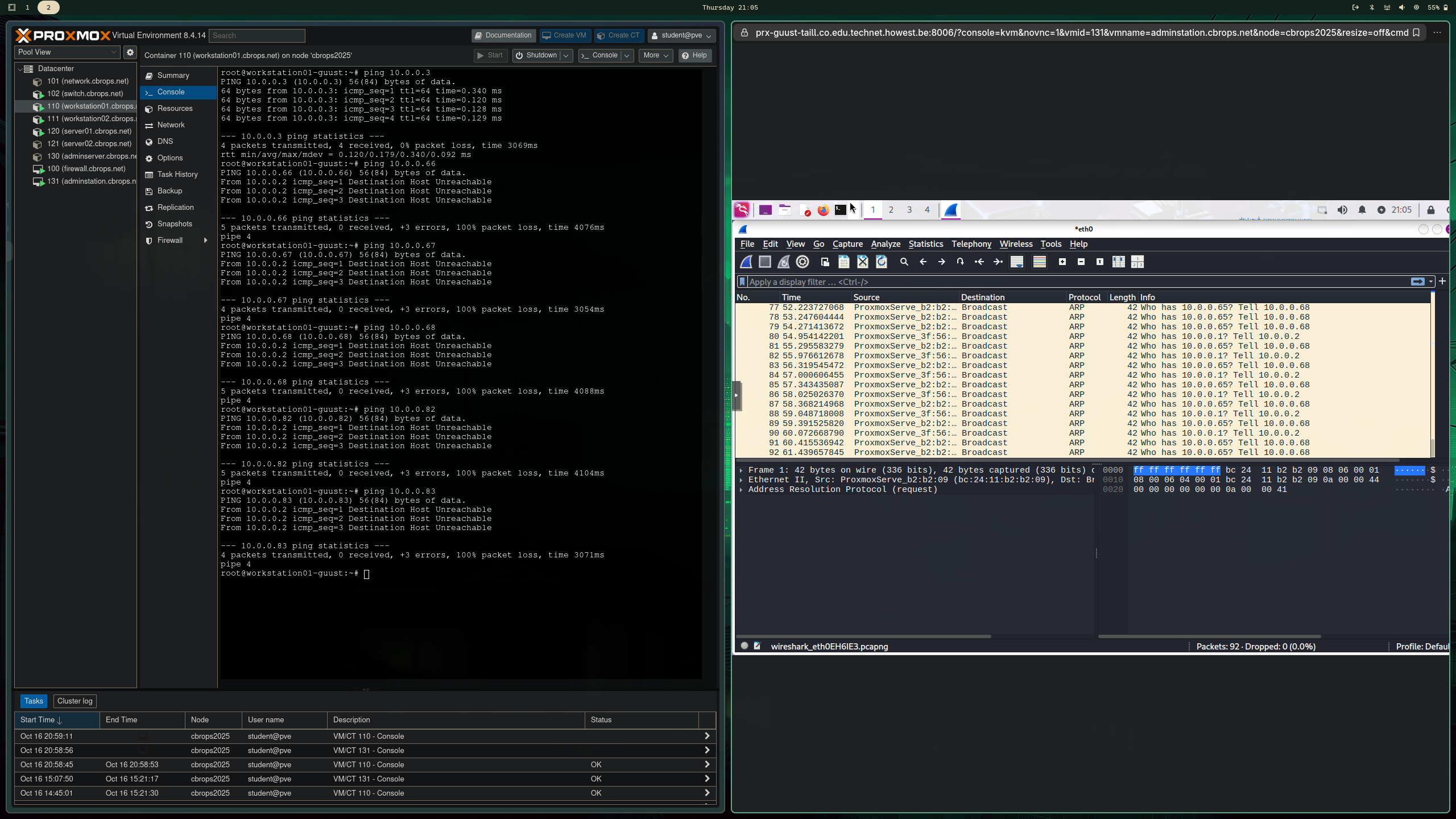
1. Change the IP address of the **adminstation** VM back to its correct value in the management subnet

# Replacing the Network Device

As you may have noticed, the **network** VM is not ideally suited for secure networking because any device that is connected to it can see too much information about the network traffic. Therefore, we decided to replace it by a better device, **switch.cbrops.net** . This **switch.cbrops.net** VM uses a service called “OpenVSwitch” (OVS) [1] to implement the virtual equivalent of a hardware network switch.

1. Turn off the **network.cbrops.net** VM and turn on the **switch.cbrops.net** device. Add your personalized user account to the **switch** VM, personalize its hostname (as you previously did for the other VMs) and assign the correct IP address and network settings (you can use the same IP address as was used for the **network** VM).
2. **Important:** In the current environment, a bug seems to prevent OpenVSwitch from starting. This can be detected by running “ip a”, if the “CBR-net” interface is not present, this indicates OVS is malfunctioning. To solve this (temporarily), run “**ovs-vswitchd --detach**” in the switch VM and **reboot** the VM. After the reboot, the interface should be present and have an active IP address.
3. Now log in to your Kali VM (**adminstation**), and start capturing the communication on its network interface with wireshark.

1. On **workstation-01**, once more perform the following:
2. Clear the arp cache of **Workstation-01** (hint: ip or arp command)
3. Ping to the other VMs
4. Make a screenshot with the different ping commands (+ their output), and the corresponding wireshark capture.



1. Do you notice any difference when connecting to the other VMs?   
   What (types of) traffic between the different hosts does the adminstation VM see in wireshark as a result of the pings?  
   The limitation stays the same so I can only ping hosts in the same subnet, but the traffic is no longer visible on the adminstation.
2. What impact does this have for a hacker who can connect his laptop to any port of the switch device? Compare this with the results for the network VM.  
   Now the traffic you can capture is limited to the subnet you’re in

**Optional**: while it is significantly more difficult to do, hackers may still be able to listen in on the communication between devices using special man-in-the-middle techniques. If you want, you can try this here.

# Network Monitoring – SPAN Port

As we’ve seen, replacing the network VM by the switch VM significantly reduces the amount of network traffic that can be seen by a hacker. However, in some cases (e.g. the IDS or SIEM systems we’ll see later on) we as administrators still want to be able to see what is happening on the network. For this reason, network switches typically allow creating so-called SPAN- or mirror-ports that mirror the traffic from one port (or a set of ports) to a specific port at which we can listen in on its communication.

1. Configure the OpenVSwitch service on the switch VM in such a way that all traffic to/from **workstation01** is mirrored to the port on which **adminstation** is connected [2] [3]. To do this, use the following command:

ovs-vsctl -- set Bridge CBR-net mirrors=@m \

-- --id=@workstation01 get Port eth1 \

-- --id=@adminstation get Port eth6 \

-- --id=@m create Mirror name=mymirror select-dst-port=@workstation01 select-src-port=@workstation01 output-port=@adminstation

1. Start logging the traffic with wireshark on adminstation and once more perform the following on workstation-01:
2. Clear the arp cache of Workstation-01 (hint: ip or arp command)
3. Ping to the other VMs

Also perform a ping from server-01 to server-02.

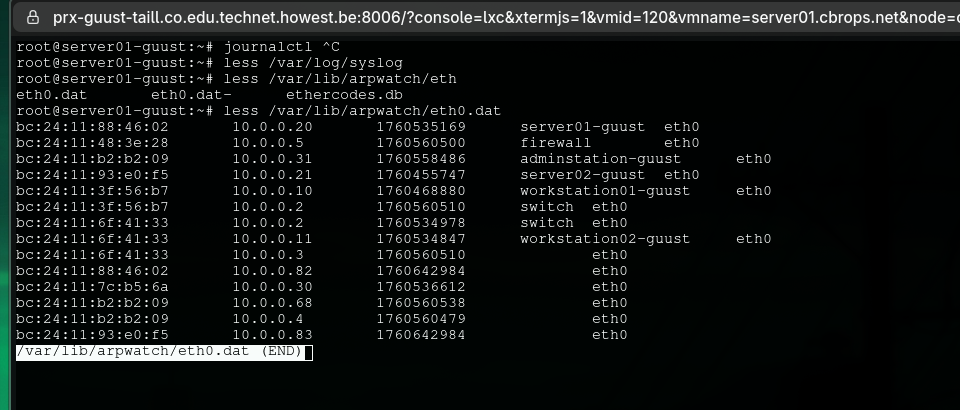
What traffic can you see in the wireshark capture?  
Only the pings and the resuest to delete arp cache from the workstation but no traffic between the servers.

1. remove the SPAN/Mirroring ports on the switch VM with the following command:

ovs-vsctl clear bridge CBR-net mirrors

# One last thing…

1. Monitor the logs of arpwatch, which you installed earlier on server01. Did it detect the modified IP addresses?  
   Yes:



# Bibliography

|  |  |
| --- | --- |
| [1] | "OpenVSwitch," [Online]. Available: https://www.openvswitch.org. [Accessed 09 2024]. |
| [2] | "OpenVSwitch: ovs-vsctl man page," 21 06 2024. [Online]. Available: https://www.openvswitch.org/support/dist-docs/ovs-vsctl.8.txt. |
| [3] | "OpenVSwitch - basic configuration FAQ," [Online]. Available: https://docs.openvswitch.org/en/latest/faq/configuration/. [Accessed 09 2024]. |

1. The hostnames of the VMs used in this document are the generic, non-personalised versions. For your own environment, you should of course use the personalized versions. E.g. workstation01 becomes workstation01-<firstname> [↑](#footnote-ref-2)