Lab 02: Network Segmentation part 1 - Subnets

Network Infrastructure Security (CSP)  
 2024-2025

© Henk Brouckxon, Nico Declerck, Chris Roets

# Introduction

## Lab concept

In the previous lab, we started configuring the NIS network. However, it is clear that some problems exist in this network with regard to security. During this lab, we will start improving upon the originally flat (and insecure) layout of the network. 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.nis.net” device and its security impact.

## Learning goals

* Monitoring network traffic with wireshark
* Network segmentation using subnets + its impact on security
* 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 NIS lab environment

A diagram of a computer system

Description automatically generated

Figure 1 Equivalent physical layout for the NIS network

A computer icons on a black background

Description automatically generated

Figure 2 Initial logical layout for the NIS network

# Baseline – A Flat Network

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

* firewall.nis.net
* network.nis.net
* workstation-01.nis.net
* workstation-02.nis.net
* server-01.nis.net
* adminstation.nis.net

1. Log in to the **workstation-01** VM, and check that you can ping to **workstation-02** and **server-01**. If this is not possible, try to solve any problems that prohibit the communication.

A screen shot of a computer

Description automatically generated

Workstation 1 works

A screen shot of a computer

Description automatically generated

Server 1 also works

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

the port is **eth0**

by default there is nothing happening, so I did a quick ping from workstation 1 to server 1 for fun

A screen shot of a computer

Description automatically generated

The ARP looks satisfying.

A screenshot of a computer

Description automatically generated

Also some random pings from an unknown host.

1. On **Workstation-01**, perform the following:
2. Clear the arp cache of Workstation-01 (hint: ip or arp command)

doas arp -d <ip>

doas ip neigh flush all

1. Ping to the other VMs

A screenshot of a computer program

Description automatically generated

The table for ARP also filled

A screenshot of a computer

Description automatically generated

1. Access a website (e.g. [www.howest.be](http://www.howest.be)) on the internet (hint: wget command)

A screenshot of a computer

Description automatically generated

What (types of) traffic between the different hosts does the **adminstation** VM see in wireshark as a result of these pings? 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? What impact does this have for a hacker who can connect his laptop to any port of the **network** device?

This device seems to be a switch, considering the among of ports being used currently. It looks like it also solves MAC addresses and acts like a layer 2 device.

A bit of googling and a question asked to the teacher, they said it is acts like a switch, but is not a switch. Hence, it should be a **hub.** I googled what a hub Is, and the difference is that the hub really just broadcasts information to the whole network. That is probably the case why I saw the ARP requests overall, or maybe it is yet another case. In any case, I know it broadcasts traffic to everyone on the network.

This may be impactful as for example you can easily spoof ip addresses, or sniff upon everything in the network in this case.

# Network Segmentation - Subnets

Up to now, the network has been configured as 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 same networking 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 the application of firewall policies (which we will see in later labs) and/or ACL (Access Control List) rules.

During this part of the lab, some of your VMs may lose access to the internet and some other services. It may also be necessary to use IP addresses instead of hostnames when you try to connect between VMs. This will be corrected during the next lab.

1. The following network range is assigned to you: 10.0.0.0/24.   
   You are now asked to **efficiently** design three subnets and assign new IP addresses to the VMs, taking into account the following 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 th0e level of central management and user rights these machines will pose a medium to high level of risk, and are therefore 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: **workstation-01**
* third useable IP address: **workstation-02**

* 1. **Management** subnet
* This subnet is dedicated to crucial network management devices such as switches, routers, and administrative servers. Access should be restricted to authorized personnel (administrators) only and strong authentication methods should 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 should 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: **server-01**
* third useable IP address: **server-02**

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

How to calculate all of this:

First they say we need 50 machines max for **workstation group.** We have the default network being 10.0.0.0/24, which does not work well for us.

What I did is start calculating with **ipcalc** a subnet that will allow less IP addresses. I ended up with /26 which allows 62 usable addresses.

A screenshot of a computer code

Description automatically generated

The broadcast is 63, which is already all host bits set to 1. Hence, to calculate the other one, I need to choose the next address, and do a calculation with that. The next address is .64 and I need to find a subnet that will allow for 10 machines at least. So I chose that IP address and continued to change the subnet.

A screenshot of a computer code

Description automatically generated

So I end up with /28 (10.0.0.64/28). It is important to know why I started with .64, and that is because it is not inside the /26 network, and hence the bits will not collide.

Repeat the same and the next IP address is .80, hence I take that one and continue with the subnets. I end up with 10.0.0.80/29, since I need to have about 5 hosts.

A screenshot of a computer code

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| Subnet name | Network address | prefix | netmask |
| Workstation | 10.0.0.0/26 | 26 | 255.255.255.192 |
| Management | 10.0.0.64/28 | 28 | 255.255.255.240 |
| DMZ(DeMilitarized Zone) | 10.0.0.80/29 | 29 | 255.255.255.248 |

Table 1 Subnet documentation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| hostname | IP address | Subnet mask | Default Gateway | DNS | Operating System |
| workstation-01 | 10.0.0.2/26 | 255.255.255.192 | 10.0.0.1/26 | 10.0.0.5 | Alpine Linux v3.19 |
| workstation-02 | 10.0.0.3/26 | 255.255.255.192 | 10.0.0.1/26 | 10.0.0.5 | Alpine Linux v3.19 |
| network/switch | 10.0.0.66/28 | 255.255.255.240 | 10.0.0.65/28 | 10.0.0.5 | Alpine Linux v3.19 |
| adminserver | 10.0.0.67/28 | 255.255.255.240 | 10.0.0.65/28 | 10.0.0.5 | Ubuntu 22.04.4 LTS |
| adminstation | 10.0.0.68/28 | 255.255.255.240 | 10.0.0.65/28 | 10.0.0.5 | Kali GNU/Linux Rolling 2024.2 |
| server-01 | 10.0.0.82/29 | 255.255.255.248 | 10.0.0.81/29 | 10.0.0.5 | Alpine Linux v3.19 |
| server-02 | 10.0.0.83/29 | 255.255.255.248 | 10.0.0.81/29 | 10.0.0.5 | Alpine Linux v3.19 |

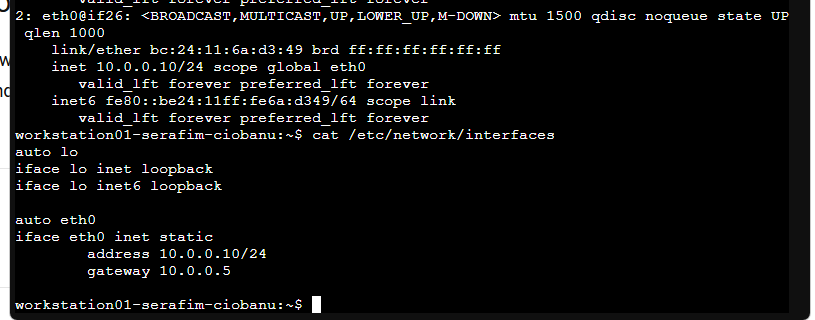
Table 2 Host documentation

1. Implement the segmented network you designed by applying the above IP addresses to the different hosts in the network. 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!

A screenshot of a computer

Description automatically generated

I have also made sure to put those addresses in the DNS Server.



ALPINE LINUX

As far as I understand first thing is to set up the /etc/network/interfaces (first thing)

A screen shot of a computer

Description automatically generated

and restart the service.

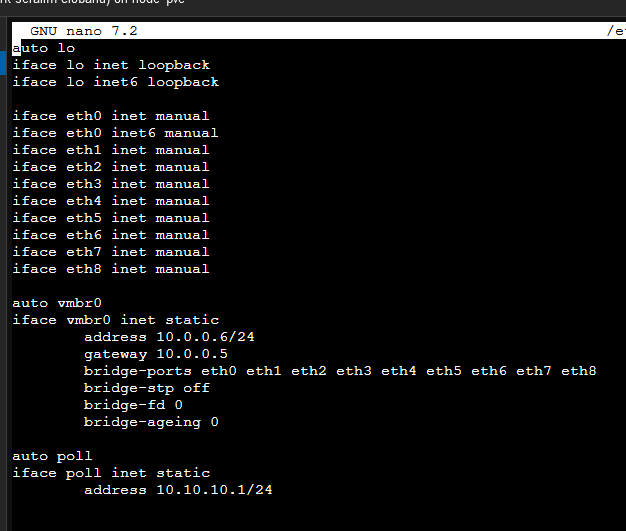
A screenshot of a computer

Description automatically generated

doas service networking restart

The DNS stays the same (10.0.0.5)

NETWORK MACHINE



A computer screen with white text

Description automatically generated

UBUNTU

To change - go to /etc/netplan

A screenshot of a computer

Description automatically generated

then do **sudo netplan try**

**Looks like the IP addresses were now changed**

1. Now log in to your Kali VM (adminstation), and start capturing the communication on its network interface with wireshark.

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

I can not connect to all VMs but only to workstation01. The traffic present is UDP (DNS), and ICMP. All this adminstation sees is the communication that the machine tries to do in its own network. It does not see the tries to access other machines.

If he connects to the network device then I think it will still be vulnerable since the device is a hub and it will broadcast everything.

Can you still connect to all VMs? What (types of) traffic between the different hosts does the **adminstation** VM see in wireshark as a result of these pings?  What impact does this have for a hacker who can connect his laptop to any port of the **network** device?

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 7?

By cleaning the ARP I now see ARP, UDP (DNS), and ICMP traffic. Since the machines need to know the gateway and other parts for the networking.

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

A screenshot of a computer program

Description automatically generated

# 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.nis.net** . This **switch.nis.net** VM uses a service called “OpenVSwitch” (OVS) [1] to implement the virtual equivalent of a hardware network switch.

1. Turn off the **network.nis.net** VM and turn on the **switch.nis.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).

A screenshot of a computer program

Description automatically generated

Alpine wiki says **setup-user** can be used so I did.

It has also been added to the wheel group automatically.

A screen shot of a computer

Description automatically generated

Hostname looks fine

A screen shot of a computer

Description automatically generated

Check the /etc/network/interfaces of course

Now to change it to the network one.

A screen shot of a computer

Description automatically generated

And IP updated, now to restart the service.

A screenshot of a computer program

Description automatically generated

Command: **doas service networking restart**

A screenshot of a computer program

Description automatically generated

Some pings even work so it is already a good sign.

And I can not ping outside my network, or the default gateway, but I can ping the DNS server (Firewall).

1. 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)

Command: **doas ip neigh flush all**

I also did **doas arp -d <ip>**

A screen shot of a computer

Description automatically generated



1. Ping to the other VMs

A computer screen shot of a black screen

Description automatically generated

A screenshot of a computer

Description automatically generated

Make a screenshot with the different ping commands (+ their output), and the corresponding wireshark capture. 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 these pings?  What impact does this have for a hacker who can connect his laptop to any port of the **switch** device? Compare this with theresults for the **network** VM.

A screenshot of a computer

Description automatically generated

Comparing to the network device, this one actually does not broadcast the traffic to the outside world.In this case we can only see ARP from the **adminstation itself**, from within its own network. While the workstation01 can still ping to the machines of its own, it does not interfere with other networks (so we succesfully split the machines).

**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 **workstation-01** is mirrored to the port on which **adminstation** is connected [2] [3]. Start logging the traffic with wireshark on adminstation and once more perform the following on workstation-01:

Commands:

**doas ovs-vsctl show (to show the ports)**

**doas ovs-vsctl list Interface (to find the UUIDs)**

**I thought I needed to create a bridge and add ports to it, but It told me I already had one (nis-lan)**

**ovs-vsctl --if-exists del-br br0 – to delete a bridge**

***ovs-vsctl clear Bridge nis-lan mirrors - to clear any mirrors from the nis-lan bridge.***

According to the image below, I can suppose that the ports used are eth1 (workstation01) and eth5 (adminstation)

A diagram of computer servers

Description automatically generated

workstation01 – eth1 - bad79459-8eab-4f78-8311-9a1705e0184c

adminstation – eth5 - e94d2ff8-b2f0-4bf2-9be7-859ce6777ace

A screen shot of a computer code

Description automatically generated

ovs-vsctl -- --id=@m create Mirror name=mymirror select-dst-port=<UUID\_eth1> select-src-port=<UUID\_eth1> output-port=<UUID\_eth5> \

-- set Bridge nis-lan mirrors=@m

Finally made it work:

A screenshot of a computer screen

Description automatically generated

doas ovs-vsctl -- set Bridge nis-lan mirror=@m \

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

-- --id=@eth5 get Port eth5 \

-- --id=@m create Mirror name=mirror0 select-dst-port=@eth1 select-src-port=@eth1 output-port=@eth5

A computer screen shot of a computer program

Description automatically generated

A computer screen shot of a black screen

Description automatically generated

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

doas arp -d <ip>

doas ip neigh flush all

1. Ping to the other VMs

A screenshot of a computer program

Description automatically generated

Pings work of course.

And the traffic can be also seen.

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

What traffic can you see in the wireshark capture?

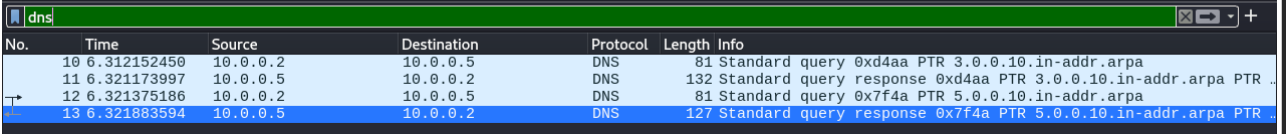
I can see ARP traffic from all the zones, asking for the “default gateway” which do not get responses.

Apart from that, I can also see ICMP traffic.

A screenshot of a computer

Description automatically generated

I can also see DNS traffic.



A screenshot of a computer

Description automatically generated

# One More Thing…

1. During the lab, you may have noticed that one of the server VMs tries to send out some unexpected network packets to the outside world from time to time. What VM does this, and what is the hidden message that is contained in these messages?

It was at the very beginning

A screenshot of a computer

Description automatically generated

I only got the response from it, unfortunately, and I can not redo it at this point, but this machine was surely outside of the network (192.168.30.2), which might be the proxmox machine itself, however it seems to have some code in it – **stephenz123!** – or something among the lines. No clarity upon what this is, but It surely looks like something interesting.

# 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. workstation-01 becomes workstation-01-<firstname> [↑](#footnote-ref-2)