 ****

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

Lab 03 – NAT Networks

v1.2

TABLE OF CONTENTS

**[1.](#_Toc170113325)****[Purpose](#_Toc170113325)** [2](#_Toc170113325)

**[2.](#_Toc170113326)****[Prerequisites](#_Toc170113326)** [2](#_Toc170113326)

**[3.](#_Toc170113327)****[Requirements](#_Toc170113327)** [2](#_Toc170113327)

**[4.](#_Toc170113328)****[Instructions](#_Toc170113328)** [2](#_Toc170113328)

# Purpose

In this lab we will practically explore the concepts of NAT networks, how we can use them in different applications, as well as PAN networks and their functionality. We will also explore the concept of Transversal NAT and its application with different applications.

# Prerequisites

* Basic understanding of networking.
* Basic understanding of VMWare.
* Kali Linux machine.
* Windows Machine with RDP activated.
* Metasploitable2 machine.
* Anydesk software installer.

# Requirements

* Verify Kali Linux machine has the tool ARPSpoof
* Mitmproxy tool installed in Kali
* Windows machine

# Instructions

**Activity 1 NAT Rules(50%)**

For this exercise we need Kali and a Windows machine connected to the same NAT Network and with access to the internet.

**PART 1: intercepting the traffic (25%)**

Provide a screenshot of the following:

1. Windows a Kali machines in the same network and with access to the internet (Ping each other machines and 8.8.8.8)

|  |
| --- |
|  |

1. Download or check that your Kali Linux has the tool ARPSpoof installed and ready to use.

|  |
| --- |
|  |

1. Use your windows machine to navigate to any web service like google

|  |
| --- |
|  |

1. Execute the following arpspoof command in a Kali terminal

*arpspoof -i <interface> -t <Target-IP> <Gateway-IP>*

|  |
| --- |
|  |

1. Try to navigate again in your Windows machine, and tell what happened

|  |
| --- |
| When I tried to navigate to Google on the Windows machine while ARPSpoof was running, the page did not load correctly. This happened because ARPSpoof redirected all the traffic from the Windows machine to Kali, but since IP forwarding was not enabled yet, the packets were not forwarded to the gateway. As a result, the Windows machine temporarily lost internet connectivity. |

1. Go back to Kali and in a sudo terminal (root) execute the following command

*echo 1 > /proc/sys/net/ipv4/ip\_forward*

explain what it is used for.

|  |
| --- |
| I use this command to enable IP forwarding on Kali. Normally, Linux does not forward packets between the Windows machine (192.168.10.5) and the gateway (192.168.10.2). By enabling IP forwarding, Kali acts like a router, so the Windows machine keeps internet access while I can still intercept and analyze the traffic. |

1. Try to navigate again in your Windows machine, and tell what happened

|  |
| --- |
| I tried to navigate again on my Windows machine after enabling IP forwarding. This time the internet worked normally, and I was able to access Google without problems. The reason is that Kali was now forwarding the traffic to the gateway, so the Windows machine kept internet access while Kali was still in the middle capturing the traffic. |

1. Open Wireshark in Kali Linux and identify the traffic coming from the Windows machine.

|  |
| --- |
|  |

1. In your Windows machine, navigate to this web service <http://testphp.vulnweb.com/login.php> and introduce the given credentials. (they are in the same webpage)

|  |
| --- |
|  |

1. In Kali Linux, In your Wireshark in the traffic filter bar put the following filter:

*http contains "test"*

Then right click the POST connection packet and select FOLLOW and HTTP Stream in the new window, search for:

*uname=*

Explain what you can see and how this is possible, provide also a screenshot.

|  |
| --- |
|  |

**PART 2: decrypting the traffic (25%)**

For this exercise we need Kali and a Windows machine connected to the same NAT Network and with access to the internet.

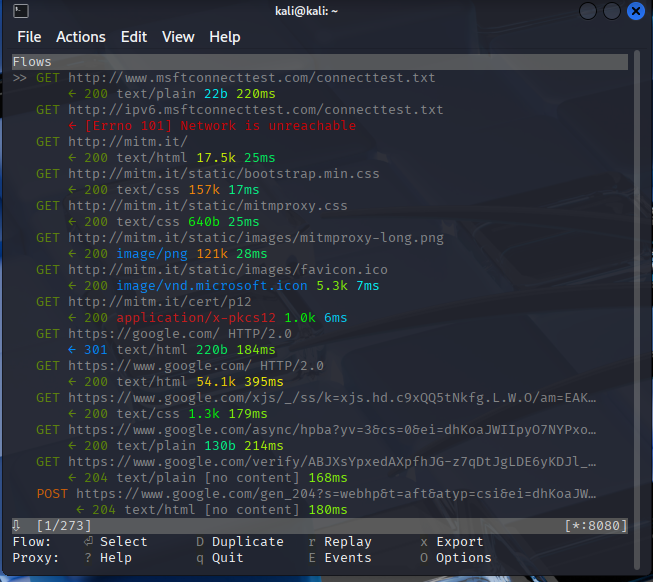
1. Now lets use a tool to intercept HTTPS traffic, in another terminal execute the command:

* *Mitmproxy*

It will execute an interactive window like this

A screenshot of a computer

AI-generated content may be incorrect.



1. Now in another terminal, execute the following commands
   * *sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080*
   * *sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 8080*
   * *sudo iptables -t nat -L*

Explain that those commands do:

**These commands configure iptables rules in the NAT table to redirect all incoming HTTP (port 80) and HTTPS (port 443) traffic to port 8080, where mitmproxy is listening. This way, Kali can transparently intercept and decrypt both HTTP and HTTPS traffic without having to manually set the proxy in Windows. The last command lists the rules in the NAT table to verify that the redirections were successfully applied. At first, I mistakenly added one of the rules twice, but later I cleared the NAT table and re-added only the correct rules.**

|  |
| --- |
|  |

1. In the windows machine visit <https://login.procampus.study/login/index.php>   
   what is the browser telling you?

|  |
| --- |
|  |

1. Accept, procced to the web service and provide any credentials in the login form.

When I visited https://login.procampus.study/login/index.php, the browser showed a NET::ERR\_CERT\_AUTHORITY\_INVALID error with HSTS enabled. Because the website enforces HTTP Strict Transport Security, it was not possible to bypass the warning and proceed to the login page. This prevents man-in-the-middle attacks even if a fake certificate is installed

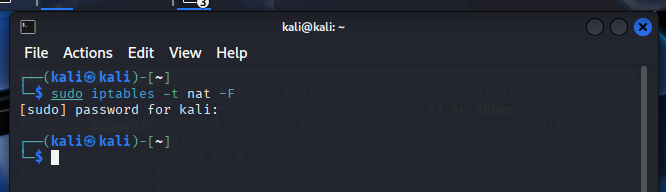
1. Go back to your Kali machine and in the terminal where you executed mitmproxy, look for this line:



Click on it and explain what you can see.

|  |
| --- |
|  |
| I went back to my Kali machine where mitmproxy was running and clicked on the POST /login/index.php line. In the request details I could clearly see the form data, including the username and password I entered from the Windows machine. This shows that mitmproxy was able to intercept and decrypt the HTTPS traffic, displaying the credentials in plain text. |

1. Stop the arpspoof and delete the iptables rules.

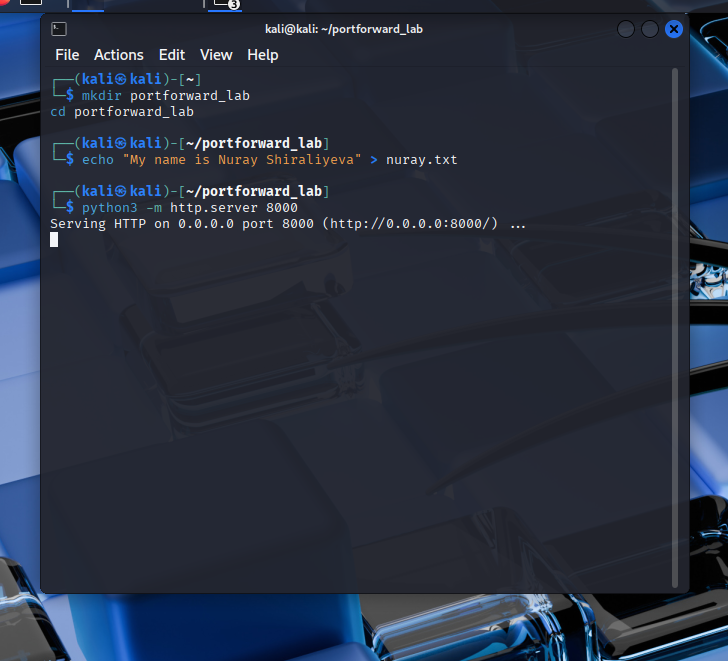


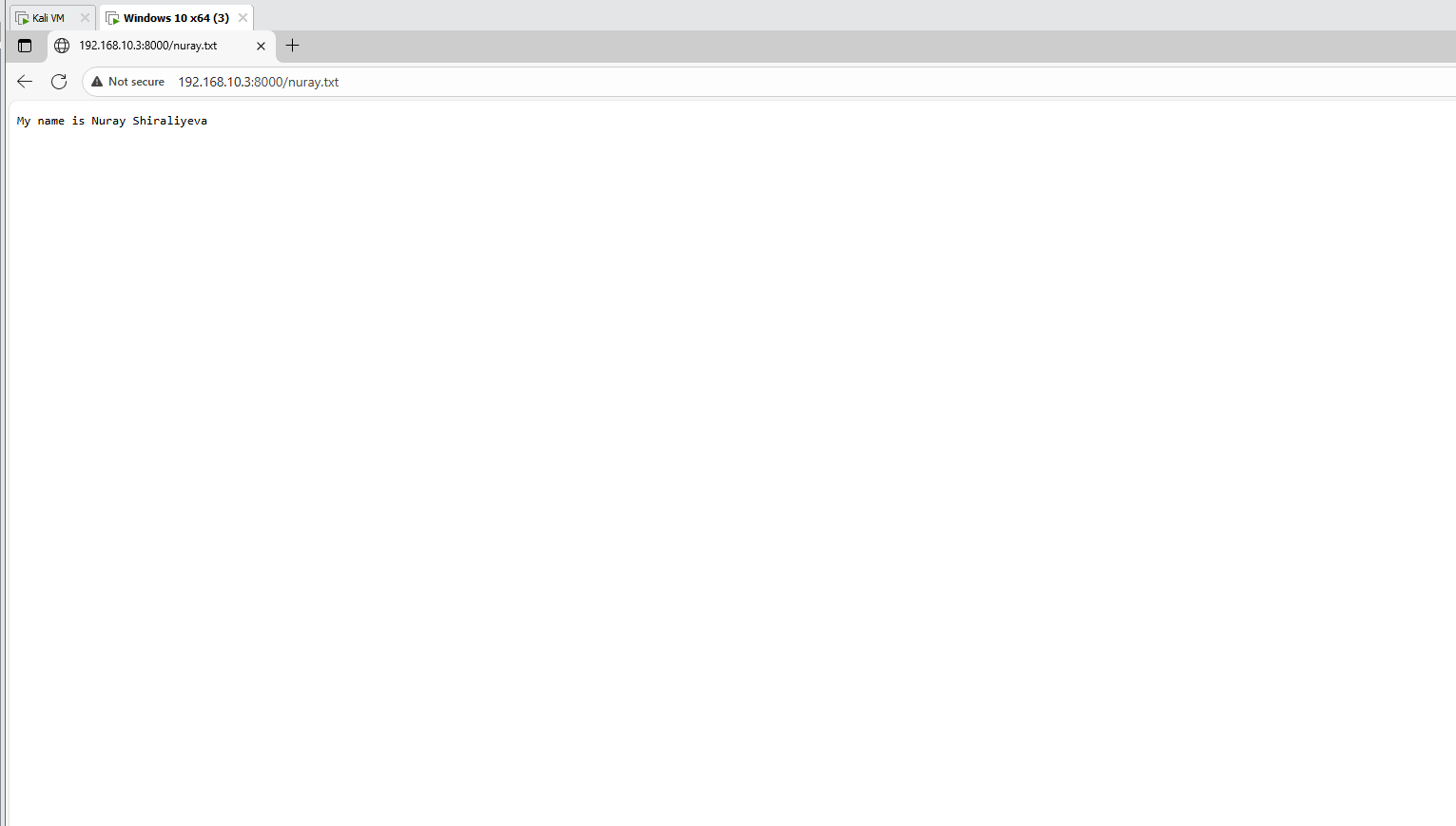
**Activity 2 Port Forwarding (30%)**

For this activity we need 3 Virtual machines, Kali Linux, Windows Machine and metasploitable2. All connected in the same NAT network.

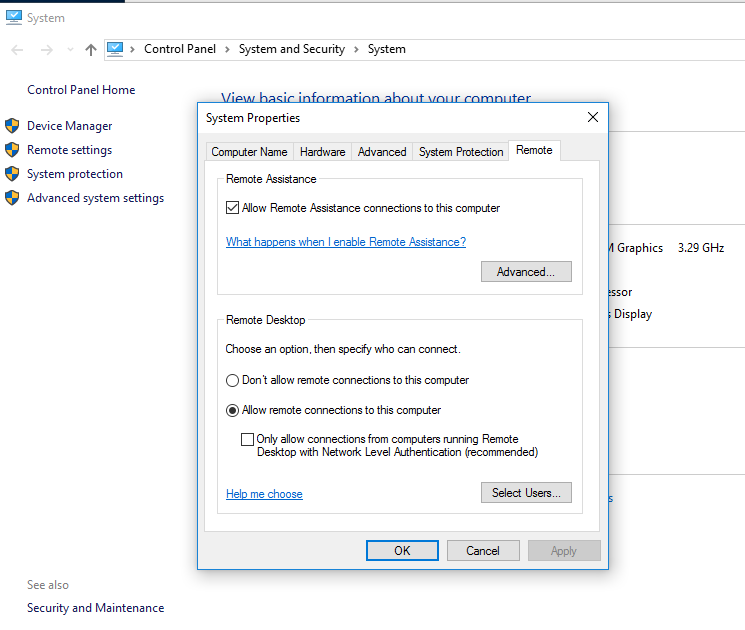
**NOTE:** this is a guided lab you should replace all the screenshots with your own.

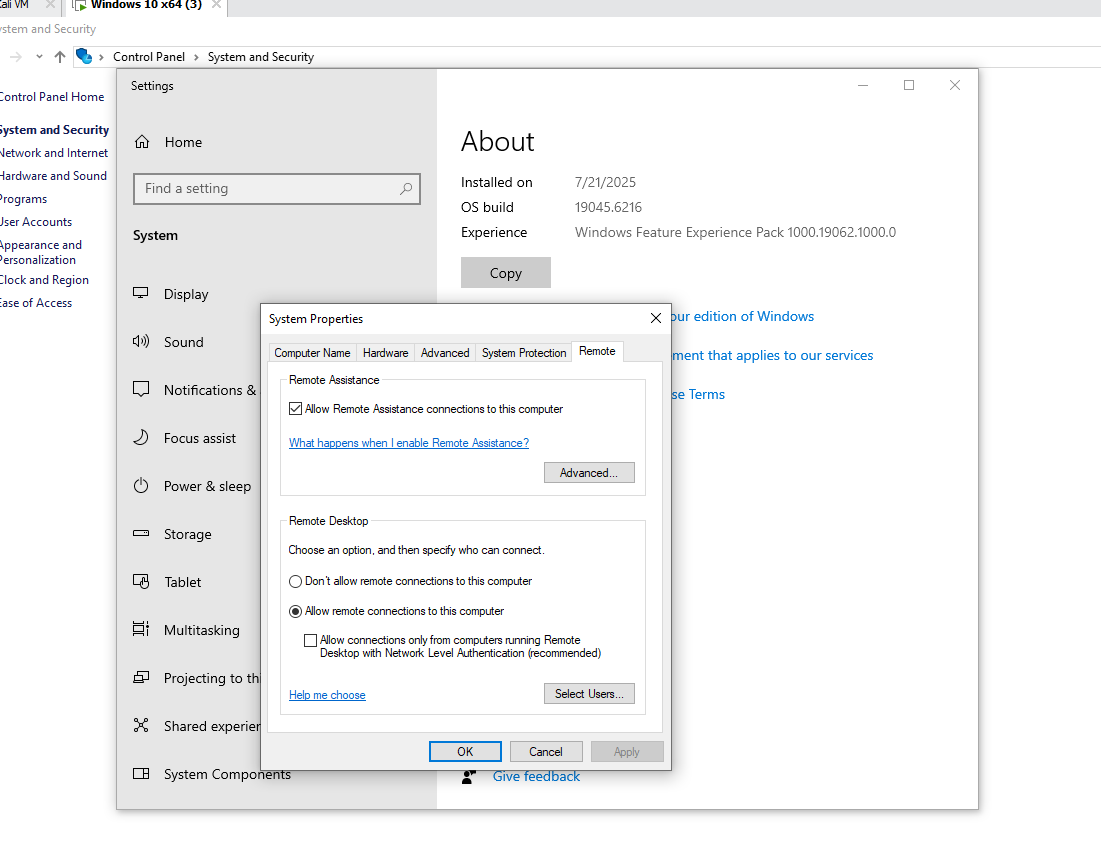
1. In your Kali Machine create a new folder and from the terminal move to that folder, create a new text document with your name on it. When finished take one screenshot of that folder and file and then execute the following command:
   * python -m http.server



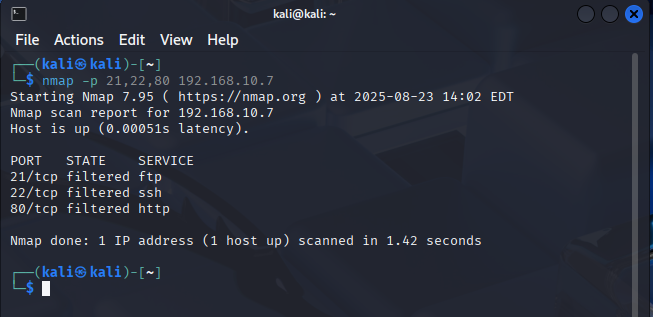


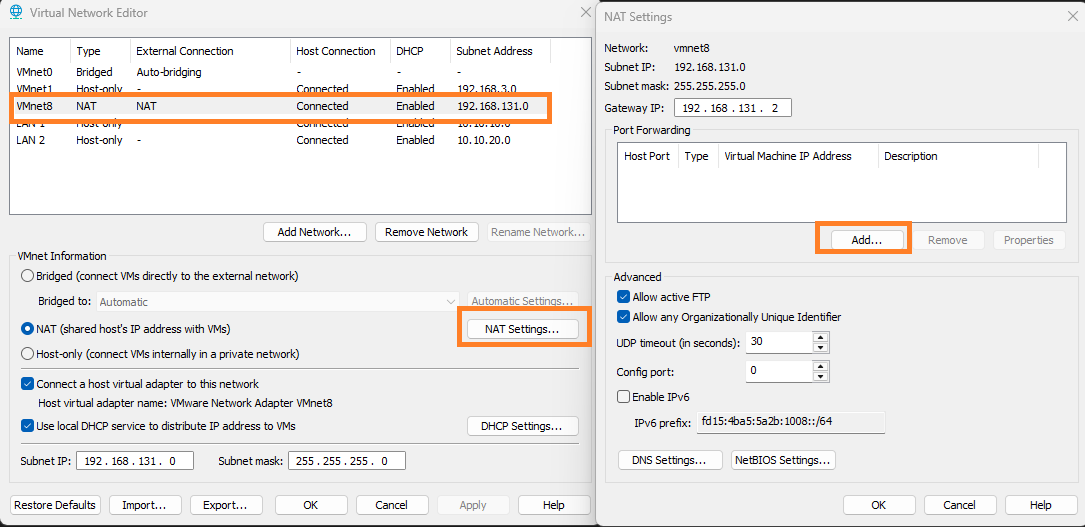
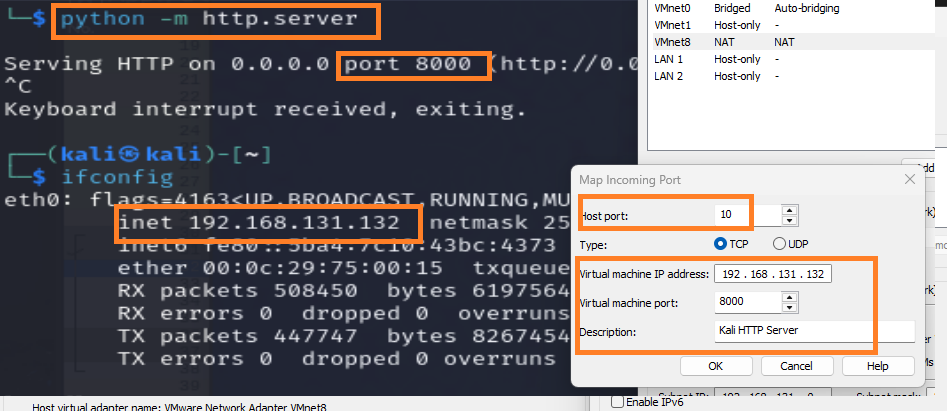


1. In your Windows machine, enable the RDP protocol, for example:  
   



1. In metasploitable machine we already know what the default services are running so obtain the IP Address and chose from one of this default ports (21, 22, 80).



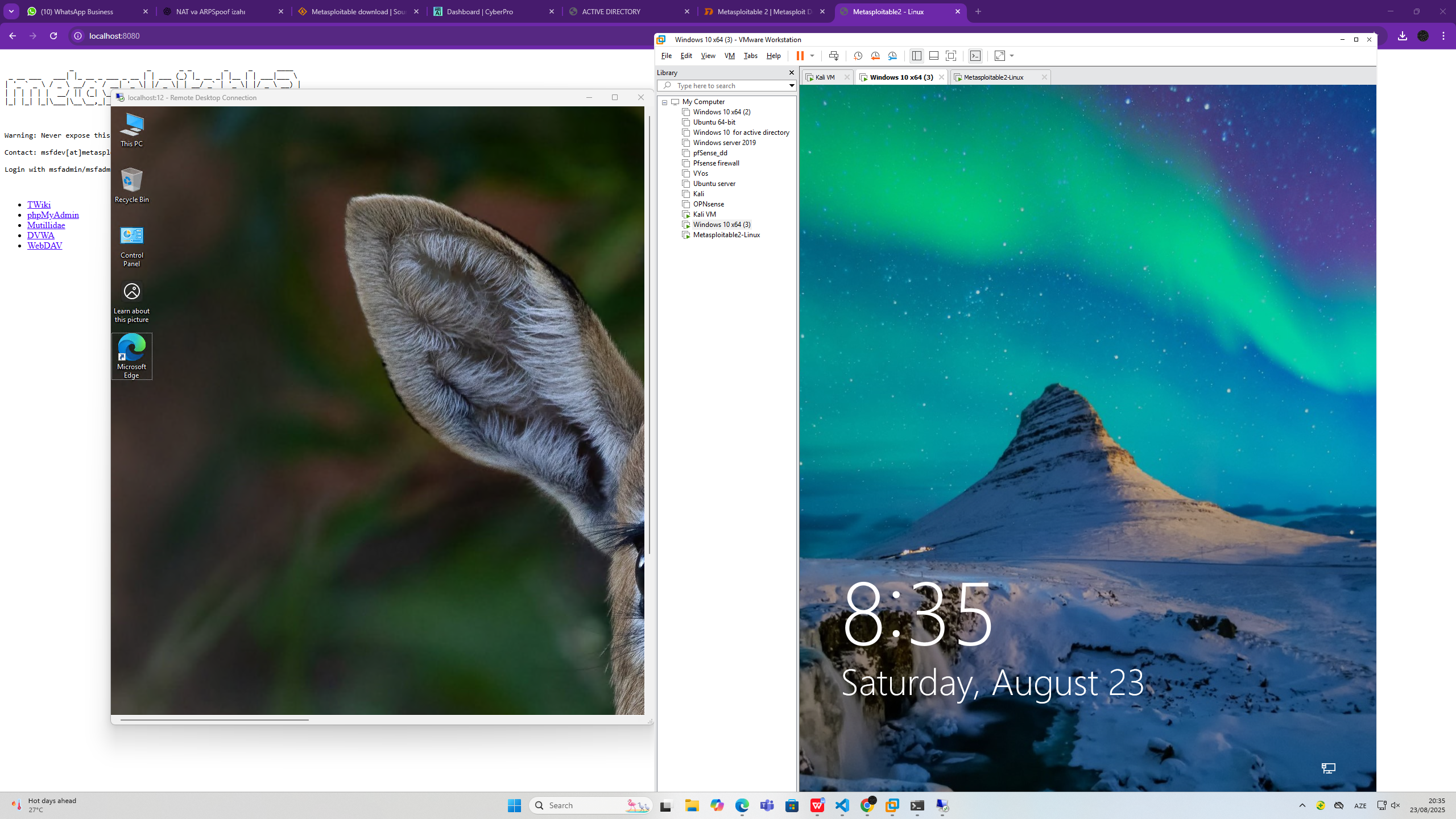
1. In the Virtual network editor of VMWare choose your NAT network selected for this lab, click NAT Settings and then in port forwarding click Add button.  
   
2. In the Map Incoming Port (new opened window) select the host port (from your physical machine) can be any one and start linking the services in your machines, for example in the Kali Linux Machine.  
   
3. For Windows do the same with RDP and remember the internal and external port should be 3389 for easy connection. And for MetaSploitable should be any of the mentioned before. Your configuration should look like this:

A screenshot of a computer

AI-generated content may be incorrect.

1. Now explain how this is possible and provide screenshots of your successful connections.

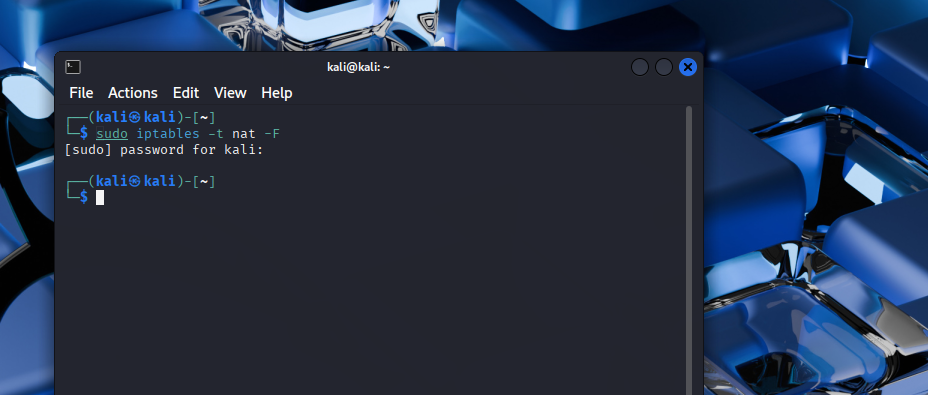
|  |
| --- |
|  |

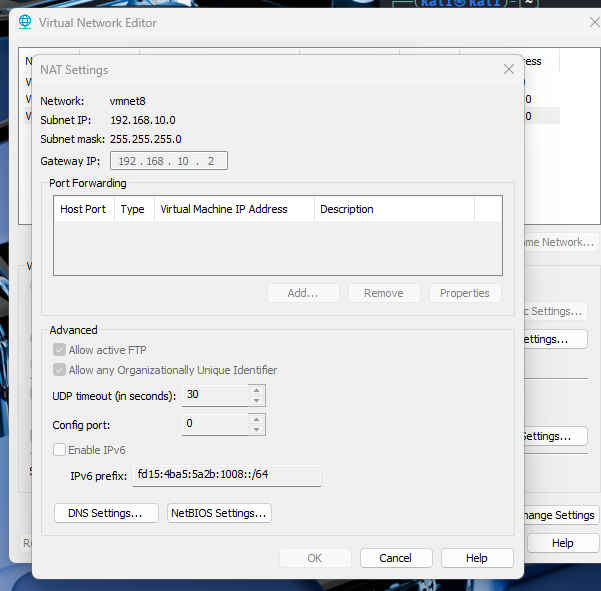


1. Ask your classmates to connect to your physical IP Address and those ports you mapped as “host ports” then answer this questions:

|  |  |
| --- | --- |
| Question | Answer |
| If you run Wireshark in your physical machine, you can see their connections going to your Virtual machines or your physical IP? | When I ran **Wireshark** on my physical machine, I saw the connections coming to my **physical IP** address instead of directly targeting the virtual machines (VMs). This is due to **NAT Port Forwarding**. VMware forwards the traffic from the **host port** on my physical machine to the respective virtual machine's port.  So, even though the incoming traffic is reaching my physical IP address, VMware NAT ensures the traffic is **forwarded internally** to the **correct VM**. However, if I run Wireshark inside the Kali VM, I would only see traffic related to the Kali VM and not from other VMs. **Challenges faced:** 1.It was not immediately clear why all traffic was appearing to come to the **physical IP** rather than individual VMs. This behavior is expected because **NAT hides the internal network** details.  2.It took me a while to understand how **NAT Port Forwarding** works and how it forwards traffic to the correct VM behind the firewall. |
| Open Wireshark in Kali and your Virtual Windows machine, why are all the incoming connections from your classmates are the same everywhere? | When I opened Wireshark on both my **Kali** and **Windows** virtual machines, I noticed that **all the incoming connections from my classmates were identical** across both VMs. This happens because of **NAT (Network Address Translation)**. NAT essentially translates all **incoming traffic** destined for my VMs to look like it is coming from my **physical host IP** address.  NAT **masks the original source IP** and replaces it with the **host IP**. This causes all traffic, regardless of which virtual machine it is supposed to go to, to look like it's coming from the **same source** — my physical IP. Even though the traffic is forwarded to different VMs (like Kali, Windows, or Metasploitable), the original source IP address appears the same because the **NAT gateway** is handling the routing. **Challenges faced:** 1.At first, I was confused as to why all incoming connections appeared identical in **Wireshark** across different VMs. The solution to this issue is understanding how **source IP** addresses are rewritten by NAT.  2.I also had to research the **differences between SNAT (Source NAT)** and **DNAT (Destination NAT)** to fully grasp how NAT handles these connections. |
| Is it possible to use this, for malicious purposes? Explain how. | Yes, using **NAT port forwarding** in a scenario like this can be **exploited for malicious purposes**. By exposing internal services running inside virtual machines to the external network via **port forwarding**, an attacker can gain access to those services. For example:  **Metasploitable2** has known vulnerabilities in its **Apache HTTP** service, **FTP**, and **SSH**. If port forwarding is set up, an attacker can directly target these services from the **internet**, bypassing the NAT firewall.  Similarly, **Windows RDP** can be exposed using port forwarding (Host Port: **3389**), making it vulnerable to **RDP brute force attacks** or other **remote exploits**.  When **port forwarding** is enabled, even if the VMs are not publicly accessible, attackers can access these services as if they were running on the **public-facing host IP**. This can lead to:  **Unauthorized access** to internal systems via exposed services.  **Exploitation of known vulnerabilities** in the exposed services (Metasploitable2 is intentionally vulnerable for penetration testing).  **Denial of Service (DoS)** attacks against the services. **Challenges faced:** 1.I was initially unaware of how easily internal services could be exposed to the outside world by using **port forwarding**. It was an eye-opener to realize how seemingly **secure internal services** could become **vulnerable** if port forwarding was enabled without proper caution.  2.This also highlighted the importance of **securing network services** and ensuring that only trusted traffic is allowed via port forwarding. |

1. Delete all the forwarded ports you created for this exercise.

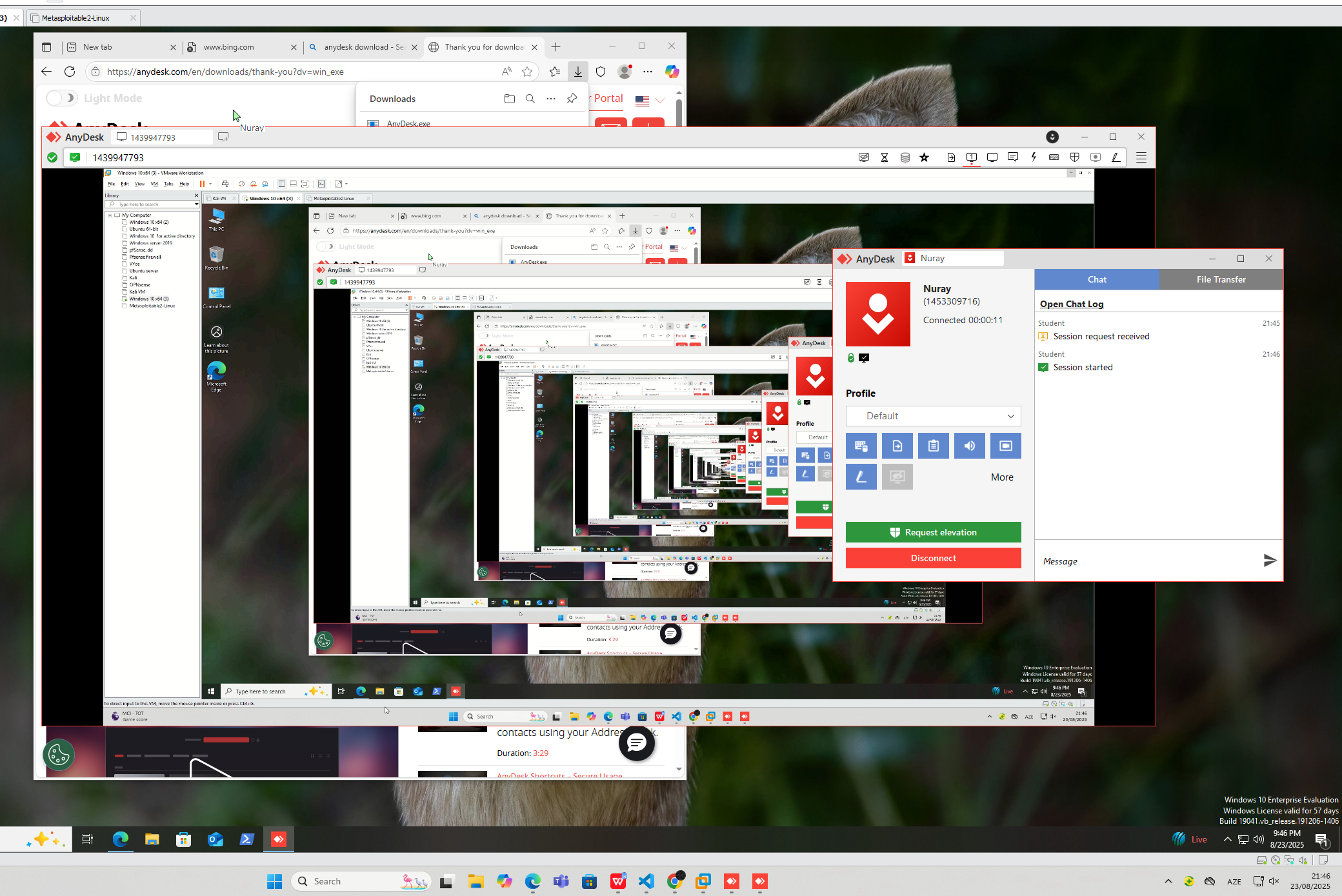




**Activity 3 NAT Transversal (20%)**

For this activity we are going to use the Windows virtual machine and your Windows Physical Machine.

1. Download Anydesk executable <https://anydesk.com/en/downloads/windows>
2. Run the program in the Windows Virtual and physical machines, create one invitation in your virtual machine using the address that the physical machine app shows you with Full profile.
3. Check your physical machine app to see the connection and start the connection.
4. Answer the following questions



|  |  |
| --- | --- |
| Question | Answer |
| What is the primary purpose of AnyDesk, and how does remote connection work between two devices? | The primary purpose of AnyDesk is to provide remote desktop access to another device over the internet. The remote connection works by entering the unique **AnyDesk ID** of the target device into the AnyDesk app. The device providing access must either grant permission manually or be set to allow remote access, depending on the configuration. Once the connection is established, the user on the remote device can control the target device as if they were physically sitting in front of it. |
| What are the security risks associated with using AnyDesk in uncontrolled environments? | Using AnyDesk in environments where things aren’t well-controlled does come with some security risks. The biggest concern is **unauthorized access**. Without a **password** or **two-factor authentication**, someone could easily connect to your system and control it. Another danger is **phishing** or **social engineering**, where malicious actors trick you into letting them in. There’s also the possibility of **malware** being transferred or installed while the remote session is ongoing. From my experience, it's always important to **ensure strong passwords** and enable **two-factor authentication** to keep everything safe. |
| How can AnyDesk be configured to require authorization before allowing remote access? | I’ve found it super easy to configure AnyDesk to require authorization before someone can access my machine. In **Settings → Security**, I simply enabled the option **"Always require confirmation"**. This means that every time someone tries to connect, I’ll be asked to approve the session, so I’m in control of who gets to access my system. It’s reassuring to know that I can’t be accessed without my permission. |
| What privacy options does AnyDesk offer to hide the cursor on the target side? | I really appreciate that AnyDesk offers **privacy features** to ensure I can work discreetly. One of my favorite features is the option to **hide the cursor** on the target side. This means that while I’m working remotely, the person on the other side can’t see my cursor moving around their screen, which gives me a sense of control and privacy. To enable it, I just go to **Settings → Privacy** and check the **"Hide cursor"** option. It’s a small touch, but it makes a big difference in feeling secure during remote work. |
| How can AnyDesk be configured to limit access to certain features, such as file transfer or full control? | AnyDesk allows me to limit what the remote user can do, which is so important for keeping things secure. I can easily control access to features like **file transfer**, **clipboard access**, or even **full control**. To do this, I just go to **Settings to Security**, where I can **uncheck** options I don’t want the remote user to have access to. It’s great knowing that I can **restrict permissions** and only allow the features I’m comfortable with. |
| What is the potential misuse of AnyDesk in social engineering or phishing attacks? | AnyDesk can be **misused** in **social engineering** or **phishing attacks**. Imagine someone impersonates a trusted support agent and convinces you to install AnyDesk and give them access to your computer. Once they’re in, they can steal sensitive information, install malicious software, or even lock you out. This is a serious concern, especially if you’re not aware of the risks. It’s crucial to always verify who’s asking for remote access and make sure you trust them before granting access. |
| How can AnyDesk be secured against unauthorized access using strong passwords or two-factor authentication? | Securing AnyDesk is easy if you use **strong passwords** and **two-factor authentication (2FA)**. In **Settings → Security**, I can enable **2FA**, which adds an extra layer of protection, making it harder for anyone to gain unauthorized access. Additionally, I make sure to set a **strong password** to prevent anyone from logging in without my permission. It’s so important to take these steps because it adds a lot of security to the remote access process. |
| What logs does AnyDesk generate, and how can they be used in a security investigation? | AnyDesk generates **logs** that can be really helpful in case of a security investigation. These logs include **connection details** like timestamps, the user who accessed the machine, and the duration of the session. I can find these logs in the **AnyDesk logs directory** on my system. If anything suspicious happens, these logs can be used to track the **origin of the connection** and figure out if there was any unauthorized access or malicious activity. |
| How can auto-start be disabled in AnyDesk to enhance device security? | To make my device more secure, I disabled **auto-start** for AnyDesk. By doing this, I make sure that **AnyDesk doesn’t start automatically** when my system boots up. This is really useful if I don’t want anyone accessing my device remotely without my explicit permission. I simply go to **Settings to General**, and I uncheck **Start AnyDesk with Windows**. This way, I have to manually open AnyDesk when I want to use it. |
| What cybersecurity best practices should be followed when using remote access tools like AnyDesk? | When I use remote access tools like AnyDesk, here’s what I always keep in mind to make sure everything stays secure:  **Two-factor authentication (2FA) is a must**: It’s simple, but so important. It adds an extra layer of protection that makes it way harder for unauthorized people to access my devices.  **Always verify the person requesting access**: You never know who’s on the other end. Sometimes, it’s easy to be tricked by phishing or social engineering attacks. That’s why I make sure I know who’s asking to connect before giving them access.  **Strong passwords** are key: It’s tempting to just use simple passwords or default ones, but I know they’re easy to guess. I make sure to use unique, complex passwords for every device.  **Keep AnyDesk updated**: I always check for updates because sometimes, new vulnerabilities are found and patched. I don’t want to be exposed to any old weaknesses in the software.  **Only allow trusted devices**: I use **whitelisting** to make sure that only the devices I trust can connect to my system. This way, I can be sure that no one is sneaking in.  **Keep an eye on the logs**: I regularly check the connection logs. It’s a good habit to watch for any unusual activity or access from unfamiliar IPs. If something feels off, I’ll know immediately. |