Research Project Progress Report

Capstone Design Project I

Submitted by
Md. Abdul Halim Zayed
Student ID: 222 206 038

Submitted to
Prof. Dr. Shahidul Islam Khan
Head & Dean
(Supervisor)
&
Sheikh Hasanul Banna
Lecturer
(Co-Supervisor)

Department of Computer Science & Engineering
Presidency University

Tentative Project Name:

Network Attack Simulation Using GNS3: A Study on ARP Spoofing and MAC Flooding.

1. Abstract:

This project focuses on creating a complete virtual network environment using GNS3 and VirtualBox to prepare for studying network attacks such as ARP spoofing and MAC flooding. In this first phase, the goal was to build the network, configure all devices, and make sure every part of the system communicates properly. Cisco routers, Kali Linux, and multiple virtual PCs were used to form two separate LANs that can interact with each other. Static routing, DHCP services, and basic connectivity testing were successfully done, and all devices received correct IP addresses and responded to ping commands across both networks. This stable setup now provides the foundation needed to simulate attacks and analyze their effects in the next phase of the project, Capstone Design 2.

2. Introduction:

Computer networks are an essential part of modern communication, and understanding how they work along with how they can be attacked is a key skill for any network engineer. This project is designed to explore two common network attacks, **ARP spoofing** and **MAC flooding**, by building a complete virtual network environment and preparing it for security testing in the next phase.

a. Network Attacks we'll Study: In this project, the main focus is on understanding three important network concepts related to LAN-based attacks. These are **ARP**, **ARP spoofing**, and **MAC flooding**. ARP is used inside a local network to match IP addresses with MAC addresses, and attackers often take advantage of this process. ARP spoofing happens when a fake MAC address is sent to a device to trick it, while MAC flooding targets the switch to overload its MAC table. Studying these helps us understand how network traffic can be hijacked or exposed inside a LAN.

b. What is GNS3? why GNS3 is important?

GNS3 is a free tool that lets you design and test virtual network setups easily. GNS3 is used in this project because it gives a very realistic network environment without needing real hardware. It lets you add routers, switches, PCs, servers, and even full operating systems like Kali Linux. All of this can be done on one computer. This makes the tool cost-effective for students and flexible enough to build any type of network. It also allows you to test routing,

IP addressing, connectivity, packet flow, and even security attacks safely.

c. Why Researchers Prefer GNS3?

Researchers prefer GNS3 because it supports real Cisco images, offers packet-capture options, and gives full control over a network. They can isolate the virtual lab, repeat experiments easily, and integrate real and virtual networks together. It is a trusted platform for testing new protocols, simulating attacks, and analyzing how devices behave without damaging real equipment.

- **Cost-Effective and Accessible Testbed**
  Setting up a physical lab with routers, switches, and firewalls for a research project can be extremely expensive. GNS3 eliminates this barrier by allowing researchers to build complex topologies with a wide range of virtual devices on a standard computer.

- **Realistic and Accurate Simulation**
  A major advantage of GNS3 is its use of real network device operating systems (like Cisco IOS, Juniper Junos, etc.) within the emulation. Researchers can:

  - ✓ Test new protocols
  - ✓ Analyze network behavior
  - ✓ Verify theoretical models

- **Ability to Create a Controlled and Isolated Environment**
  Research often requires a controlled environment where a network's behavior can be meticulously observed and analyzed. GNS3 provides this by:
  - ✓ Isolating the network
  - ✓ Packet Capture and Analysis
  - ✓ Reproducibility

- **Hybrid Network Integration**
  GNS3 allows researchers to connect their virtual labs to real-world networks or other virtual machines. This capability is particularly useful for:
  - ✓ Testing real-world interoperability
  - ✓ Simulating specific attacks
  - ✓ Developing and testing applications

- **Multi-Vendor Interoperability**
  Researchers often need to study network environments that contain devices from multiple vendors. GNS3's multi-vendor support enables them to build these complex, heterogeneous networks, allowing for the

study of interoperability issues, multi-vendor security, and the effectiveness of different network solutions.

d. What is ARP?

ARP, or Address Resolution Protocol, is a networking protocol that dynamically maps an Internet Protocol (IP) address to a physical hardware address, also known as a Media Access Control (MAC) address. This translation is essential for devices to communicate with each other on the same local network (LAN).

e. What is ARP spoofing?

ARP spoofing, also known as ARP poisoning, is a cyberattack that exploits a vulnerability in the Address Resolution Protocol (ARP). An attacker sends fake ARP messages onto a local network to trick devices into associating the attacker's MAC address with the IP address of a legitimate device.

f. What is Mac flooding?

MAC flooding, also known as a MAC table overflow attack, is a network security attack that targets a switch's limited memory. The goal is to overwhelm the switch's MAC address table (also called a Content Addressable Memory or CAM table) to force it to act like a network hub, allowing the attacker to intercept network traffic.

3. Tools and Technologies:

GNS3: GNS3 is the main platform used to build and simulate the entire network topology for this project. It allows routers, switches, and virtual machines to run together in one environment. Because it supports real Cisco router images and provides accurate packet flow, it gives a realistic lab setup without any physical hardware.

VirtualBox: VirtualBox is used to run the virtual machines needed in the network. In this project, the GNS3 VM and the Kali Linux VM were both created and managed through VirtualBox. This lets the machines communicate with the routers inside GNS3 just like in a real network.

Kali Linux: Kali Linux acts as a workstation in the network and is also used for configuration tasks. It was used to assign IP addresses, test connectivity, check ARP tables, and set up the DHCP server. In the next phase of the project, Kali will also be used to perform security attacks such as ARP spoofing and MAC flooding.

Cisco Router Images: Real Cisco router images were added to GNS3 to create the routers in the topology. Using these images gives more accurate

behavior because the routers respond exactly like physical Cisco devices. This helps make the routing and network testing more reliable.

4. Network Design and Implementation:

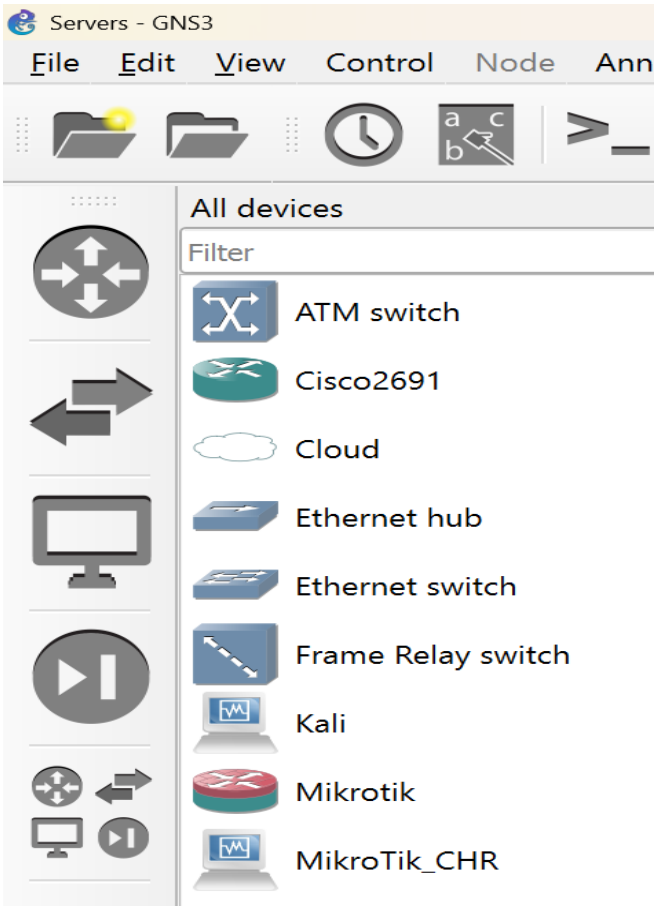
4.1: Adding Images and Setting Up the Environment:

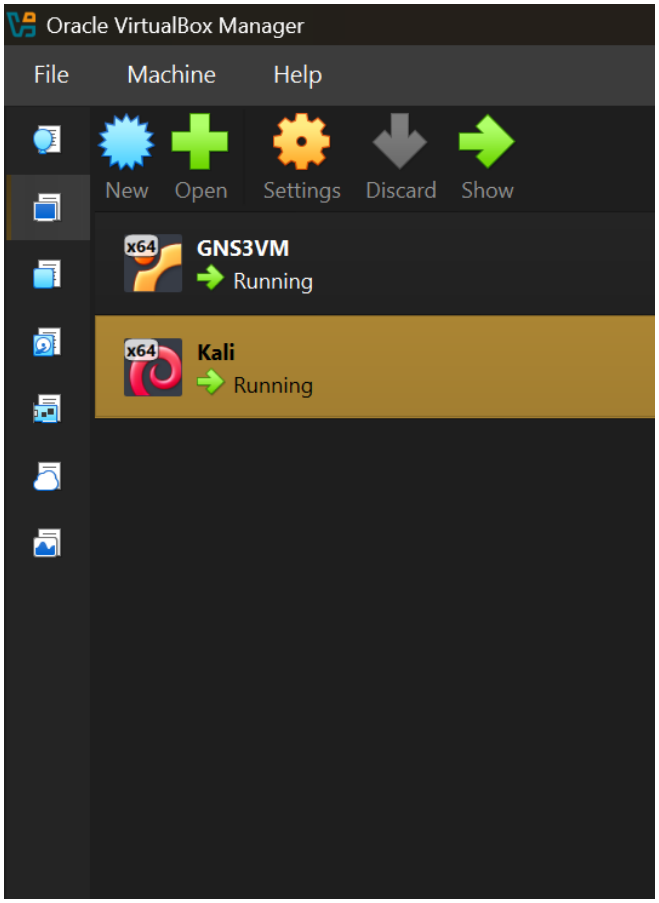
The first task in this project was preparing the simulation environment by adding all the required images and virtual machines into GNS3. To begin, a **Cisco router image** was imported into GNS3 so that real router functionality could be used inside the network. This made it possible to work with Router1 (R1) and Router2 (R2) just like real Cisco devices.

After that, **VirtualBox** was installed and the **GNS3 VM** was added inside it. Once the GNS3 VM was set up, it was linked with the GNS3 application. This integration helped GNS3 run devices more smoothly by using the resources of the GNS3 VM.

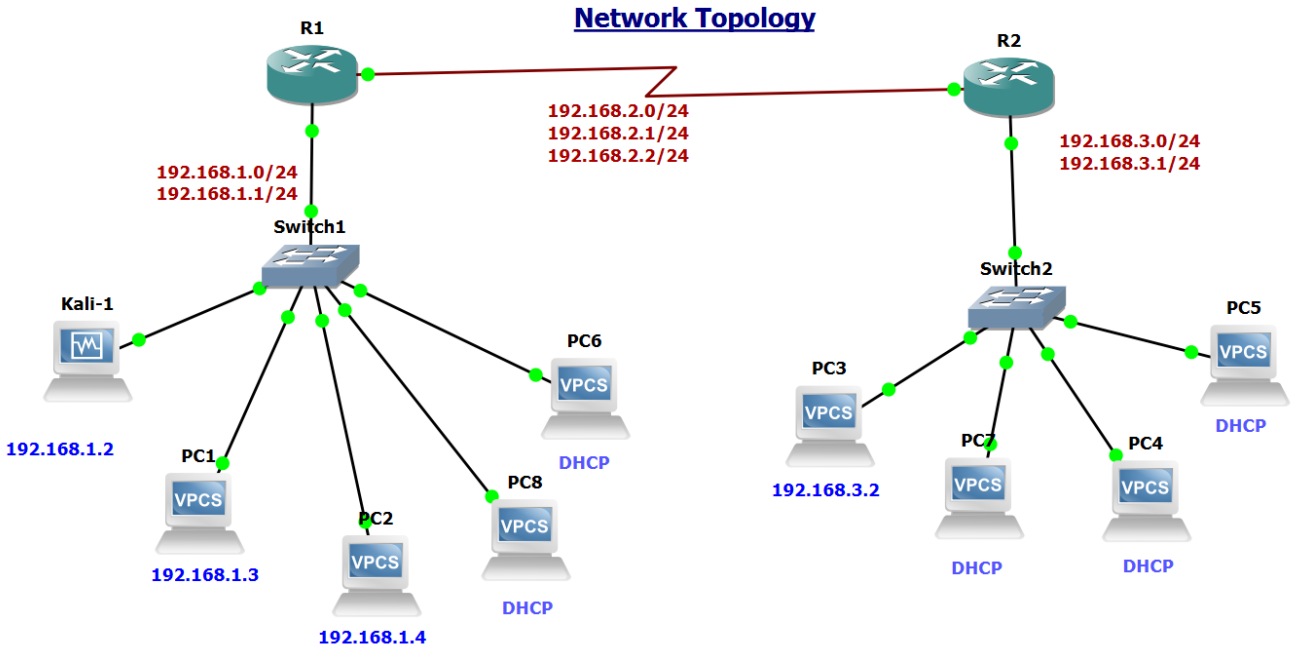
Next, a **Kali Linux** virtual machine was created in VirtualBox. This Kali VM was then imported into GNS3 as a node, and it appeared in the workspace as **Kali-1**. This machine was later used for DHCP configuration and will be used for security attack simulations in the next phase.

With the Cisco router image, the GNS3 VM, and the Kali Linux VM all added and running properly, the environment was ready to build the complete topology.





4.2 Network Topology Overview:



Once the environment was configured, the full network topology was created in GNS3. The topology consists of two LANs connected through two routers, R1 and R2.

On the **left side (LAN1 – 192.168.1.0/24)**:

- **R1** is the gateway with IP **192.168.1.1**
- R1 is connected to **Switch1**, which connects to:
    - **Kali-1** — 192.168.1.2
    - **PC1** — 192.168.1.3
    - **PC2** — 192.168.1.4

On the **right side (LAN2 – 192.168.3.0/24)**:

- **R2** is the gateway with IP **192.168.3.1**
- R2 is connected to **Switch2**, which connects to:
    - **PC3** — 192.168.3.2
    - **PC4** — Through DHCP
    - **PC5** — Through DHCP

R1 and R2 are linked through a **point-to-point network (192.168.2.0/24)**:

- R1: 192.168.2.1
- R2: 192.168.2.2

This link allows traffic to flow between LAN1 and LAN2 using static routing.

Step 1: Network Setup and Basic Configuration:

In the first phase of the project, the entire network simulation environment was prepared using **GNS3** integrated with **VirtualBox**. This phase focused on importing and configuring virtual devices, establishing connectivity between the virtual machines and routers, and laying the groundwork for the overall topology.

1. GNS3 and VirtualBox Integration

To build a virtualized network, GNS3 was connected with VirtualBox, ensuring smooth operation of all virtual machines inside the topology.

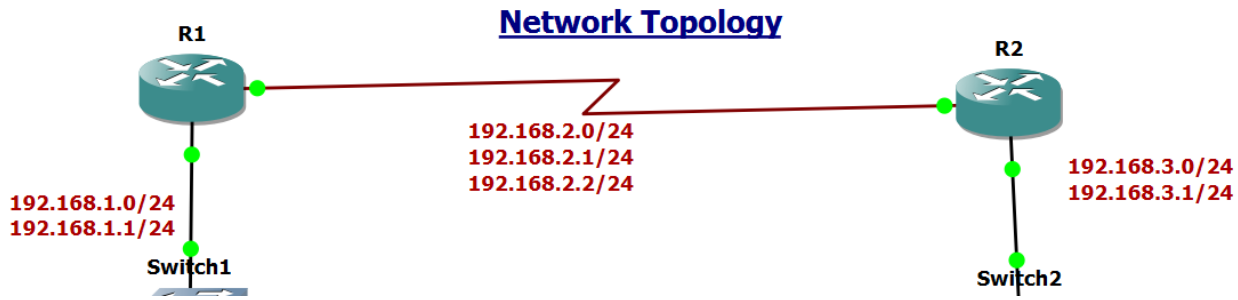
- The **GNS3 VM** was successfully imported into VirtualBox and linked to the GNS3 application.
- The **Kali Linux VM** was also imported into VirtualBox and added to GNS3 as a PC node (Kali-1).

- Network adapters of Kali Linux in VirtualBox were correctly configured, allowing it to communicate with routers and switches inside GNS3.

2. Router Configuration

Two Cisco routers (Router1 and Router2) were added to the topology using the imported Cisco IOS image.

- Router1 was configured with the IP **192.168.1.1/24** for its LAN interface.
- Router2 was configured with **192.168.3.1/24** for its LAN interface.
- A point-to-point link was created between the routers using the **192.168.2.0/24** network.
- Static routing was configured on both routers to enable communication between LAN1 and LAN2.




```
R1#
R1#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#interface serial1/0
R1(config-if)#ip address 192.168.2.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#
*Mar  1 00:02:39.707: %LINK-3-UPDOWN: Interface Serial1/0, changed state to up
R1(config)#
*Mar  1 00:02:40.711: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed state to up
R1(config)#interface
*Mar  1 00:03:02.787: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed state to down
R1(config)#interface fa0/o
                         ^
% Invalid input detected at '^' marker.

R1(config)#interface fa0/0
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
*Mar  1 00:03:58.623: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Mar  1 00:03:59.623: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1(config-if)#exit
R1(config)#
```

3. PC Configuration

PC nodes on the LAN were configured as follows:

- **PC1** received a DHCP-assigned address (**192.168.1.3**) statically.
- **PC8** received a DHCP-assigned address (**192.168.1.50**) from the DHCP server later configured on Kali Linux.
- **Kali Linux (Kali-1)** was manually assigned **192.168.1.2** and configured with the appropriate default gateway (**192.168.1.1**).
- Router interfaces were all assigned static IP addresses to ensure stable routing between the networks.

```
PC1> ip 192.168.1.3/24 192.168.1.1
Checking for duplicate address...
PC1 : 192.168.1.3 255.255.255.0 gateway 192.168.1.1
```

```
PC8> ip dhcp
DDORA IP 192.168.1.54/24 GW 192.168.1.1
```

Step 2: Static IP Configuration and Connectivity Testing:

After completing the basic network setup, the second step involved verifying device communication, routing, and ARP resolution. This ensured that the network was functioning as expected before implementing dynamic IP allocation and advanced mechanisms.

1. Static IP Configuration on Router Interfaces

Both routers were configured with static IP addresses:

- Router1: **192.168.1.1** (LAN), **192.168.2.1** (inter-router link)
- Router2: **192.168.3.1** (LAN), **192.168.2.2** (inter-router link)

Configuration command of static routing:

For R1: (ip route 192.168.3.0 255.255.255.0 192.168.2.2)

For R2: (ip route 192.168.1.0 255.255.255.0 192.168.2.1)

These configurations ensured proper forwarding of packets across the two LANs.

2. Routing Verification:

Routing verification was completed by:

- Adding static routes on both routers
- Running routing table checks using command: (***show ip route***)
- Verifying end-to-end communication between LAN1 and LAN2

```
R1#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    192.168.1.0/24 is directly connected, FastEthernet0/0
C    192.168.2.0/24 is directly connected, Serial1/1
S    192.168.3.0/24 [1/0] via 192.168.2.2
R1#
```

```
R2#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

S    192.168.1.0/24 [1/0] via 192.168.2.1
C    192.168.2.0/24 is directly connected, Serial1/1
C    192.168.3.0/24 is directly connected, FastEthernet0/0
R2#
```

3. Ping and ARP Testing

To verify connectivity:

- Ping tests were performed from **Kali Linux (192.168.1.2)** to all other devices including PC1, PC2, Router1, and Router2.
- The ARP cache on Kali was checked using command in Linux: (***arp -a***)

This displayed the IP-to-MAC mappings of all devices, confirming proper ARP functionality.

Ping test:

```
PC3> ping 192.168.1.3
192.168.1.3 icmp_seq=1 timeout
84 bytes from 192.168.1.3 icmp_seq=2 ttl=62 time=24.455 ms
84 bytes from 192.168.1.3 icmp_seq=3 ttl=62 time=22.495 ms
84 bytes from 192.168.1.3 icmp_seq=4 ttl=62 time=22.567 ms
84 bytes from 192.168.1.3 icmp_seq=5 ttl=62 time=23.383 ms

PC3> ping 192.168.1.3
84 bytes from 192.168.1.3 icmp_seq=1 ttl=62 time=22.812 ms
84 bytes from 192.168.1.3 icmp_seq=2 ttl=62 time=25.007 ms
84 bytes from 192.168.1.3 icmp_seq=3 ttl=62 time=31.814 ms
84 bytes from 192.168.1.3 icmp_seq=4 ttl=62 time=26.004 ms
84 bytes from 192.168.1.3 icmp_seq=5 ttl=62 time=24.901 ms

PC3>
```

```
zayed@kali: ~
Session  Actions  Edit  View  Help
192.168.4.0/24 via 192.168.1.1 dev eth0 proto static metric 104
   Internet should work via eth1, GNS3 via eth0.

┌──(zayed㉿kali)-[~]
└─$ ping -c 4 192.168.3.2
PING 192.168.3.2 (192.168.3.2) 56(84) bytes of data.
64 bytes from 192.168.3.2: icmp_seq=1 ttl=62 time=3042 ms
64 bytes from 192.168.3.2: icmp_seq=2 ttl=62 time=2019 ms
64 bytes from 192.168.3.2: icmp_seq=3 ttl=62 time=1009 ms
64 bytes from 192.168.3.2: icmp_seq=4 ttl=62 time=30.7 ms

--- 192.168.3.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3054ms
rtt min/avg/max/mdev = 30.692/1525.079/3041.994/1123.028 ms, pipe 3
```

ARP test:

```
┌──(zayed㉿kali)-[~]
└─$ ping -c 4 192.168.1.3
PING 192.168.1.3 (192.168.1.3) 56(84) bytes of data.
64 bytes from 192.168.1.3: icmp_seq=1 ttl=64 time=2.64 ms
64 bytes from 192.168.1.3: icmp_seq=2 ttl=64 time=1.37 ms
64 bytes from 192.168.1.3: icmp_seq=3 ttl=64 time=1.54 ms
64 bytes from 192.168.1.3: icmp_seq=4 ttl=64 time=1.36 ms

--- 192.168.1.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 1.358/1.725/2.635/0.530 ms

┌──(zayed㉿kali)-[~]
└─$ arp -a
? (10.0.3.2) at 52:55:0a:00:03:02 [ether] on eth1
? (192.168.3.2) at c0:01:3a:54:00:00 [ether] on eth0
? (192.168.1.3) at 00:50:79:66:68:04 [ether] on eth0

┌──(zayed㉿kali)-[~]
└─$
```

4.3 DHCP Server Setup and Configuration

In this phase of the project, a DHCP server was implemented on Kali Linux to automate the distribution of IP addresses for both LAN1 and LAN2. The objective was to eliminate the need for manual IP assignment and to enable dynamic IP allocation based on predefined ranges. This configuration also played a critical role in ensuring that the network behaved realistically, similar to an actual enterprise environment.

4.3.1 Installing the DHCP Server on Kali Linux

The DHCP service used in this project is the **ISC DHCP Server**, a widely used open-source DHCP daemon. It was installed on Kali Linux using the following command:

sudo apt install isc-dhcp-server -y

After installation, the service files and configuration directory were created in

/etc/dhcp/

4.3.2 Configuring the DHCP Pools (dhcpd.conf):

The most important part of the DHCP setup is editing the file:

/etc/dhcp/dhcpd.conf

Two separate subnets were defined:

1. DHCP Pool for LAN1 (192.168.1.0/24) and LAN2 (192.168.3.0/24):

```
# LAN1 DHCP Pool (192.168.1.50- 192.168.1.1)
subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.50 192.168.1.100;           # DHCP pool
  option routers 192.168.1.1;                 # Default gateway
  option domain-name-servers 8.8.8.8;         # DNS server (Google DNS)
}

subnet 192.168.3.0 netmask 255.255.255.0 {
range 192.168.3.50 192.168.3.100;
option routers 192.168.3.1;
option domain-name-servers 8.8.8.8;
}

#log-facility local7;
```

Explanation

- For LAN1, IP addresses between **192.168.1.50 – 192.168.1.100** and for LAN2, IP addresses between **192.168.3.50 – 192.168.3.100** are dynamically assigned
- Default gateway: **192.168.1.1 (Router1), 192.168.3.1 (Router2)**
- DNS used: Google DNS (**8.8.8.8**)

4.3.3 Configuring DHCP Relay on Router2:

Because the DHCP server is located in LAN1, PCs in LAN2 cannot directly communicate with the DHCP server (since DHCP uses broadcast traffic). To solve this, **Router2 was configured as a DHCP Relay Agent**.

The following command was executed on R2:

R2(config)# ***interface FastEthernet0/0***
R2(config-if)# ***ip helper-address 192.168.1.2***

Purpose of ip helper-address

- Converts DHCP **broadcasts → unicasts**
- Forwards them to the DHCP server (Kali Linux: **192.168.1.2**)
- Enables LAN2 clients to get IPs from LAN1's DHCP server

4.3.4 Verifying DHCP Operation on LAN1 and LAN2:

After enabling the DHCP server and relay:
1. LAN1 Verification

PC1 and other devices on LAN1 were set to DHCP mode and received addresses in the range **192.168.1.50 – 192.168.1.100**.

Command used on PC nodes:

ip dhcp

```
PC6> show ip

NAME          : PC6[1]
IP/MASK       : 0.0.0.0/0
GATEWAY       : 0.0.0.0
DNS           :
MAC           : 00:50:79:66:68:05
LPORT         : 10044
RHOST:PORT    : 127.0.0.1:10045
MTU:          : 1500

PC6> ip dhcp
DDORA IP 192.168.1.51/24 GW 192.168.1.1

PC6>
```

2. LAN2 Verification
PCs in LAN2 received IP addresses from the **192.168.3.x** DHCP pool through the relay agent.
Command used on PC nodes:

ip dhcp

```
PC4> show ip

NAME          : PC4[1]
IP/MASK       : 0.0.0.0/0
GATEWAY       : 0.0.0.0
DNS           :
MAC           : 00:50:79:66:68:06
LPORT         : 10038
RHOST:PORT    : 127.0.0.1:10039
MTU:          : 1500

PC4> ip dhcp
DDORA IP 192.168.3.52/24 GW 192.168.3.1

PC4>
```

5. Conclusion

In this phase of the project, a complete and stable virtual network environment was successfully designed and implemented using GNS3, Cisco routers, switches, and VirtualBox-based virtual machines. All core networking components—such as static routing, inter-LAN connectivity, ARP operations, DHCP automation, and DHCP relay—were configured and verified through extensive testing. The network now operates exactly like a real multi-LAN setup with proper communication between devices, dynamic IP allocation, and accurate packet forwarding across routers. With this solid foundation in place, the system is fully prepared for the next phase of the project, where ARP spoofing and MAC flooding attacks will be simulated and analyzed in a controlled environment.

Key Achievements of Capstone Design – 1

- A fully functional dual-LAN network topology was built using real Cisco IOS images and GNS3 simulation.
- Static routing was implemented on both routers, enabling successful communication between LAN1 and LAN2.
- A DHCP server was configured on Kali Linux, and a DHCP relay agent was set up on Router2 to support dynamic IP allocation across both LANs.
- Connectivity, ARP mapping, and routing tests were performed to validate end-to-end communication and ensure correct packet forwarding.
- A stable and realistic testbed was prepared, creating the required platform for performing ARP spoofing and MAC flooding attacks in Capstone Design – 2.

6. Problem: Internet Connectivity Conflict in Kali Linux:

During the project, we faced a major challenge related to Internet connectivity in the Kali Linux virtual machine while it was connected to the GNS3 network. The Kali VM was configured with the IP address 192.168.1.2 on interface eth0, and the default gateway was set to 192.168.1.1, which is the GNS3 router

(R1). At the same time, a second network adapter (Adapter 2) was enabled in VirtualBox and attached to NAT to provide Internet access.

However, even after enabling the NAT interface, the Kali machine could not reach the Internet. Pinging external IPs such as 8.8.8.8 resulted in "Destination Host Unreachable". The root cause of this problem was that the default gateway of the system was still pointing to 192.168.1.1 (the GNS3 router), which has no route to the real Internet. As a result, all external traffic was being sent towards the GNS3 gateway instead of the VirtualBox NAT gateway.

- **Temporary Workaround**

As a temporary solution, we blocked or disabled the GNS3-facing interface (eth0) inside Kali and allowed only the NAT interface (eth1) to remain active as the default route. In this state, Internet access worked correctly, but Kali could no longer communicate with the GNS3 topology, which was not suitable for our experiment.

- **Final Approach**

To solve this issue in a more permanent and structured way, we adopted a dual-interface routing strategy:

1. **Configured eth0 with a static IP (192.168.1.2/24) for GNS3, but without any default gateway.**
2. **Configured eth1 (VirtualBox NAT) to receive its IP and default gateway via DHCP, and kept it as the only default route for the system.**
3. **Added specific routes for the GNS3 network subnets through eth0, so that traffic for 192.168.1.0/24, 192.168.2.0/24, and 192.168.3.0/24 goes via the GNS3 router, while all other traffic (e.g., Internet) goes through the NAT gateway.**

This configuration allowed Kali Linux to:

- Communicate with all devices inside the GNS3 topology, and
- Access the Internet at the same time (e.g., successfully pinging 8.8.8.8 and browsing external resources).

Future Works (Capstone Project Design – II)

1. ARP Spoofing Attack Simulation

- Perform ARP spoofing using tools such as arpspoof, ettercap, or Scapy.
- Observe how ARP tables of victims and routers get poisoned.

- Use Wireshark during the attack to monitor packet flow and verify MITM behavior.
- Compare ARP table entries before and after the attack.

2. MAC Flooding Attack Simulation

- Execute MAC flooding using macof or a custom Scapy script to overflow the switch CAM table.
- Use Wireshark to observe increased broadcast traffic when the switch behaves like a hub.
- Confirm whether the attacker can capture unicast traffic due to the flooding.

3. Packet Capture & Traffic Analysis

- Capture traffic before, during, and after both attacks using Wireshark.
- Analyze how packet behavior changes under ARP spoofing and MAC flooding conditions.

4. Detecting the Attacks

- Use ARP table inspection, Linux tools, and Wireshark filters to detect abnormal behavior.
- Identify suspicious ARP replies, duplicate MAC entries, or unusual broadcast patterns.

5. Mitigation / Prevention Techniques

- Demonstrate methods such as static ARP entries, Dynamic ARP Inspection, port security, and storm control.
- Use Wireshark to verify that traffic returns to normal after mitigation is applied.

6. Final Comparative Analysis

- Compare ARP spoofing vs MAC flooding based on attack method, impact, detection behavior, and Wireshark observations.
- Summarize how each attack affects the network differently.

Final Summary (Short Form):

In Capstone Design 2, the tasks will include:

1. **Performing ARP Spoofing attacks, analyzing their effects, detecting the attack behavior, and applying defense techniques.**
2. **Executing MAC Flooding attacks, capturing the traffic impact, analyzing the results, and demonstrating effective defense methods.**
3. **Conducting packet capture and traffic behavior analysis using Wireshark during both attacks.**
4. **Implementing security detection mechanisms to identify abnormal traffic patterns and ARP anomalies.**
5. **Explaining and demonstrating mitigation techniques to secure the network against ARP Spoofing and MAC Flooding.**