Layer 2 Security Lab 2

CYBR3010

Professor: Sam El’Awour

Gregory Stephens

September 21, 2025

Introduction: Layer 2 Security Attacks and Defenses

Modern networks are often assumed to be secure once they're behind a firewall or encrypted with TLS, but that's only half the story. The real vulnerabilities lie at Layer 2, where switches trust devices blindly, ARP replies are accepted without question, and MAC tables can be overwhelmed with ease. In this report, we explore these weaknesses through hands-on testing in a controlled lab environment designed to mirror real-world enterprise segments.
Our lab network consists of four key components:

- Kali Linux (192.168.1.10) – acting as the attacker node, connected via eth0/1
- Client20 (Windows, 192.168.1.20) and Client30 (Windows, 192.168.1.30) – two standard endpoints communicating over the LAN
- SWO1 (Cisco Switch, VLAN 1, 192.168.1.1) – the central Layer 2 switch managing all traffic between devices

All devices reside on the same broadcast domain (VLAN 1), making them prime targets for Layer 2 manipulation. This simple topology is intentionally flat to expose how easily trust-based protocols like ARP and MAC learning can be exploited even within a "trusted" internal network.
We begin by examining three critical Layer 2 vulnerabilities:

1. ARP Spoofing – the most insidious of the three, allowing an attacker to intercept, modify, or block traffic between any two hosts by poisoning their ARP caches.
2. MAC Spoofing – impersonating a legitimate device's hardware address to bypass access controls or steal identity-based trust.
3. MAC Flooding – overwhelming the switch's CAM table to force it into hub-like behavior, broadcasting all traffic to every port and enabling packet sniffing.

Each attack exploits fundamental design assumptions in Ethernet and switching: that ARP responses are truthful, MAC addresses are unique and immutable, and switches will always forward traffic intelligently. The impact? Complete traffic interception, denial of service, credential theft, and lateral movement all without touching IP-level defenses.

A detailed Visio diagram of the final network topology is included in Section 3, clearly labeling all devices, interfaces, IP addresses, and VLAN assignments for clarity and reproducibility.

Following the attack demonstrations, we provide step-by-step configuration steps for security measures, including port security, DHCP snooping, Dynamic ARP Inspection (DAI), and STP hardening to show how each vulnerability can be mitigated at the switch level. These aren't theoretical fixes; they're practical configurations tested and validated in our lab environment.

Test results are presented in clear before-and-after scenarios: we show traffic flow and packet captures from Wireshark and tcpdump before and after each security measure is applied, proving the effectiveness of the defenses. You'll see how ARP spoofing goes from seamless man-in-the-middle to completely blocked. How MAC flooding turns a switch into a broadcast

storm generator… then becomes harmless under port security. And how MAC spoofing is caught and shut down at the port level.

Finally, we address the key conceptual questions that tie this all together, covering VLAN Hopping, STP manipulation risks, Dynamic ARP Inspection, and how DHCP snooping, port security, and endpoint posture assessment form a cohesive Layer 2 defense strategy, each answered concisely.

This report doesn't just show how attacks work, it shows how to stop them. Because in the real world, attackers don't need to breach the firewall. They just need to plug into a wall jack.

Vulnerabilities and Potential Impacts

1. ARP Spoofing
What it is: An attacker tricks devices on a network into thinking the attacker's device is another device (like the router).
What it does: Now the attacker can secretly read, change, or block all traffic between devices, like spying on your emails or stealing passwords.
Why it's scary: You won't even know it's happening.

2. MAC Spoofing
What it is: An attacker pretends to be a trusted device by copying its unique hardware address (MAC address).
What it does: They can sneak past security that only allows known devices, like accessing a Wi-Fi network or bypassing network filters.
Why it's dangerous: It's like stealing someone's ID card to get into a restricted building.

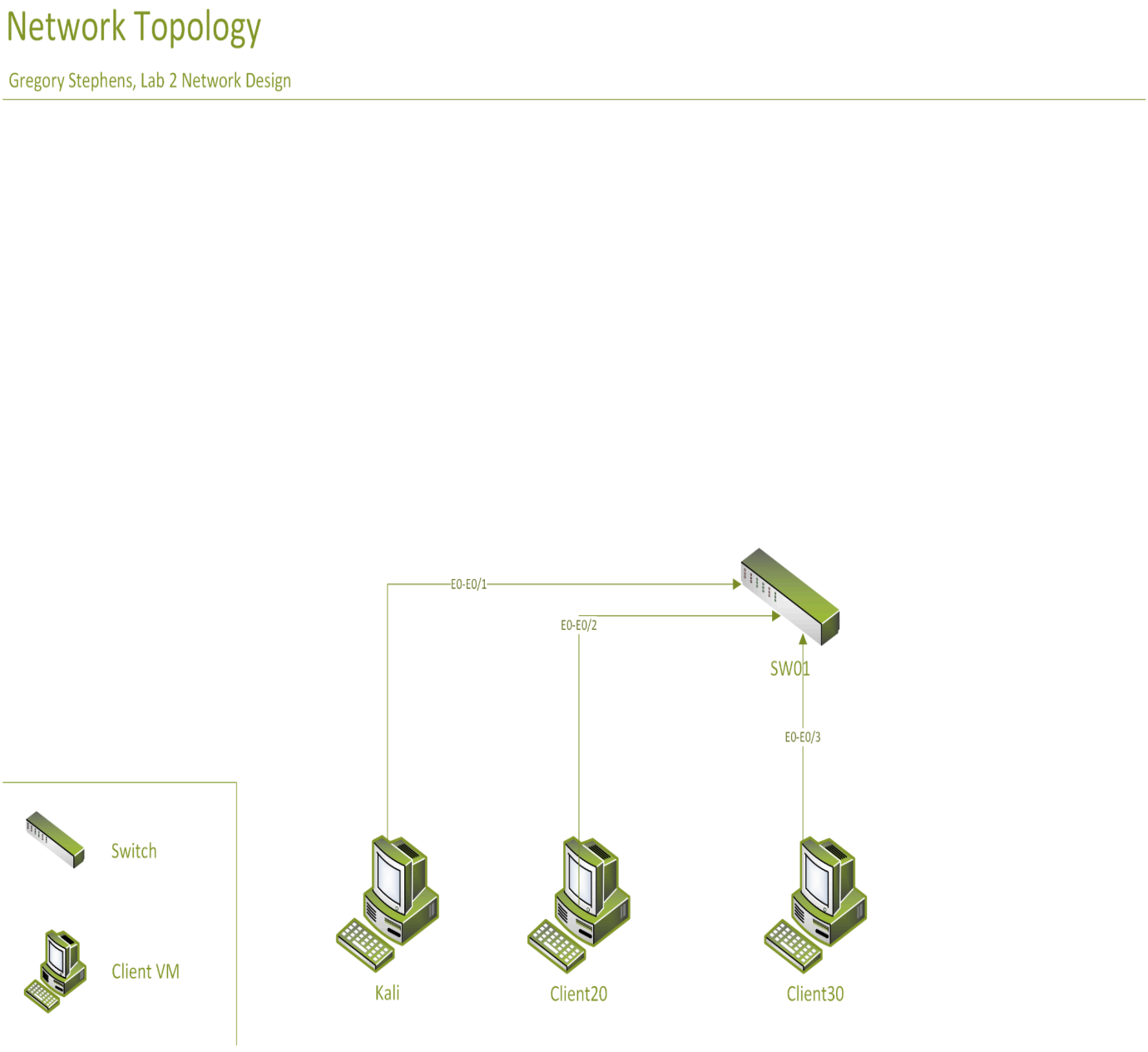
3. MAC Flooding
What it is: An attacker floods a network switch with fake MAC addresses until it gets overwhelmed.
What it does: The switch stops working properly and starts broadcasting all traffic to every device, like turning a private phone line into a loudspeaker.
Why it's risky: Now anyone on the network can sniff (eavesdrop on) all the traffic, including passwords and messages.

In short:

- ARP Spoofing = Spy on conversations
- MAC Spoofing = Fake your identity
- MAC Flooding = Turn the network into an open microphone

All three let attackers steal info or take control, without you noticing.

Network Diagram (Visio)



Attack 1 - ARP Spoofing (MITM Attack)

Devices Used:

- Kali Linux: eth0/1 → 192.68.1.10
- Client20 (Windows): eth0/2 → 192.168.1.20
- Client30 (Windows): eth0/3 → 192.168.1.30
- SWO1 (Cisco Switch): VLAN 1 → 192.168.1.1
- All connected to SWO1 via green Ethernet links
- All devices can ping each other before an attack.

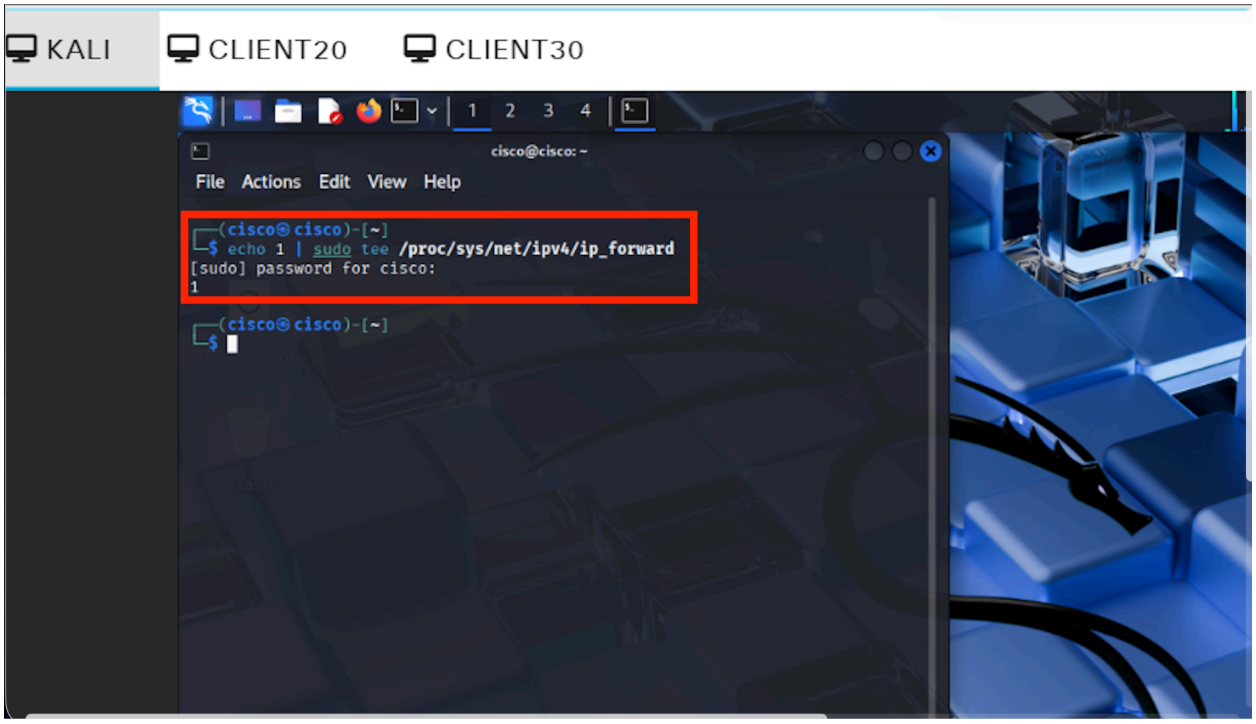
STEP 1: Enable IP Forwarding on Kali to Allow Traffic Forwarding (Prep for MITM)

Instructions:

1. Open the Kali Linux terminal.
2. Run the following command to enable IP forwarding:
   echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward

Expected Output: No output shown, command runs silently.

Why? Without this, Kali receives packets from Client20 meant for Client30, but drops them. Enabling forwarding makes Kali act as a router, allowing MITM.



IP Forwarding Enabled on Kali-Required for MITM

STEP 2: Launch ARP Spoofing to Poison Both Client20 and Client30

Instructions:

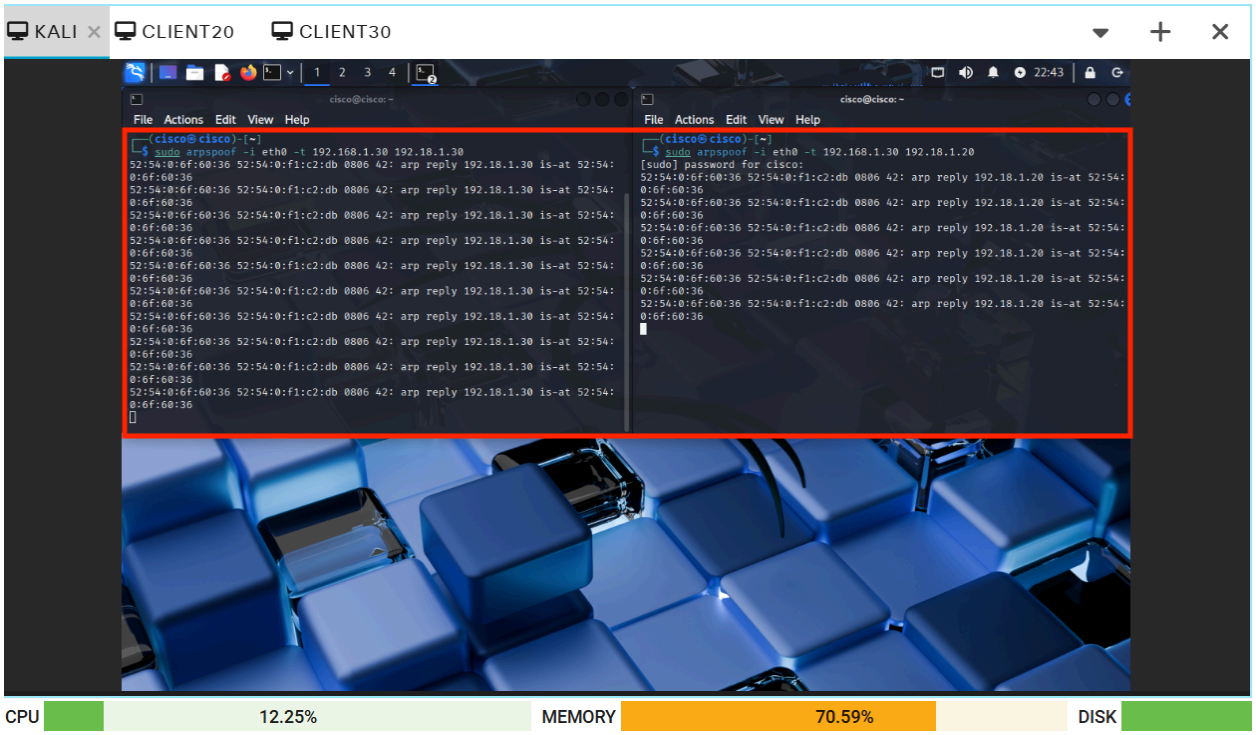
1. Open two separate terminal windows on Kali.
2. In Terminal 1, run:
   sudo arpspoof -i eth0 -t 192.168.1.20 192.168.1.30

3. In Terminal 2, run:
   sudo arpspoof -i eth0 -t 192.168.1.30 192.168.1.20

4. Wait 5–10 seconds — you'll see rapid ARP replies scrolling.

Expected Output:

The MAC shown is Kali's, not the real clients', this is the attack.



ARP Spoofing Active_Client20_Client30

STEP 3: Capture and Confirm Intercepted Traffic Between Clients

Instructions:

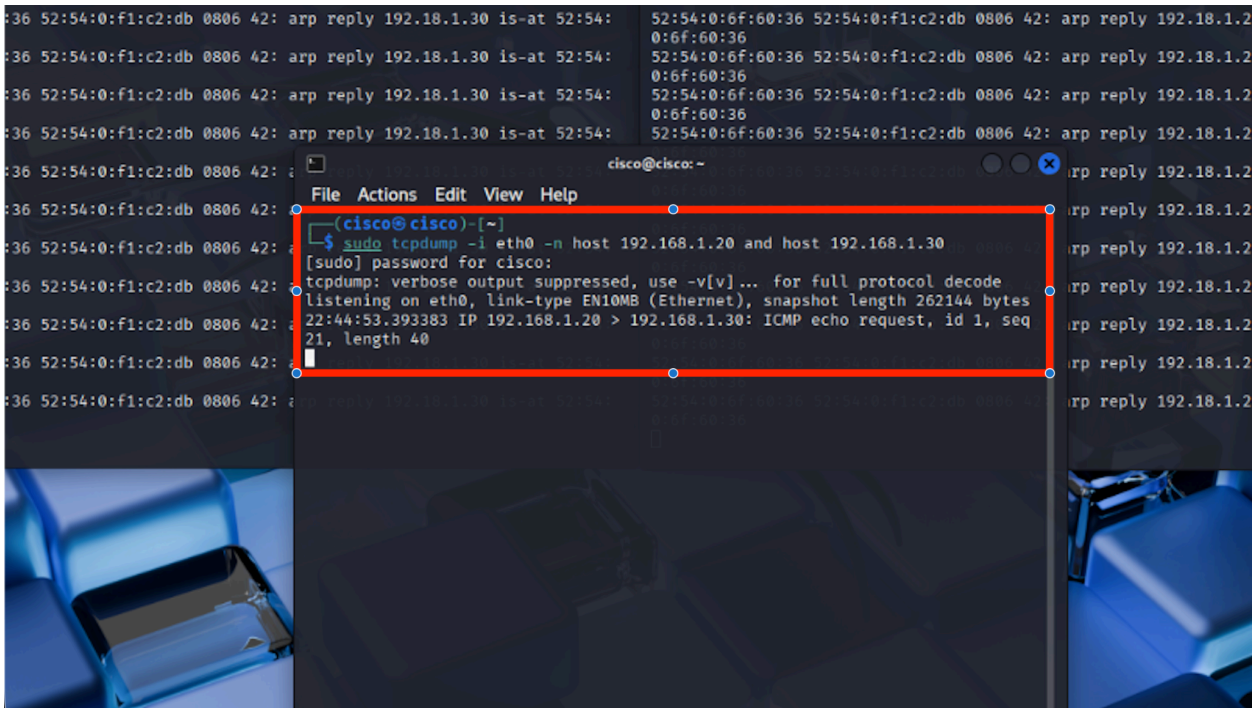
1. Open a third terminal on Kali.
2. Start capturing traffic between Client20 and Client30:
   sudo tcpdump -i eth0 -n host 192.168.1.20 and host 192.168.1.30

3. On Client20, open CMD and ping Client30:
   ping 192.168.1.30

4. Observe the tcpdump output on Kali.

Expected Output:

This proves all packets are flowing through Kali — you're MITM.



Traffic Interception Confirmed

STEP 4: Verify Pre-Attack Connectivity Between Client20 and Client30

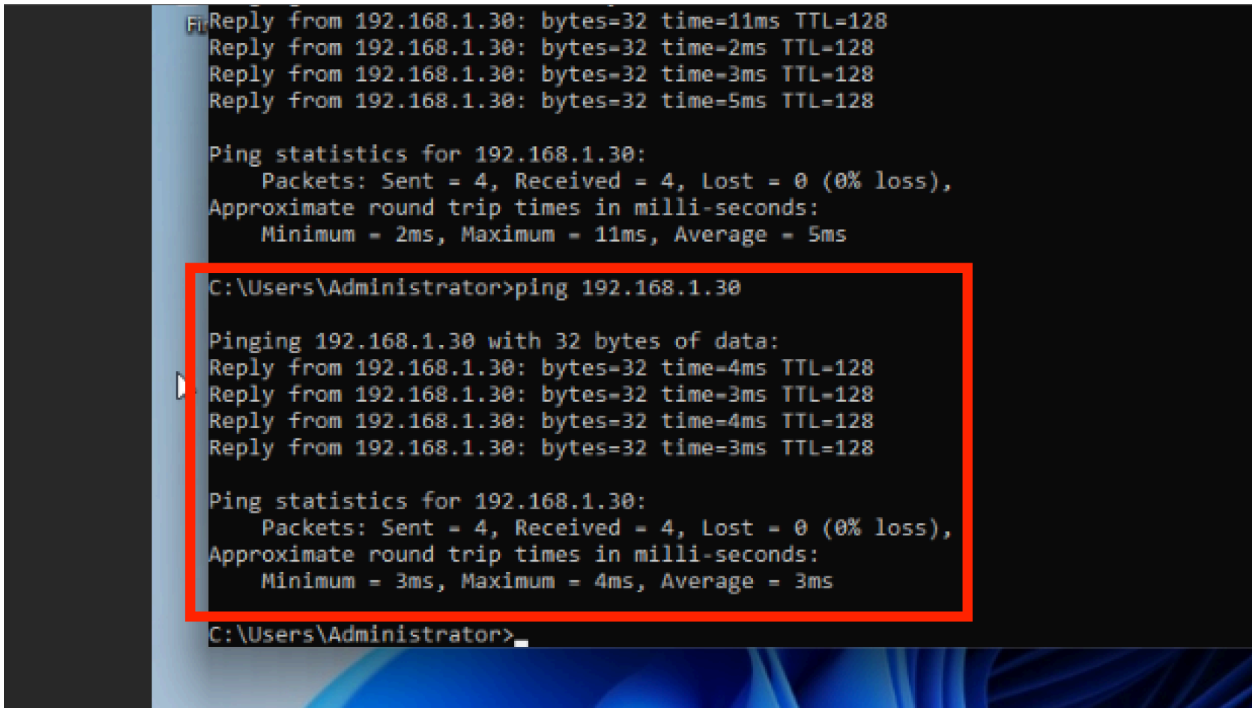
Instructions:

1. On Client20, open Command Prompt (CMD).
2. Run:
   ping 192.168.1.30

3. Ensure you get 4 successful replies (no timeouts).

Expected Output:

This proves the network is functional before any attack.



Pre_Attack- Client20 can reach Client30

STEP 5: Enable Dynamic ARP Inspection (DAI) on SWO1 to Defend Against ARP Spoofing

Instructions:

1. Double-click SWO1 in CML → open the CLI tab.
2. Enter privileged mode:
   enable

3. Enter global configuration mode:
   configure terminal

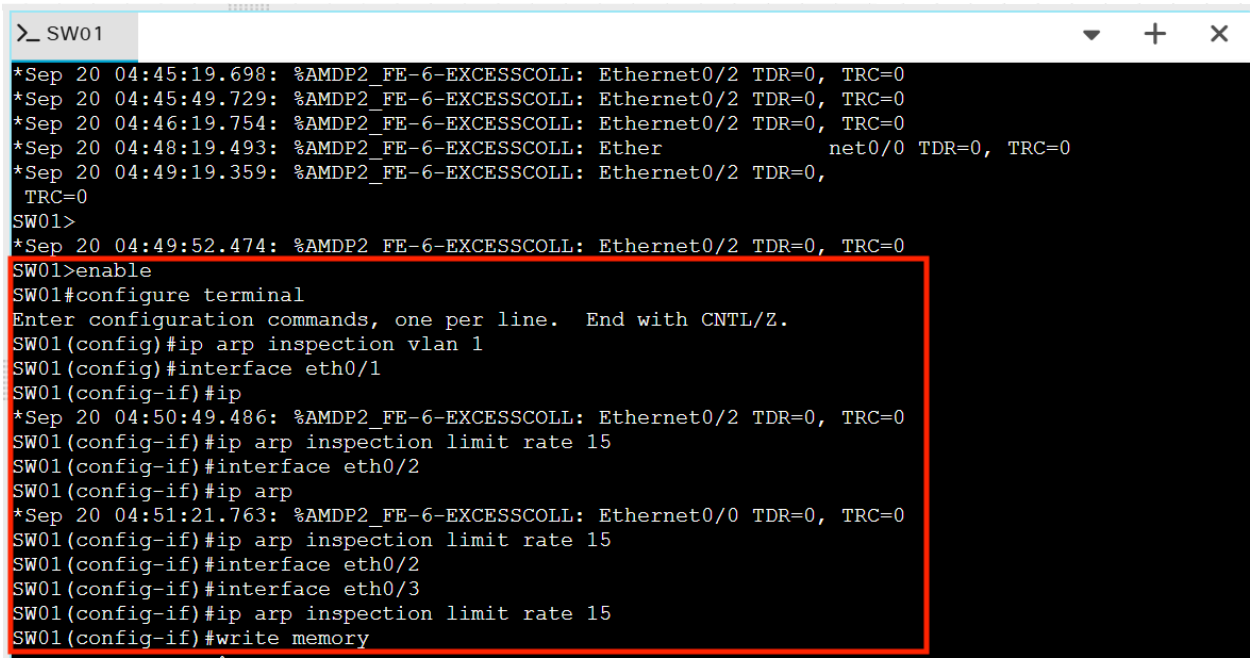
4. Enable Dynamic ARP Inspection:
   ip arp inspection vlan 1

5. Ensure no ports are trusted: (Do command for eth0/1, eth0/2, eth0/3)
   interface eth0/1
   ip arp inspection limit rate 15

6. Save the configuration:
   write memory

Expected Output:

No errors, all commands accepted.
We have no legitimate DHCP server, so do not trust any port. DAI will block all unverified ARP.

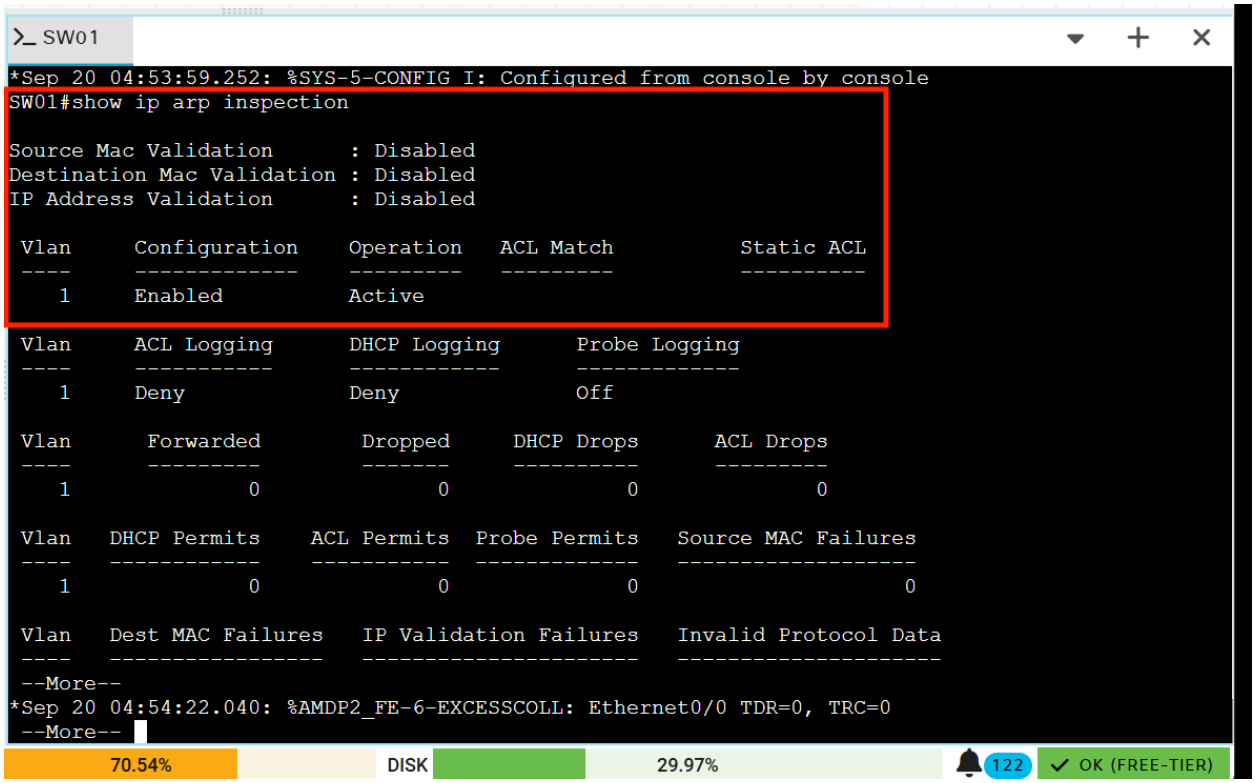


Dynamic ARP Inspection (DAI) Enabled on SW01

STEP 6: Confirm SWO1 Is Active and DAI Is Enabled
Instructions:

1. Still in SWO1 CLI, run:
   show ip arp inspection

Expected Output:
This confirms DAI is active on VLAN 1.



Shows_Active_SW01

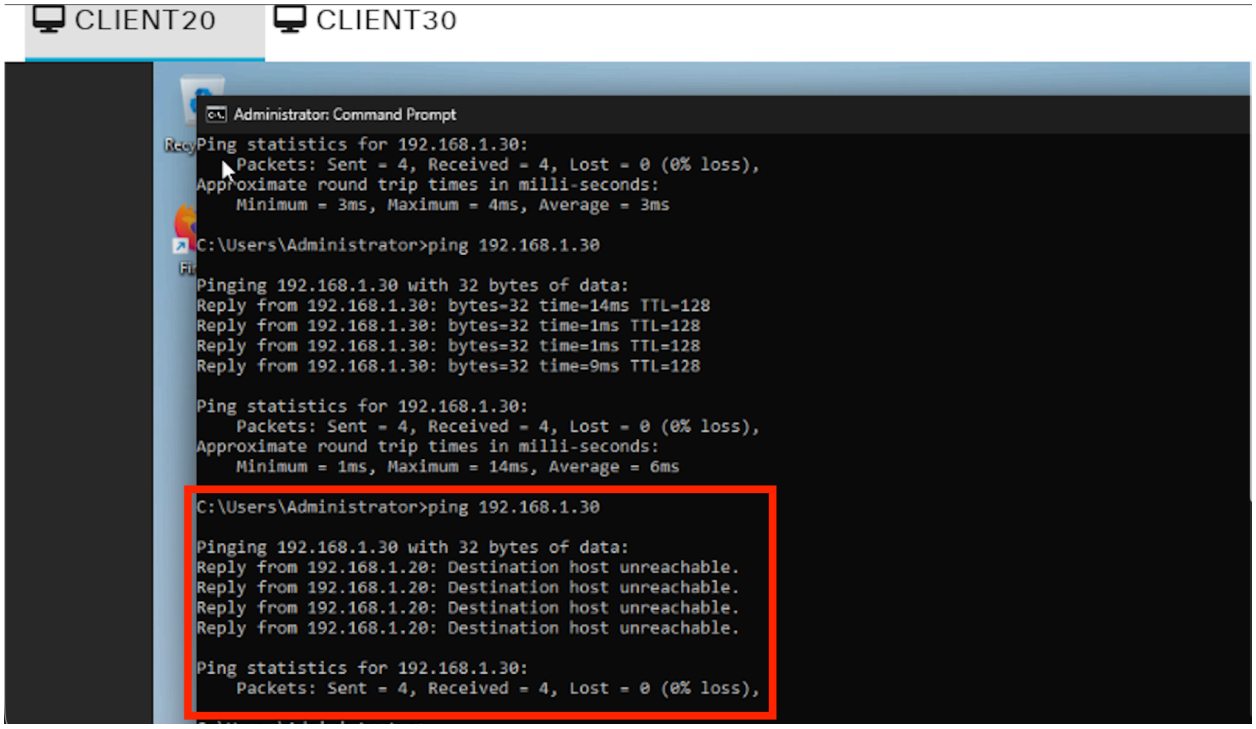
STEP 7: Verify Attack Is Blocked — Client20 Can No Longer Reach Client30

Instructions:

1. On Client20, open CMD.
   Run:
   ping 192.168.1.30

Expected Output:

DAI blocked Kali's fake ARP replies → Client20's ARP table is still poisoned (it doesn't auto-fix), but Kali is no longer forwarding because it's blocked.
So Client20 sends packets to the wrong MAC (Kali's), but Kali no longer replies — ping fails.



Attack Blocked - Client20 cannot reach Client30 after DAI is enabled

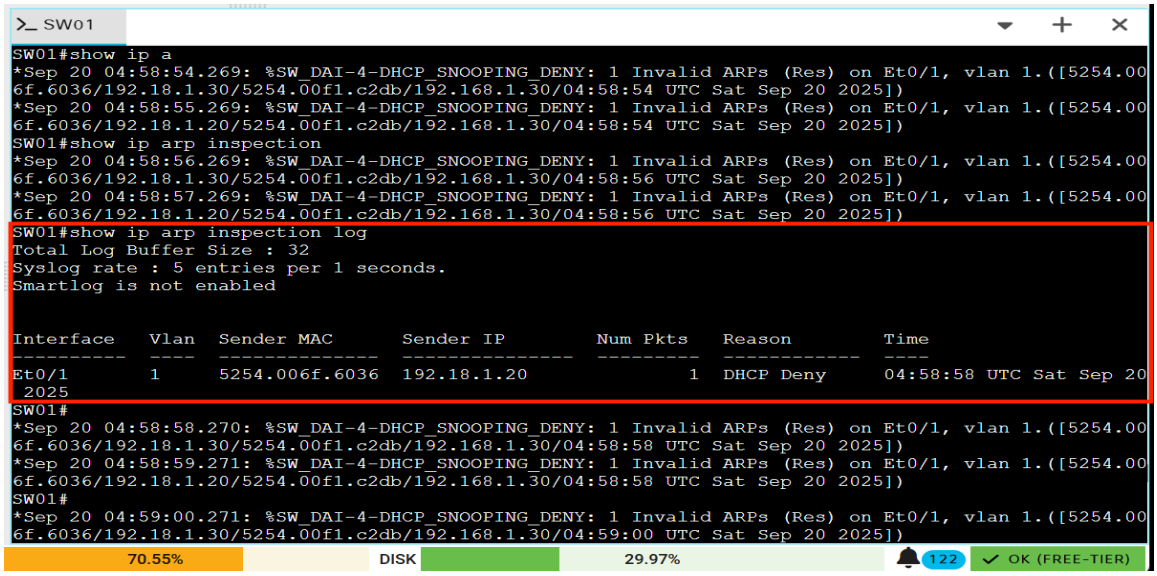
STEP 8: View DAI Log — Rogue ARP Packet from Kali Was Dropped

Instructions:

1. In SWO1 CLI, run:
   show ip arp inspection log

Expected Output:

This confirms SWO1 detected and blocked Kali's fake ARP reply.



DAI Log- Rouge ARP Packet from Kali Dropped - Attack Prevented

STEP 9: Confirm DHCP Snooping Binding Table Is Empty

Instructions:

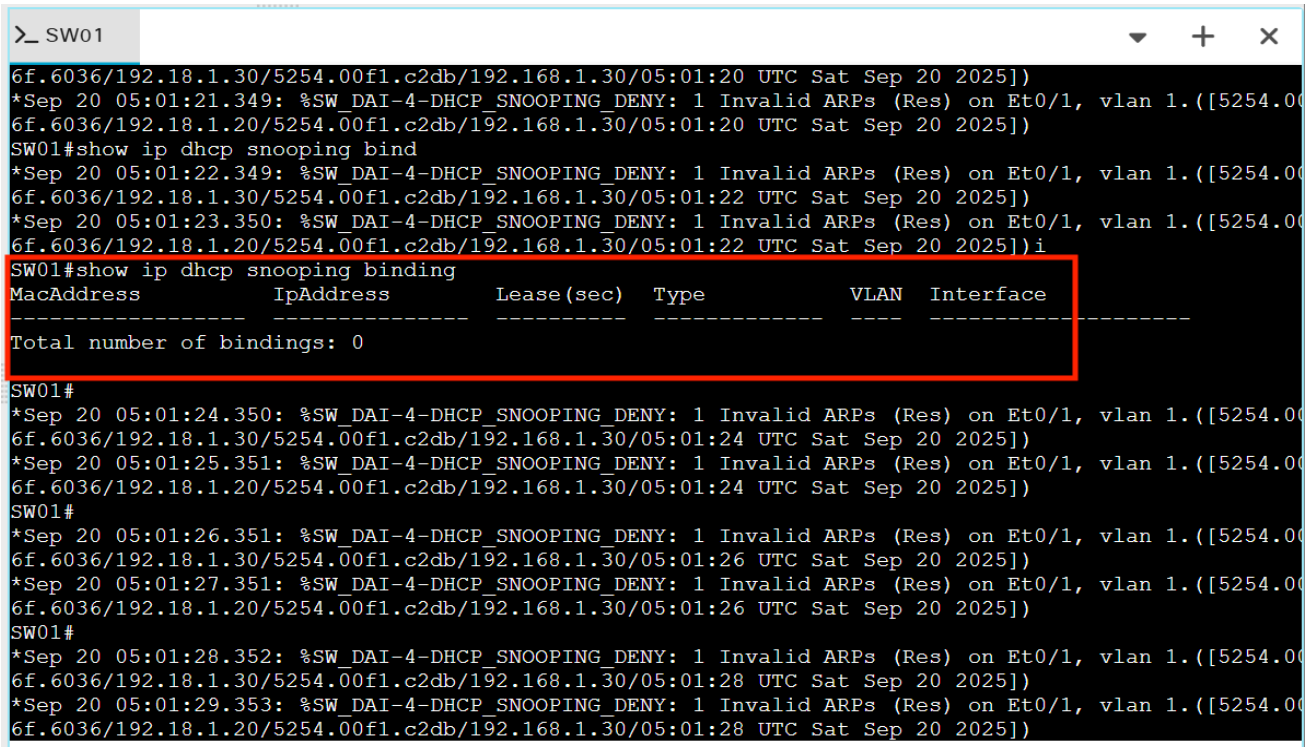
1. In SWO1 CLI, run:
   show ip dhcp snooping binding

Expected Output:

Why empty?

There is no DHCP server in this lab, so no bindings were ever created.

This is normal and expected DAI still works even without bindings.



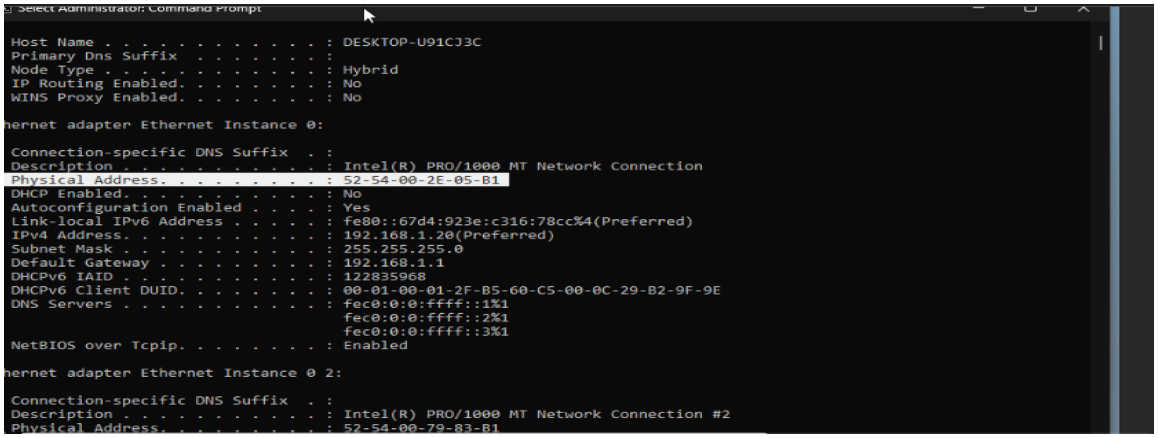
DHCP Snooping Binding table empty

Attack 2 - MAC Spoofing

STEP 1: Find Client20's Real MAC Address

Instructions:

1. On Client20 (Windows): CMD
   Ipconfig /all
2. Copy the Physical Address (example: 52-54-00-2e-05-b1)



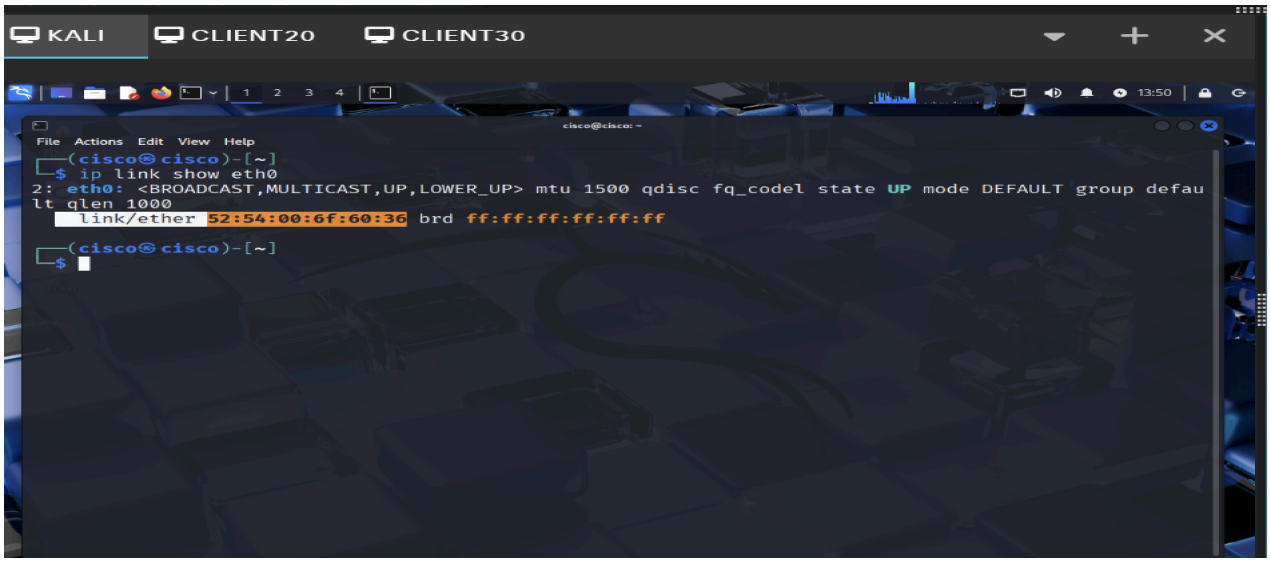
Client20 Real MAC Address

STEP 2: Check Kali's Original MAC

Instructions:

1. On Kali:

   ip link show eth0



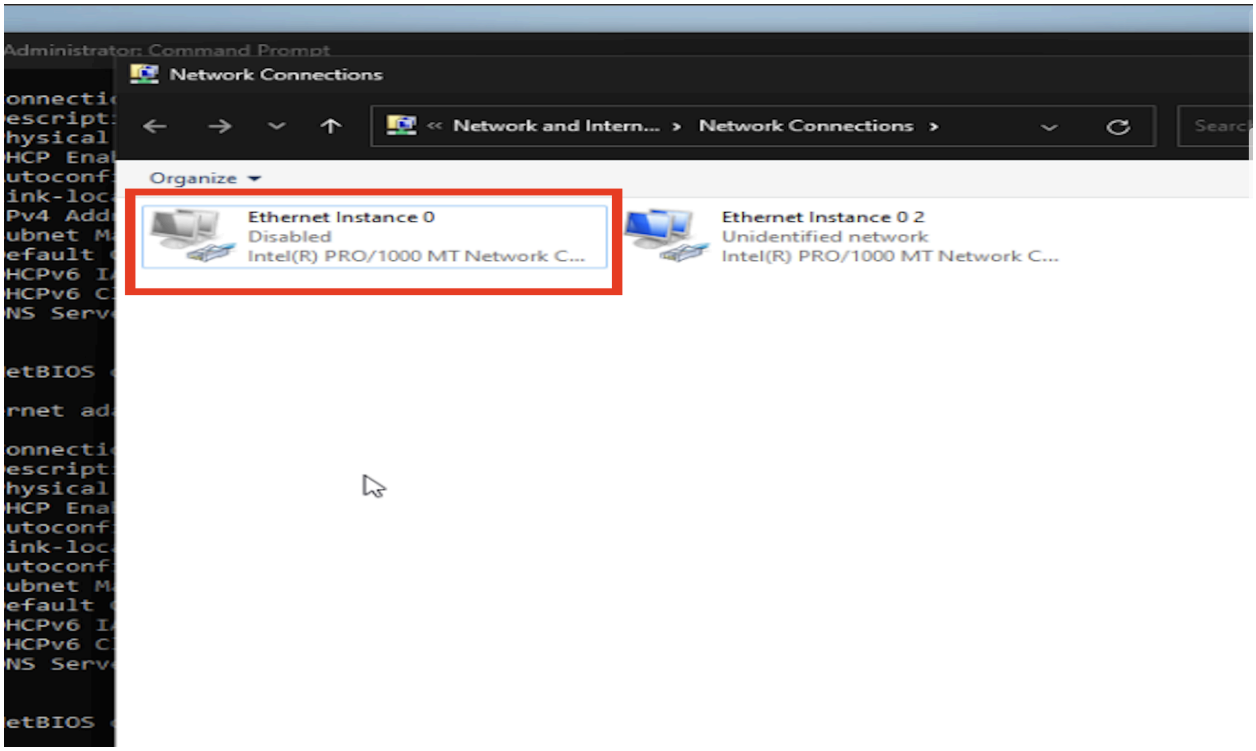
Kali's current MAC Address - Different from Client20

STEP 3: DISABLE CLIENT20'S NETWORK INTERFACE ← CRITICAL STEP

Instructions:

1. On Client20 (Windows):
2. Press Win + R → type ncpa.cpl → Enter
3. Right-click Ethernet → select Disable
4. Wait 5 seconds, the link in CML should turn red (disconnected)

This is the key; now only Kali has that MAC address on the network.



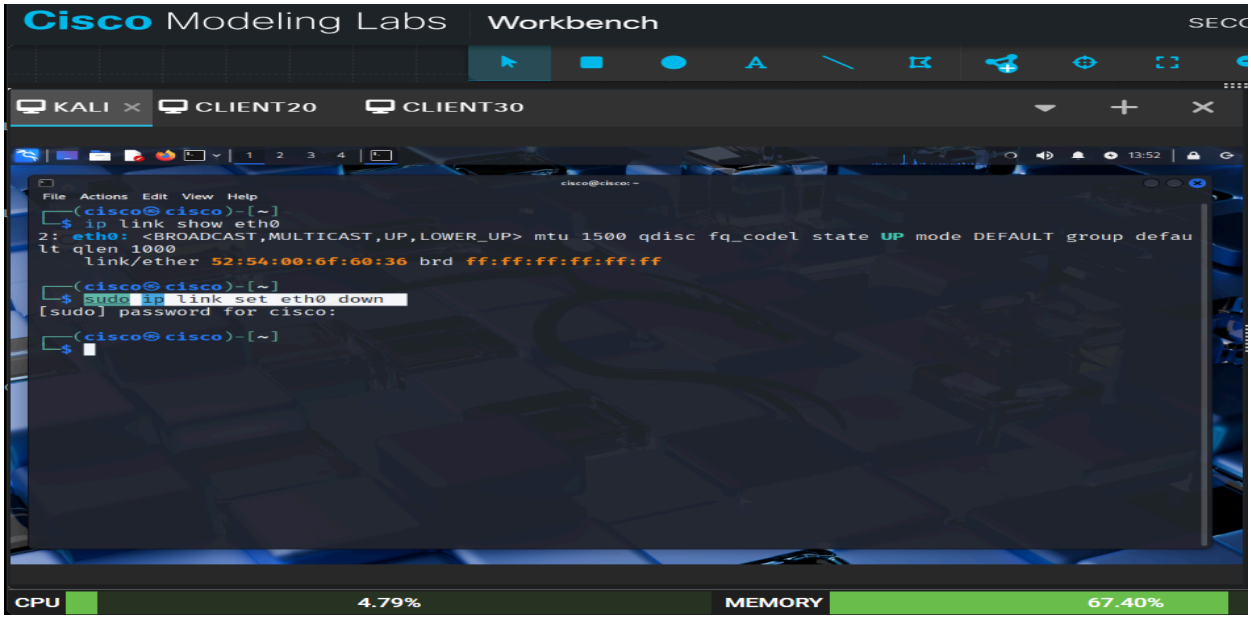
Client20 Interface Disabled - Only Kali now has its MAC

STEP 4: Disable Kali's Interface

Instructions:

1. On Kali:

   sudo ip link set eth0 down



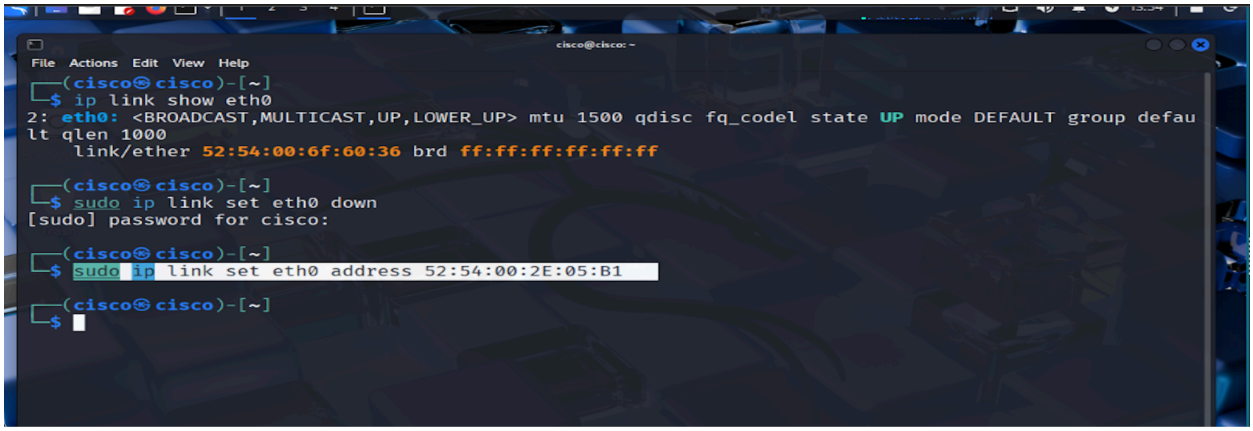
Kali eth0 Disabled - Preparing for MAC Change

STEP 5: Change Kali's MAC to Client20's MAC

Instructions:

1. On Kali:

   sudo ip link set eth0 address 52:54:00:2e:05:b1



Kali MAC Changed to Client20's MAC Address

STEP 6: Re-enable Kali's Interface

Instructions:

1. On Kali:

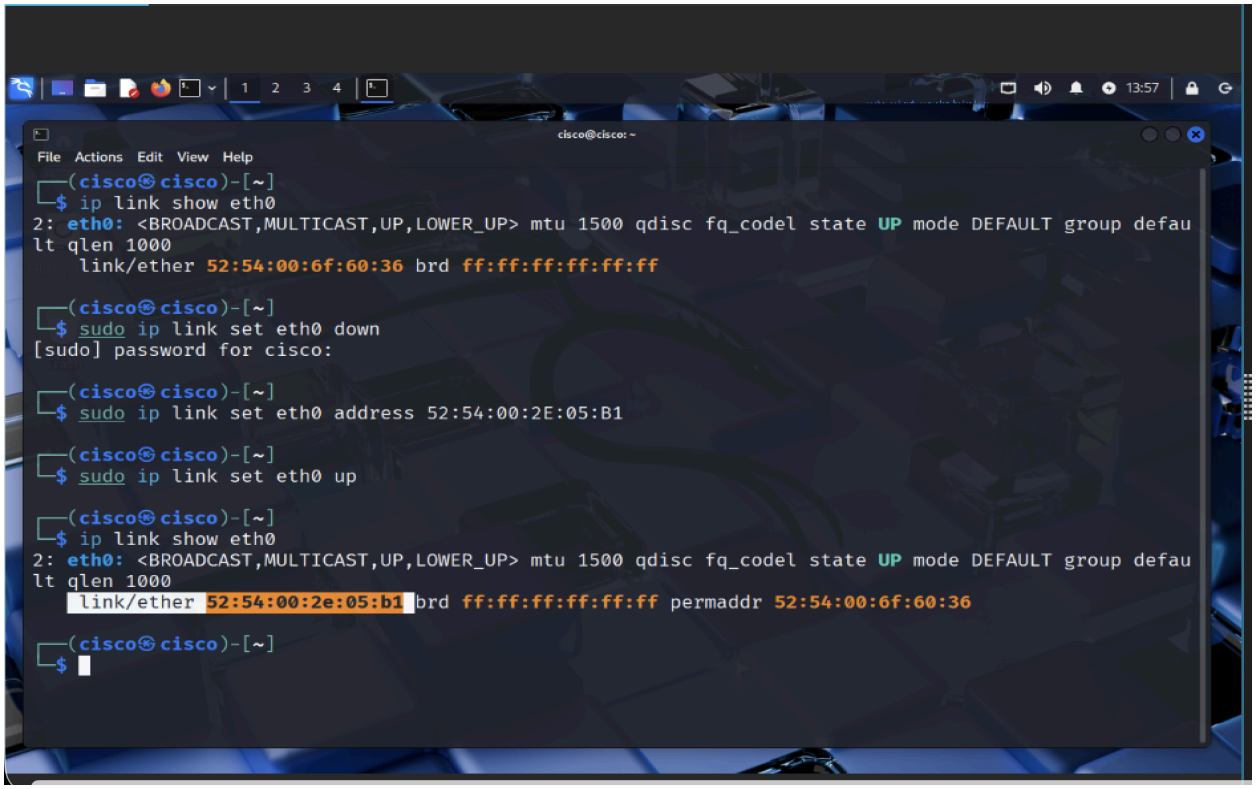
   sudo ip link set eth0 up

Kali eth0 Re-enabled - Now Spoofing Client20's MAC

STEP 7: Verify Kali Has the Correct MAC

Instructions:

1. On Kali:
   ip link show eth0



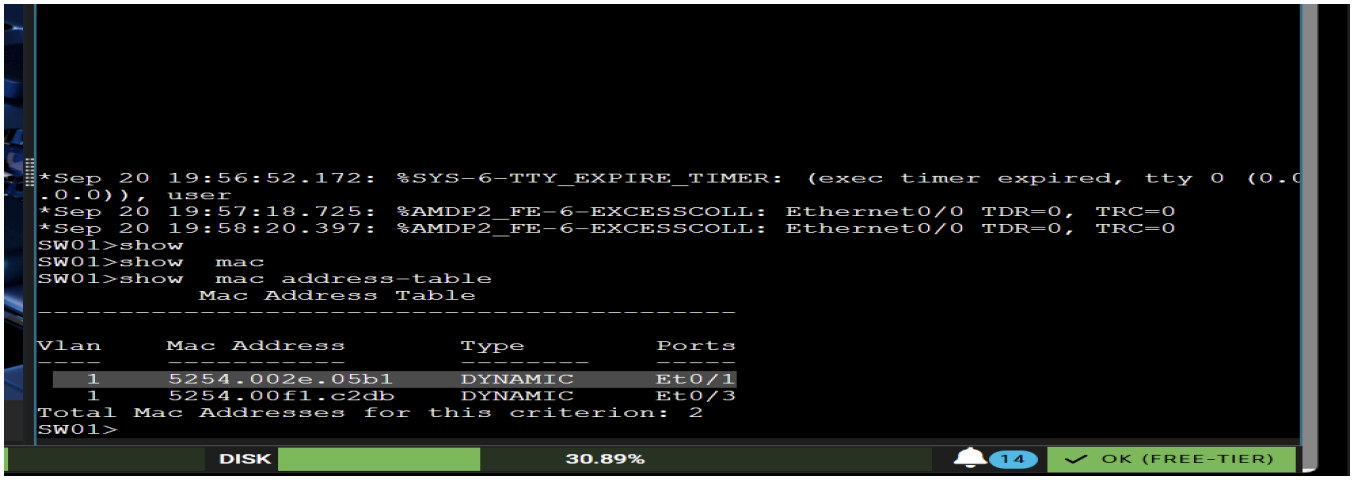
Kali MAC Verified - Now Identical to Client20

STEP 8: Check SWO1 MAC Table — MAC Is Now Only on Kali's Port

Instructions:

1. On SWO1 CLI:
   show mac address-table
   Look for:

Vlan	Mac Address	Type	Ports	
1	52:54:00:2e:05:b1	DYNAMIC	Et0/1	← Kali's port!



SW01 MAC Table - Client20's MAC now on Kali's Port Et0/1

STEP 9: Test Attack Client30 Cannot Ping Client20

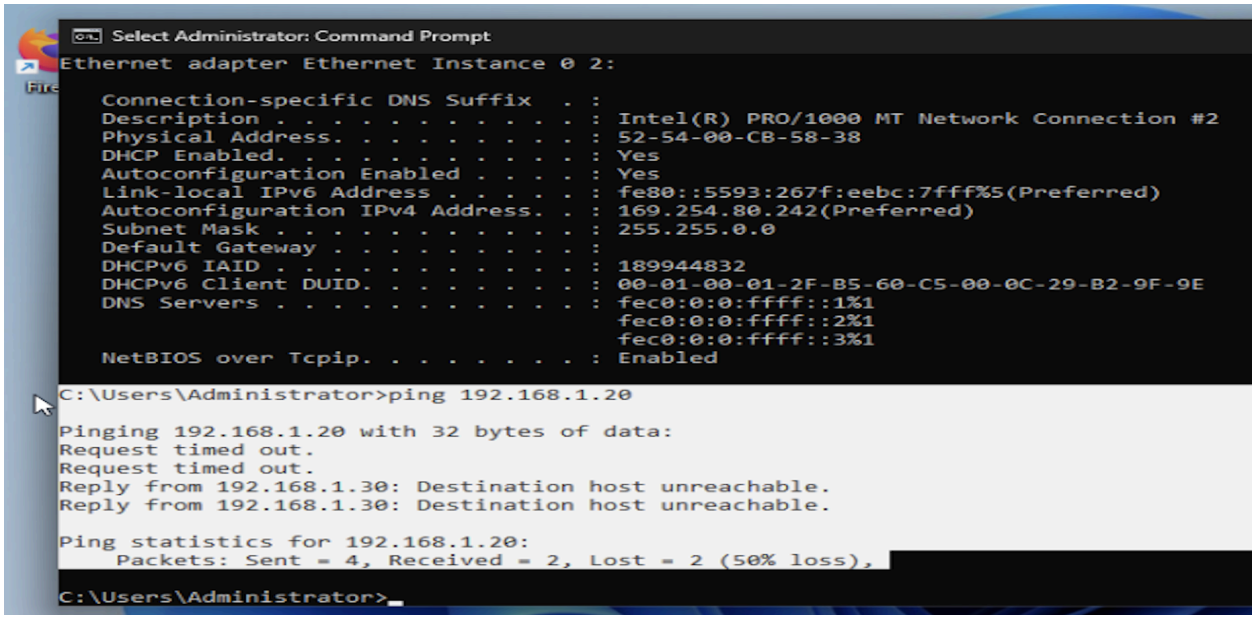
Instructions:

1. On Client30 (Windows):

   ping 192.168.1.20

   You should now see Request timed out.

Because Client20 is offline, and Kali is now the only device with that MAC, but Kali is not forwarding traffic (unless you enable IP forwarding), so no reply is sent back.



Client30 Ping to Client20 - Spoofed MAC Active, Real Device Disabled

STEP 10: DEFENSE (Block Attack)

Instructions:

Port Security on Cisco switches prevents this:

1. On SW01:
   interface et0/2
   switchport port-security
   switchport port-security mac-address sticky
   switchport port-security maximum 1

This locks the port to only one MAC. If Kali tries to use Client20's MAC on et0/1, the port shuts down.

```
interface GigabitEthernet0/1
 switchport mode access
 switchport port-security
 switchport port-security maximum 1
 switchport port-security mac-address sticky
 switchport port-security violation shutdown
```

SW01 command to lock the port to one MAC.

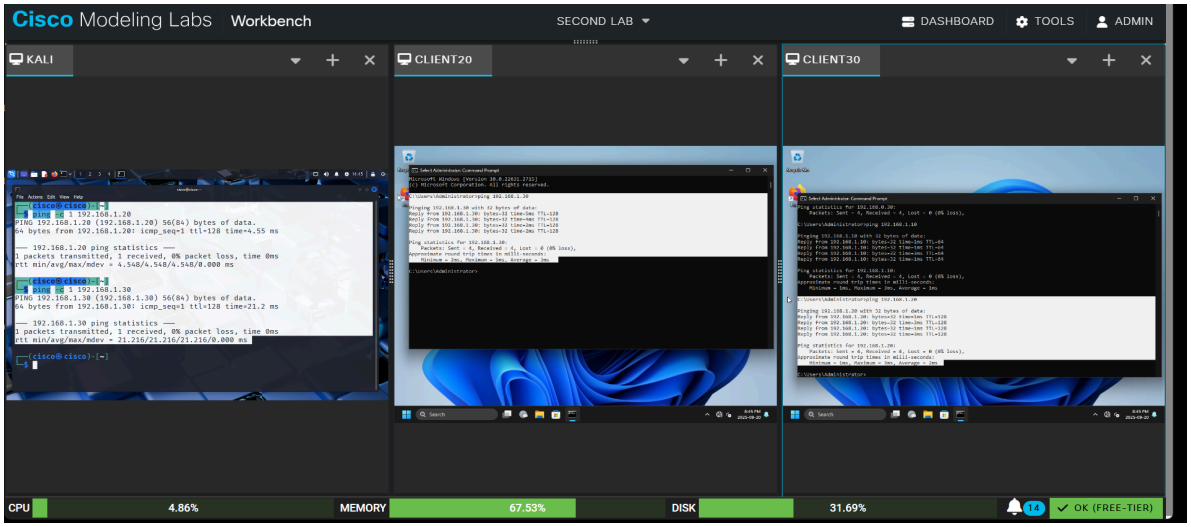
Attack 3 - MAC Flooding

STEP 1: Verify Normal Network Operation (Baseline)

Instructions:

Before attacking, confirm everything works normally.

1. On Client20 (Windows CMD):
   ping 192.168.1.30
   On Client30 (Windows CMD):
   ping 192.168.1.20
   On Kali:
   ping -c 4 192.168.1.20
   ping -c 4 192.168.1.30
   All pings should succeed.



Pre_Attack- Client20 and Client30 can ping eachother and Kali can ping Client20 and 30

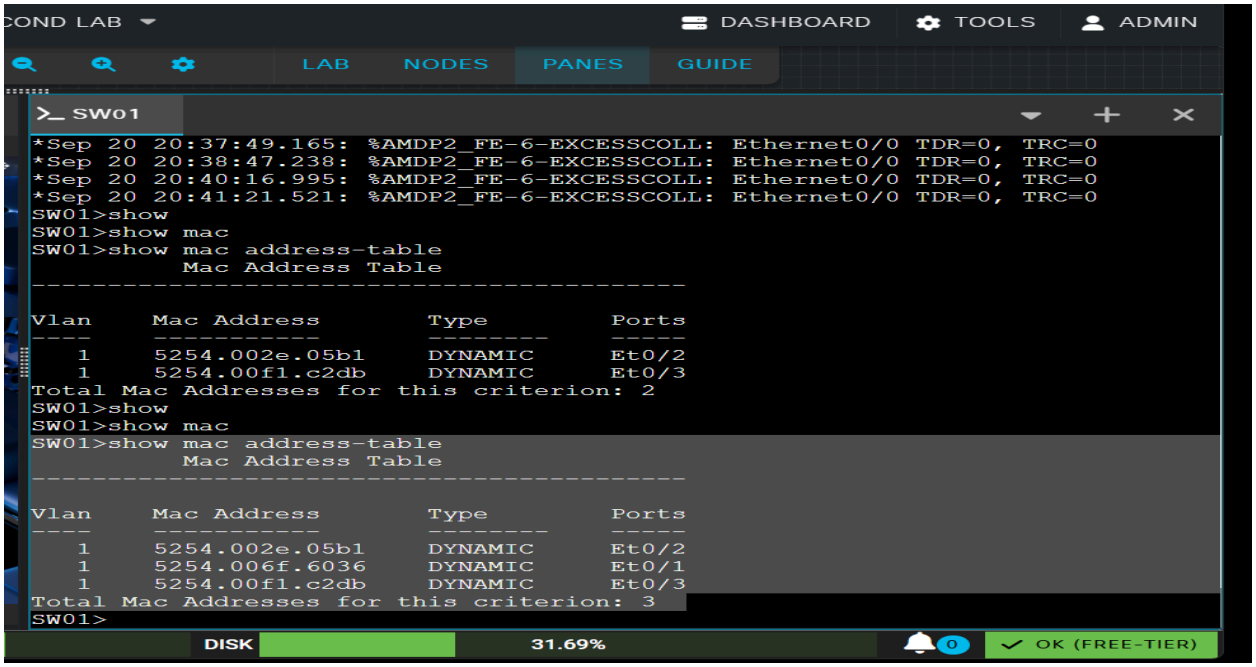
STEP 2: Check SWO1's MAC Table (Baseline)

Instructions:

1. On SWO1 CLI:
2. Double-click SWO1 → open CLI
3. Run:

   show mac address-table

   You should see 3 entries:



SW01 MAC Table - Normal (3 Entries Only)

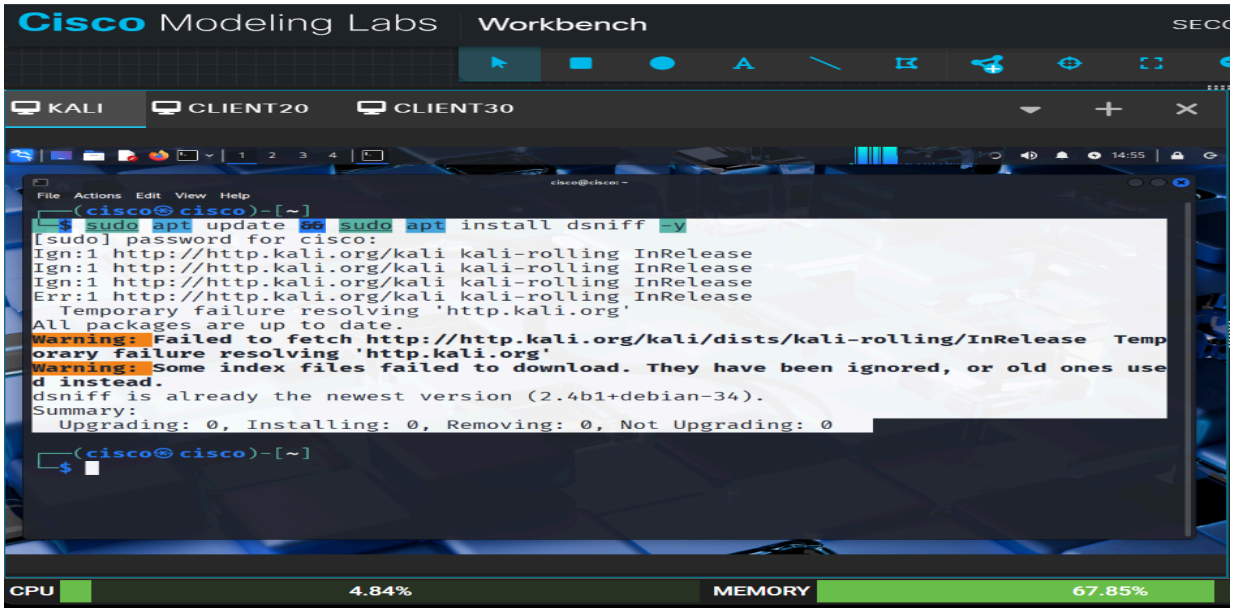
STEP 3: Install MAC Flooding Tool on Kali

Instructions:

1. On Kali:

   sudo apt update && sudo apt install dsniff -y

This installs **macof** a tool that sends hundreds of random Ethernet frames with random MAC addresses.



dsniff Installed - macof Ready for Use (Only shows failed as it was previously installed.)

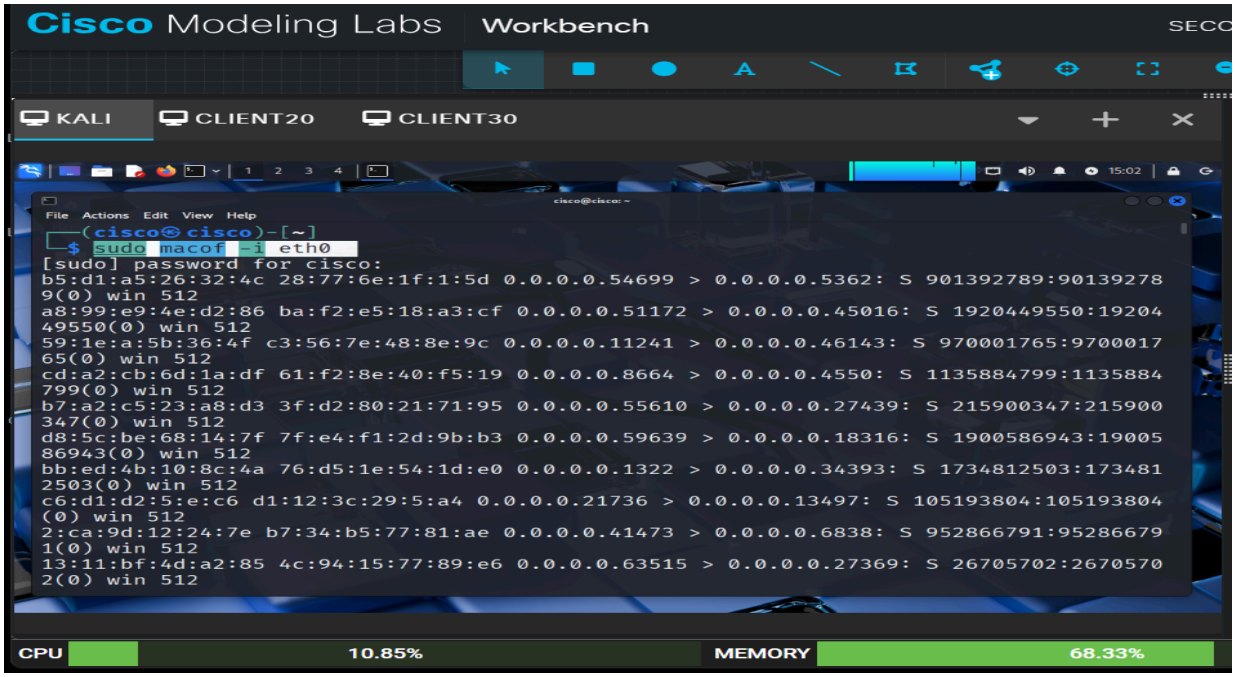
STEP 4: Launch MAC Flooding Attack

Instructions:

1. On Kali:

   sudo macof -i eth0

This sends 10–20 fake MAC addresses per second, enough to flood a small switch table.
Each line = one fake Ethernet frame with a new random MAC.
Let it run for 15–20 seconds, long enough to overflow the CAM table.



MAC Flooding in Progress - Kali Sending Hundreds of Random MACs

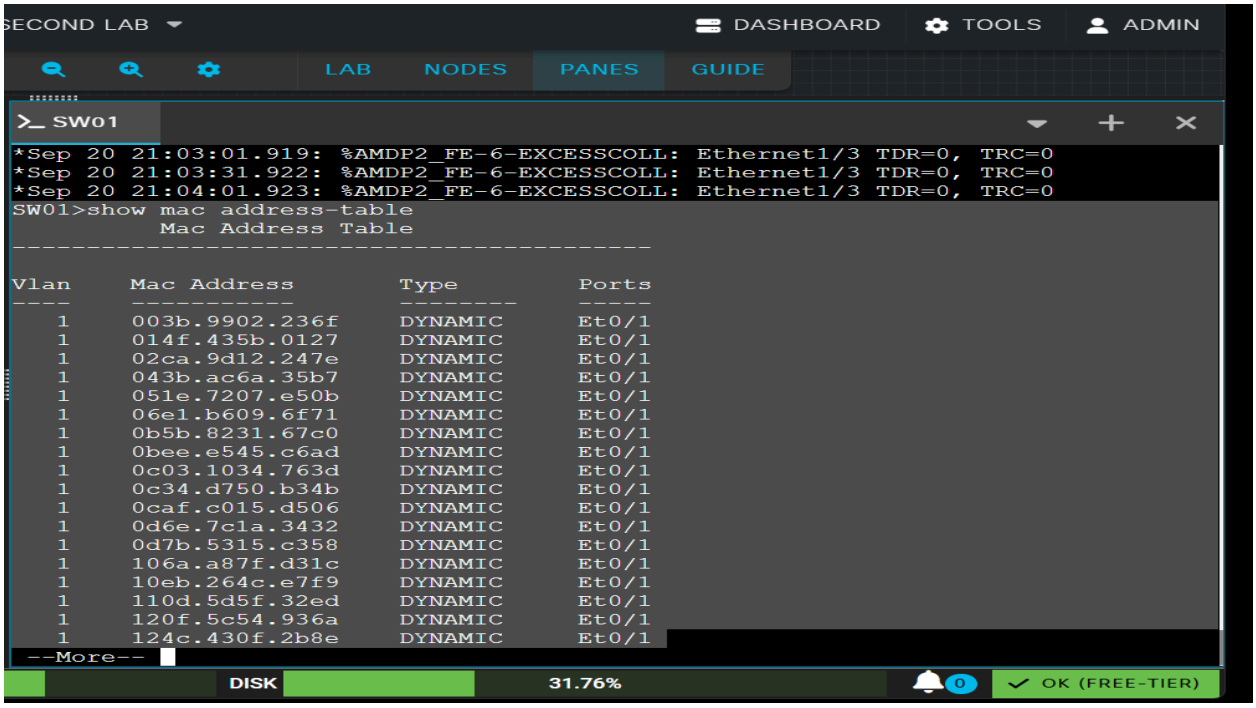
STEP 5: Watch SWO1's MAC Table Fill Up

Instructions:

1. On SWO1 CLI, while macof is running:

   show mac address-table

After 10–15 seconds, you'll see dozens or hundreds of entries:



SW01 MAC Table Overflowed - 50+ Fake MACs Learned

STEP 6: ENABLE PROMISCUOUS MODE ON

Kali's eth0 ← CRITICAL

By default, network interfaces only accept packets addressed to their own MAC.

After MAC flooding, SWO1 broadcasts all traffic to all ports, including Kali.

But Kali's interface ignores those packets, unless you put it in promiscuous mode.

1. On Kali, run:

   sudo ip link set eth0 promisc on

   No output, this is normal.

2. Verify it's enabled:

   ip link show eth0

   Look for

   2: eth0: <BROADCAST, MULTICAST, PROMISC, UP, LOWER_UP> …

   PROMISC must appear in the flags!

Kali eth0 Promiscuous Mode Enabled - Now Accepts All Traffic

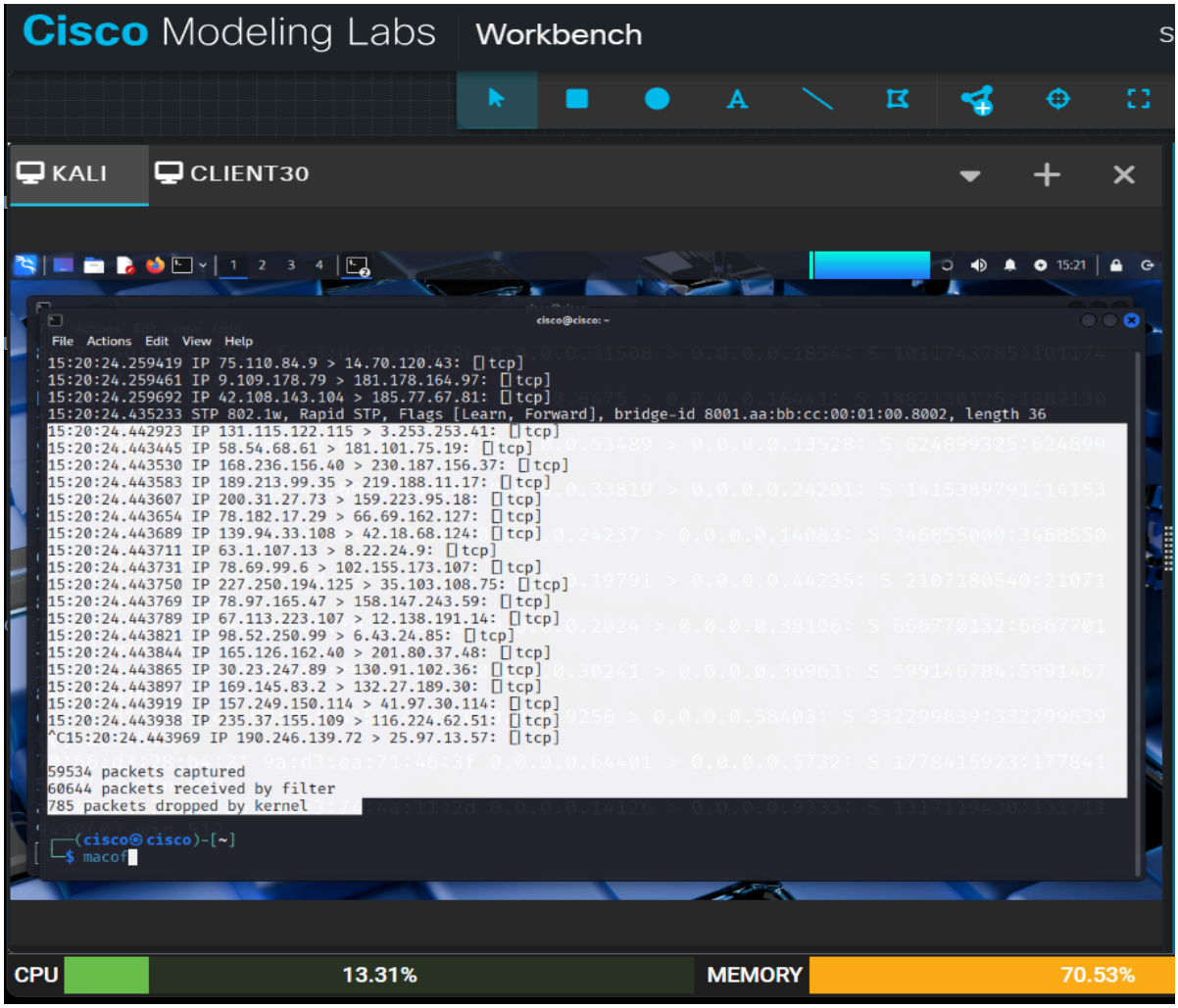
STEP 7: Observe Switch Behavior — Fail-Open Mode Activated

Instructions:

1. On Kali, start packet capture:

   sudo tcpdump -i eth0 -n host 192.168.1.20 and host 192.168.1.30

   Even if you don't ping anyone, you'll still see Client20 ↔ Client30 traffic on Kali, because the switch is broadcasting everything.



Kali's Sniffing All Network Traffic - MAC Flooding Complete

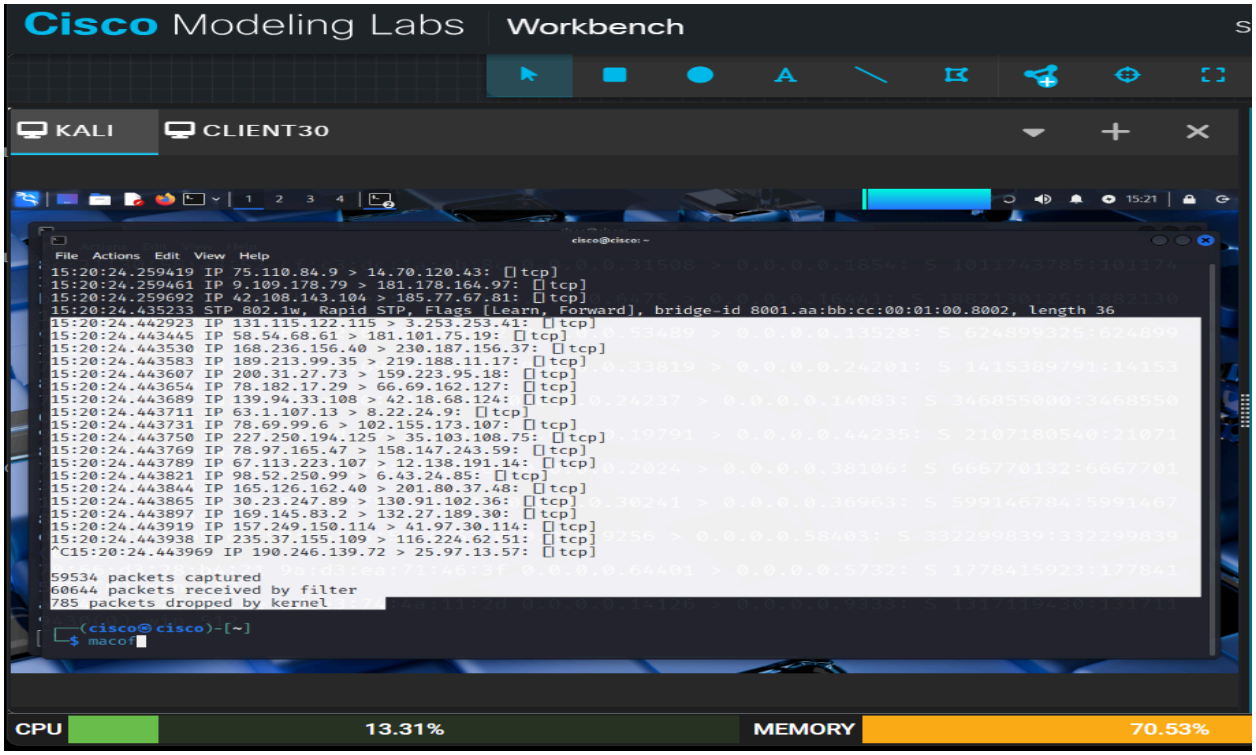
STEP 8: Stop Attack and Verify Recovery

Instructions:

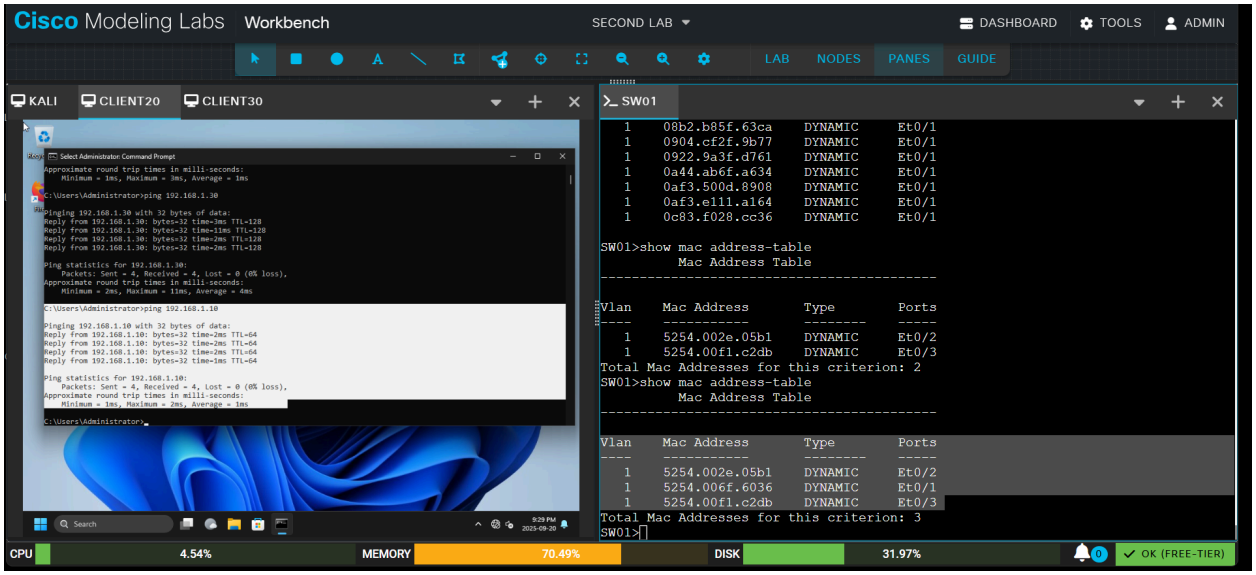
1. On Kali, press **ctrl + c** to stop macof. (Image 1)
2. Wait 10–15 seconds, and the switch will age out the fake MACs.
3. On SWO1 CLI, run again:

   show mac address-table

You should now see only the original 3 MACs: (Image 2)



ctrl + c = Stop macof



MAC Table recovered (All 3 MACs)

STEP 9: Enable Port Security on Each Access Port (Block Attack)

Port Security lets you:

- Limit how many MAC addresses a port can learn
- Automatically shut down the port if someone tries to flood it
- Only allow pre-approved MACs

Instructions:

1. On SWO1 CLI:

   enable

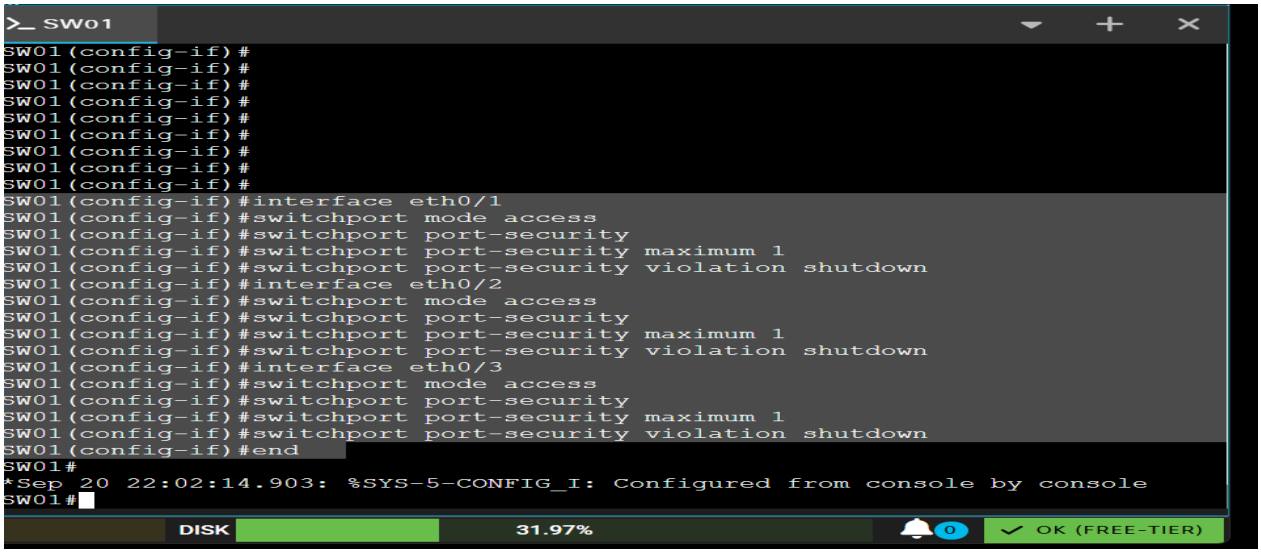
   configure terminal

2. Apply Port Security to each access port (eth0/1, eth0/2, eth0/3):

   interface eth0/1 (Change the port # each time)
   switchport mode access
   switchport port-security
   switchport port-security maximum 1
   switchport port-security violation shutdown
   end

Explanation of each command:

- switchport mode access → Ensures port is not a trunk (only one device)
- switchport port-security → Enables port security
- maximum 1 → Only one MAC address allowed per port
- violation shutdown → If a second MAC appears → port shuts down automatically



Port Security Enabled on eth01, eth02, et03 - maximum 1 MAC per Port

STEP 10: (Optional) Save MAC Addresses as "Sticky" (Recommended)

Sticky MAC means:

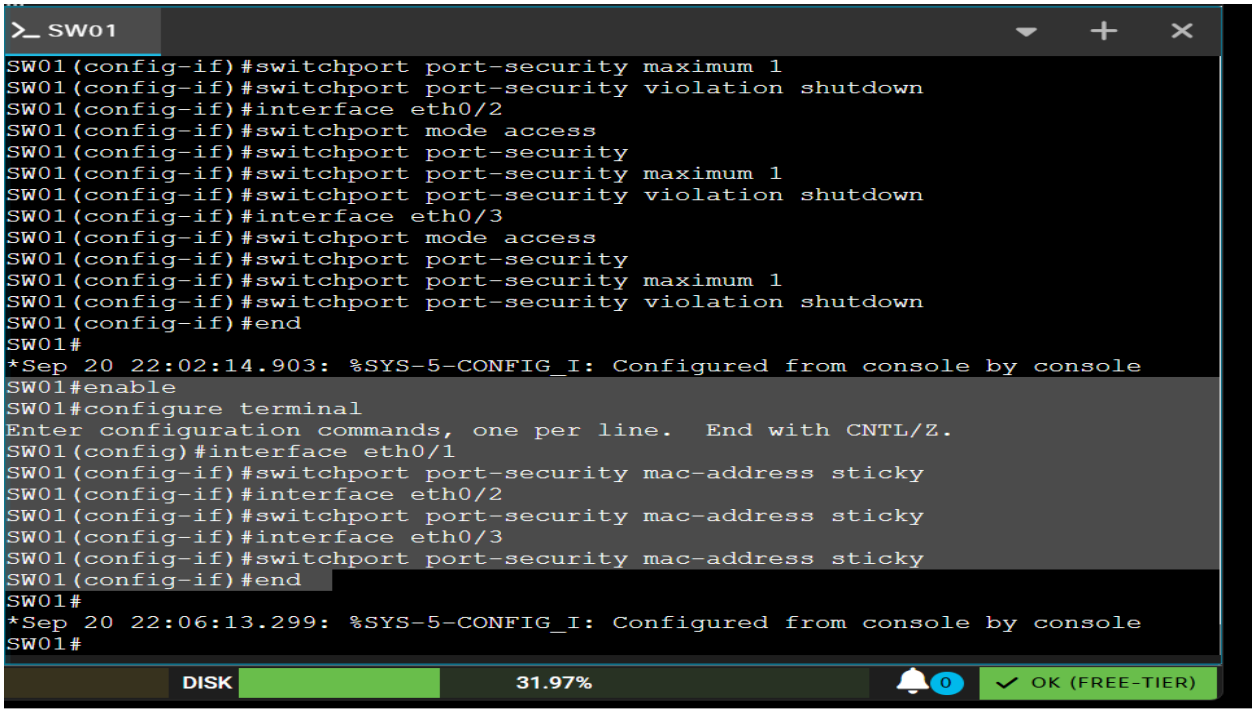
The switch automatically learns the first MAC on the port and locks it, even after a reboot.

Instructions:

1. On each interface, add this line:

   switchport port-security mac-address sticky

Do this for **all 3 ports**: eth0/1, eth0/2, eth0/3



Sticky MAC Enabled on eth01,eth02,eth03

Why Sticky?

You don't have to manually type MAC addresses. The switch learns them from real traffic and locks them perfectly for labs.

STEP 11: Verify Port Security Is Active

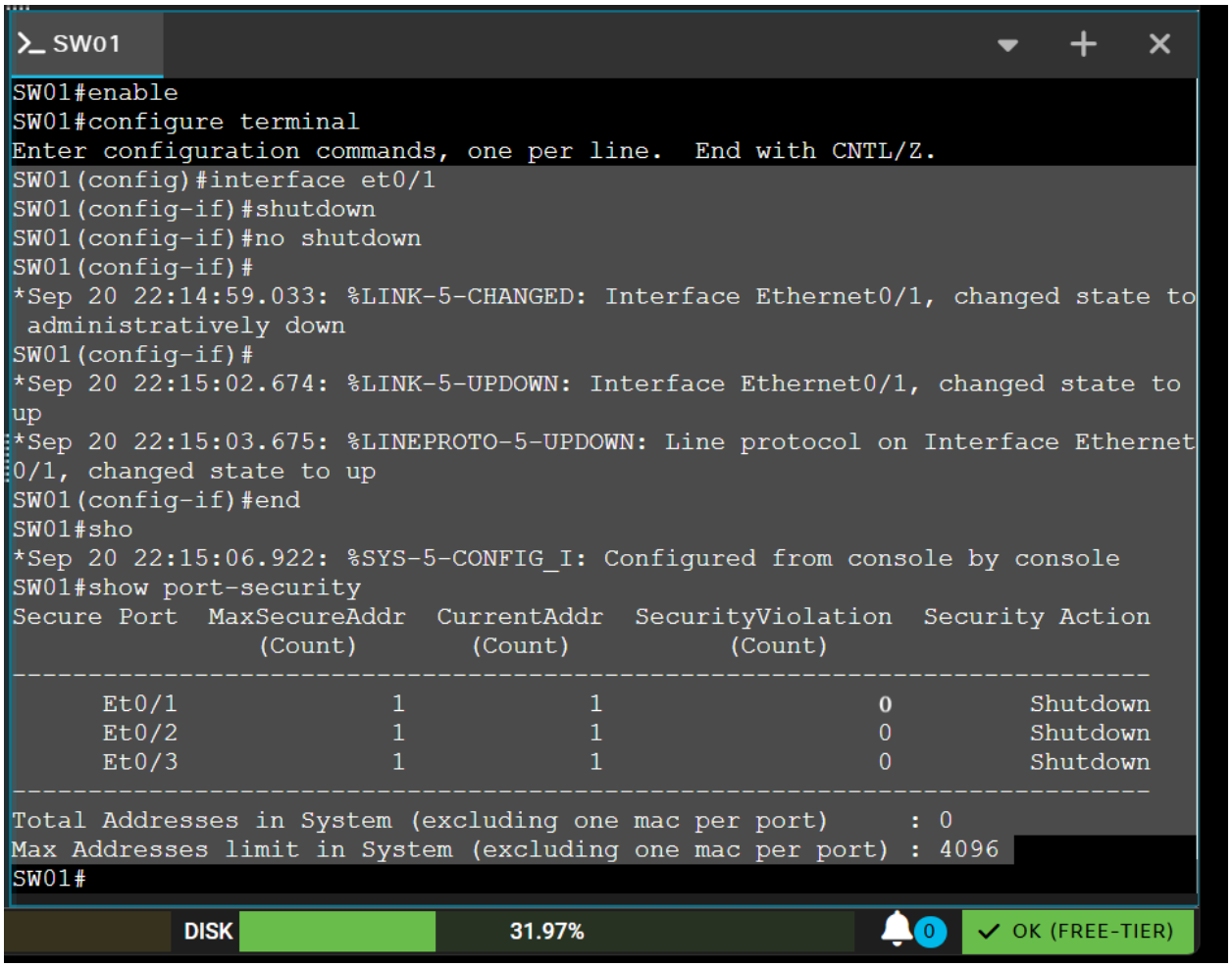
Instructions:

1. On SWO1 CLI:

   show port-security

You should see:

Secure Port	MaxSecureAddr	CurrentAddr	SecurityViolation	SecurityAction
Et0/1	1	1	0	Shutdown
Et0/2	1	1	0	Shutdown
Et0/3	1	1	0	Shutdown



Show port-security

STEP 12: Test the Prevention — Try MAC Flooding Again

Instructions:

1. On Kali, run:

   sudo macof -i eth0

   Let it run for 10 seconds.

   Observe what happens on SWO1:

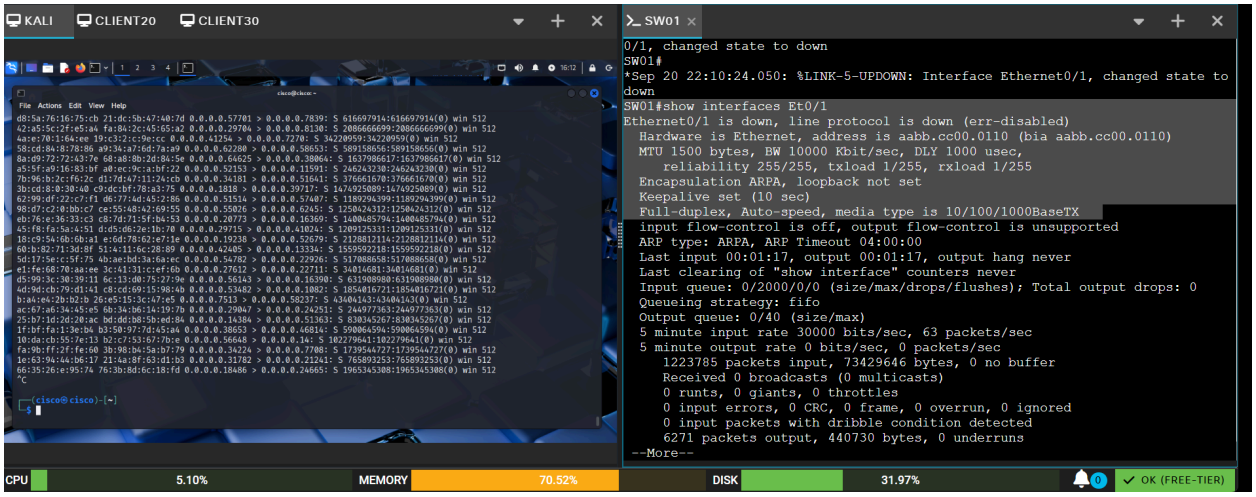
2. On SWO1 CLI, run:

   show interfaces et0/1

   You'll see:

   et0/1 err-disabled 1 a-full a-100 10/100BaseTX

   Status = err-disabled → Port is automatically shut down because Kali tried to send again.



MAC Flooding Attempt Blocked - Port et01 Err-Disabled by Port-Security

STEP 13: Restore the Port (After Attack Attempt)

The port is now disabled — but you can fix it.

Instructions:

1. On SWO1 CLI:

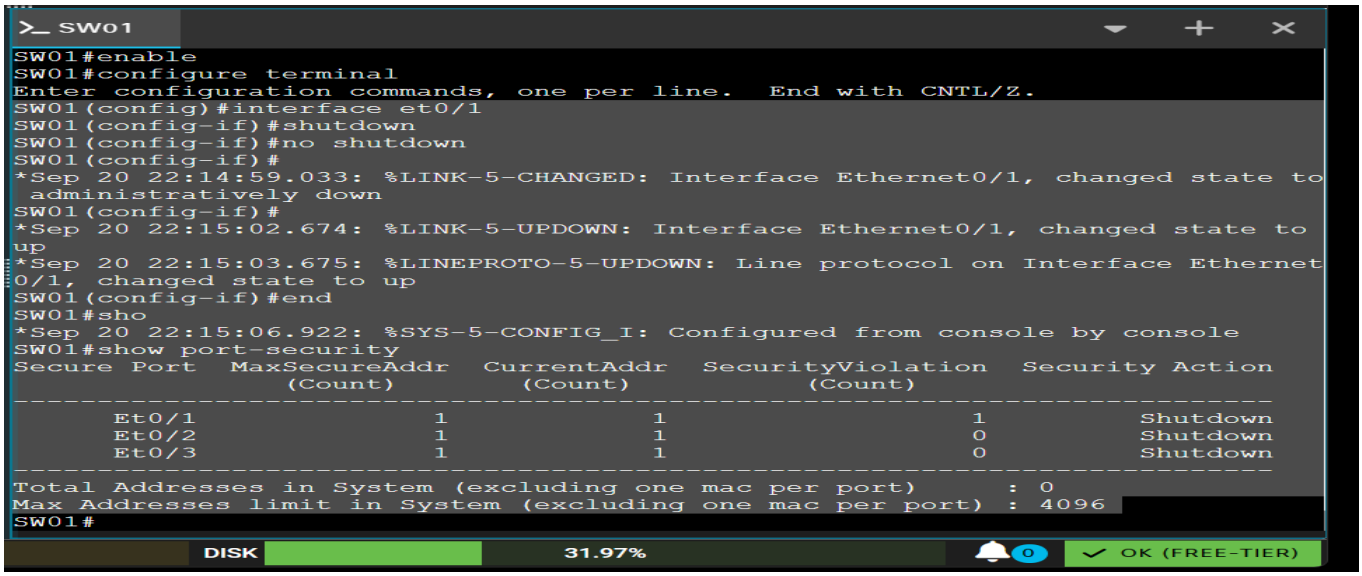
   configure terminal
   interface et0/1
   shutdown
   no shutdown
   end

This re-enables the port, and **re-learns the legitimate MAC** (Kali's).

2. Now run:

   show port-security

You'll see CurrentAddr = 1 again, and SecurityViolation = 1 (to show it was triggered).



Port et0/1 Restored After Violation, Security Triggered, and Recovered

STEP 14: Confirm Network Is Still Working

Instructions:

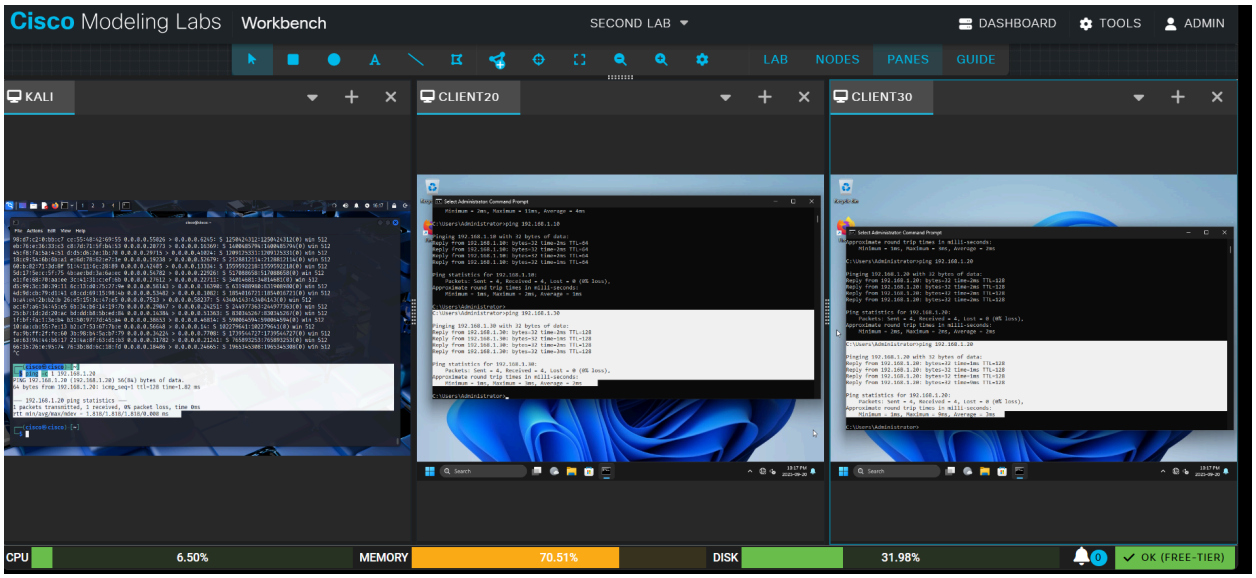
1. On Client20:

   ping 192.168.1.30
2. On Client30:

   ping 192.168.1.20
3. On Kali:

   ping -c 4 192.168.1.20

All pings work, but if you try to run macof again, the port shuts down again.



Normal Network Operation Restored - Port Security Preventing Further Attacks

Questions

Q: What's VLAN Hopping, and how do you stop it?

VLAN hopping? Yeah, it's basically when someone tries to sneak into a VLAN they're not supposed to be in, like picking a lock on a door that was supposed to be locked. There are two classic tricks attackers use.

One's called *switch spoofing*: they pretend their laptop is a switch trunk port by spamming DTP messages. If the port is set to "auto-negotiate," it might just let them in. The other is *double tagging*, where they slap two VLAN tags on a packet. The first tag matches the native VLAN, so the switch strips it off… and suddenly the second tag, meant for a different VLAN, is forwarded right into it. Boom. They're in.

Here's how I shut that down before it even starts:

- I never let access ports negotiate trunks. Ever. I slap switchport nonegotiate on every single one.
- I explicitly set every access port to switchport mode access. No ambiguity.
- I changed the native VLAN on all trunk ports to something useless—like VLAN 999. Something no one uses, no one cares about.
- And I never use VLAN 1 for anything real. Seriously. It's the default for a reason—it's the first thing attackers target.

It's not fancy. It's not sexy. But it stops 90% of VLAN hopping attempts cold. And honestly? If someone's still getting in after this, they're probably bringing a bazooka to a knife fight.

Q: What's STP's job—and how do you stop people from breaking it?

Spanning Tree Protocol (STP) is the traffic cop of Layer 2. It keeps redundant paths from turning your network into a broadcast storm nightmare. It picks one switch as the "root," blocks the rest of the loops, and lets traffic flow cleanly.

But here's the kicker: STP trusts *everything*. If someone plugs in a rogue device and sends a BPDU that says, "I'm the root now!" guess what? The whole network listens. Suddenly, all traffic flows through their laptop. That's a full-on man-in-the-middle. Or worse they flood the network with fake TCNs (topology change notifications), forcing switches to constantly recalculate. Result? Network chaos. Slowdowns. Outages. Denial of service.

So how do I defend it *without* killing redundancy?

- BPDU Guard: Enabled on every access port. If a BPDU shows up there? Port shuts down. Instantly. No questions asked. (Because your user's laptop shouldn't be sending BPDUs.)

- Root Guard: On all uplinks. If another switch tries to become root? Nope. Port gets put into a “root-inconsistent” state. Stays blocked. The real root stays in charge.
- Loop Guard: On designated ports. Catches those sneaky one-way link failures that can trick STP into thinking a link is fine when it’s not.

I’m not removing redundancy, I'm just making sure *only the right switches* get to run the show.

Q: What’s Dynamic ARP Inspection (DAI), and why should you care?

DAI is like the bouncer at a club, except instead of checking IDs, it checks ARP packets. ARP spoofing? That’s when an attacker says, “Hey, the gateway’s MAC address is *mine*.” Suddenly, every device on the subnet sends its traffic to the attacker instead of the router. They can steal passwords, hijack sessions, and sniff everything.

DAI stops that. But here’s the catch: it doesn’t work alone. It needs a list of *trusted* IP-to-MAC bindings. That’s where DHCP Snooping comes in—it builds that list by watching what legitimate DHCP servers hand out.

So I always turn them on together. DAI + DHCP Snooping = a dynamic, real-time ARP firewall. No more “my laptop just became the gateway” nonsense.

It’s not magic. But it makes the attacker’s job way harder, and most of them just walk away when they see it’s enabled.

Q: How do DHCP Snooping, Port Security, and Endpoint Posture Assessment work together to actually secure Layer 2?

Here’s the truth: no single feature makes your network secure. You need layers. Like an onion. Or a castle with moats, walls, and drawbridges.

Let me break down how I piece them together step by step, like a security workflow:

1. Port Security: First thing when a device plugs in, “Who are you?” If the MAC isn’t on the approved list? Port shuts down. No second chances. Stops rogue devices, USB dongles, or someone plugging in a neighbor’s laptop.
2. DHCP Snooping: If the device gets past Port Security, it asks for an IP. DHCP Snooping makes sure it came from a *real* DHCP server, not some rogue one someone brought in. It also builds a living table of who got which IP and where. That’s the foundation for everything else.
3. Endpoint Posture Assessment: Now we ask: “Are you *safe*?” Is your OS patched? Is antivirus running? Firewall on? If yes, welcome. If not? You get quarantined. No internet. No access to sensitive stuff. Just enough to download updates. Think of it as a health check before letting you into the building.
4. DAI: Finally, once they’re in, DAI watches every ARP packet using the binding table from DHCP Snooping.

So it’s not just random features slapped on. It’s a sequence:
Physical access → IP legitimacy → Device health → Traffic integrity.