WEEK 4 ASSIGNMENT 1:

VLANs AND SECURE SWITCH CONFIGURATION

RENE OLUOCH

CS-CNS09-25030

Contents

Introduction

In this lab, I undertook a comprehensive hands-on exercise to design, configure, and secure a network using Cisco switches and routers. The focus was on implementing Virtual LANs (VLANs), inter-VLAN routing, trunking protocols, DHCP services, and critical Layer 2 security features. Through a structured approach, I configured devices, assigned IP addressing, created and verified VLANs, enabled 802.1Q trunking, and deployed advanced switch security mechanisms such as port security, DHCP snooping, and BPDU guard. This practical experience not only reinforced my understanding of theoretical concepts but also enhanced my troubleshooting skills and my ability to manage real-world networking environments securely and efficiently.

Cabling the network

I started by physically cabling the network according to the provided topology. I connected the router's GigabitEthernet0/0/1 interface to Switch S1's FastEthernet0/5 port, which serves as the gateway for devices on the network. Then, I connected Switch S1's FastEthernet0/1 to Switch S2's FastEthernet0/1 — this link was later configured as a trunk. For end devices, PC-A was connected to S1's F0/6, and PC-B to S2's F0/18. Once connected, I powered on all devices and verified that the link lights were green to confirm physical connectivity.

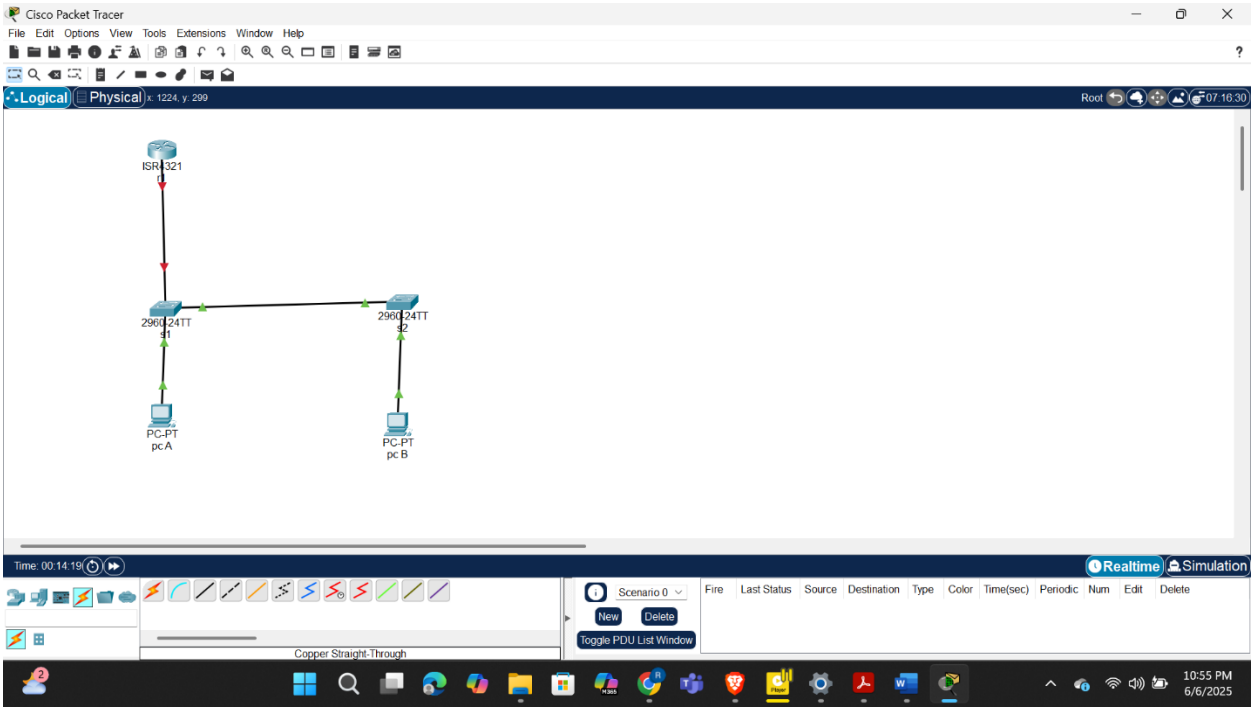


Fig 1.0 setting up the topology

Configuring Router R1

On the router, I began by changing its hostname to R1 for clarity. I then disabled DNS lookup to avoid unnecessary delay from mistyped commands triggering name resolution.

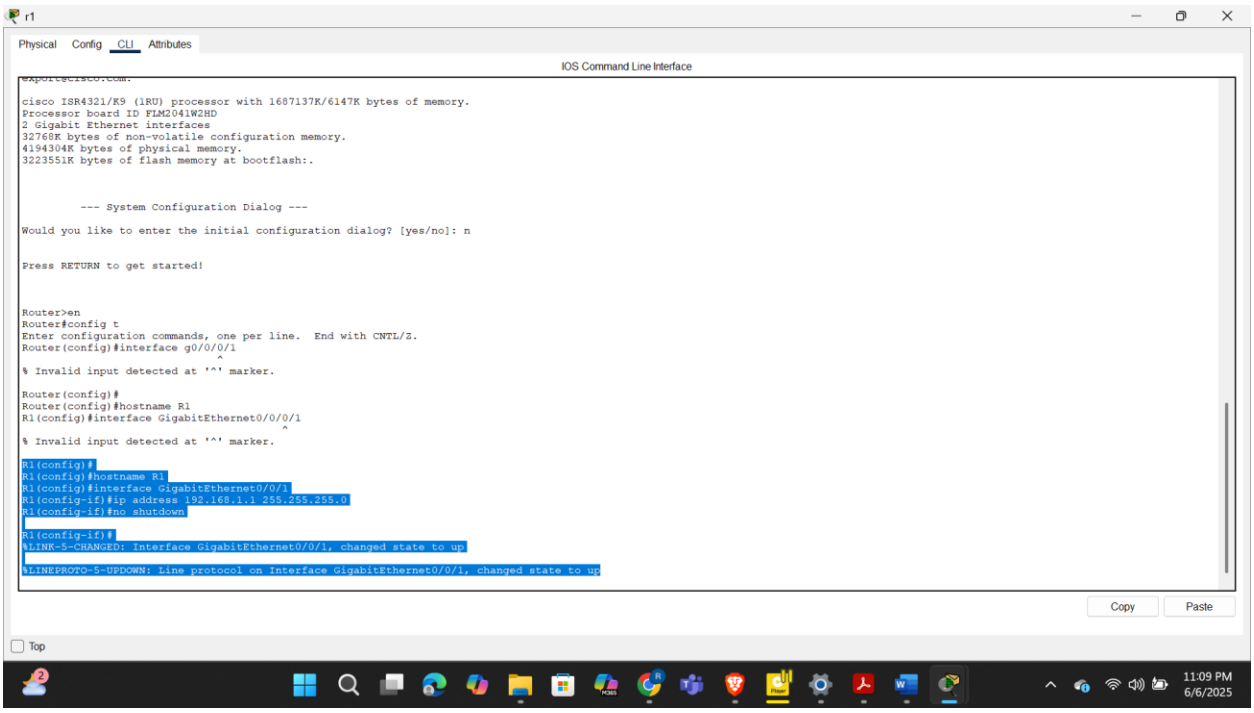
Next, I configured IP address exclusions to reserve the router's own IP address (192.168.10.1) and the two switch management addresses (192.168.10.201 and .202) so they wouldn't be handed out by the DHCP server.

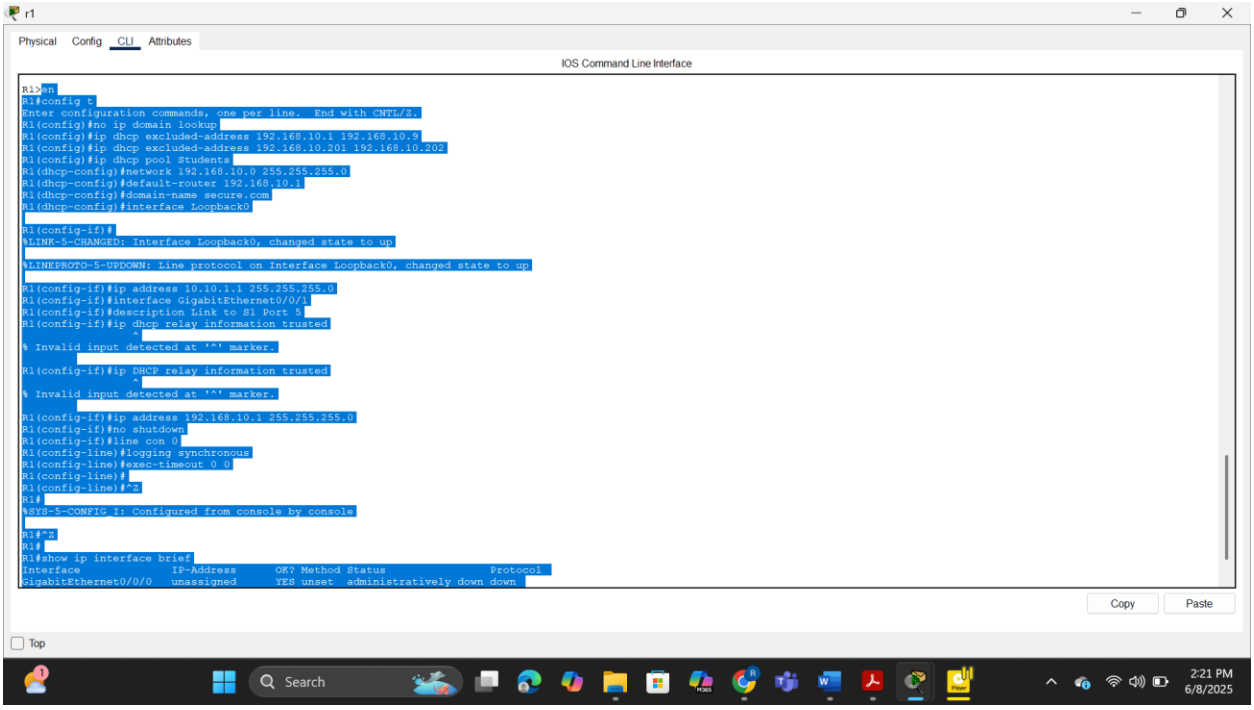
I then created a DHCP pool named "Students", assigning it the 192.168.10.0/24 subnet. I specified the default gateway as 192.168.10.1 and assigned a domain name "secure.com" for DHCP clients. This configuration allows dynamic IP assignment to both PCs.

After that, I created a Loopback0 interface with the IP address 10.10.1.1/24. This interface simulates a remote internal network or a test point for routing verification.

Finally, I configured the GigabitEthernet0/0/1 interface with the IP address 192.168.10.1, enabled it, and marked it as a trusted source for DHCP relay. This interface connects to Switch S1 and serves as the default gateway for the rest of the network.

After the configuration, I verified interface statuses to ensure both G0/0/1 and Loopback0 were up and operational.





Figs 1.1 configuring the router

Configuring VLANs on Switches

For both switches, I first renamed their hostnames to S1 and S2 to differentiate them clearly in the CLI and network monitoring tools. I disabled DNS lookup to prevent delays from mistyped commands being interpreted as hostnames.

I then added interface descriptions on both switches to label the purpose of each port — for example, marking F0/1 as "Link to S2" and F0/5 as "Link to R1." This step improves readability and management of the configuration, especially during troubleshooting.

After that, I set the default gateway of both switches to 192.168.10.1 — the router's IP — so that they can communicate with devices outside their VLAN. This is crucial for remote management (e.g., via SSH or Telnet).

Configuring VLAN 10

I created VLAN 10 on both S1 and S2 and assigned it the name "Management". This VLAN is used for management purposes, including accessing the switch remotely. Using a dedicated VLAN helps in segregating administrative traffic from user data traffic, improving security and monitoring.

Configuring the SVI for VLAN 10

I then configured a Switched Virtual Interface (SVI) for VLAN 10 on both switches. On S1, I assigned the IP address 192.168.10.201, and on S2, 192.168.10.202. I added a description to each SVI indicating it is for management purposes and then enabled the interfaces. These IP addresses allow me to remotely access and manage each switch through VLAN 10.

Configuring VLAN 333 (Native VLAN)

Next, I created VLAN 333 on both switches and named it "Native." This VLAN is assigned as the native VLAN for trunk ports between S1 and S2. The native VLAN is used for untagged traffic, and setting a custom native VLAN (rather than leaving it at VLAN 1) adds a layer of security to prevent VLAN hopping attacks.

Configuring VLAN 999 (Parking Lot)

Lastly, I configured VLAN 999 on both switches and named it "ParkingLot." This VLAN is used to isolate unused ports, effectively neutralizing them from unauthorized access or accidental connections. Assigning unused ports to an unused VLAN and disabling them is a common security best practice.

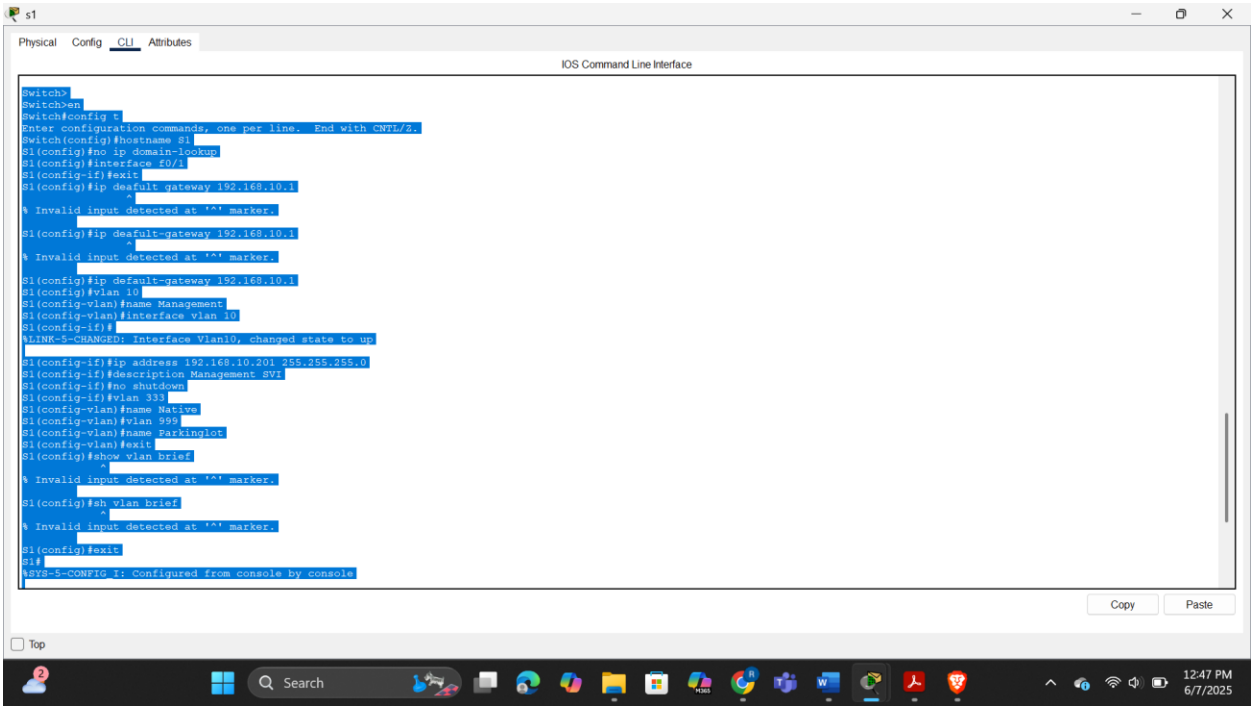


Fig 1.2 configuring s1

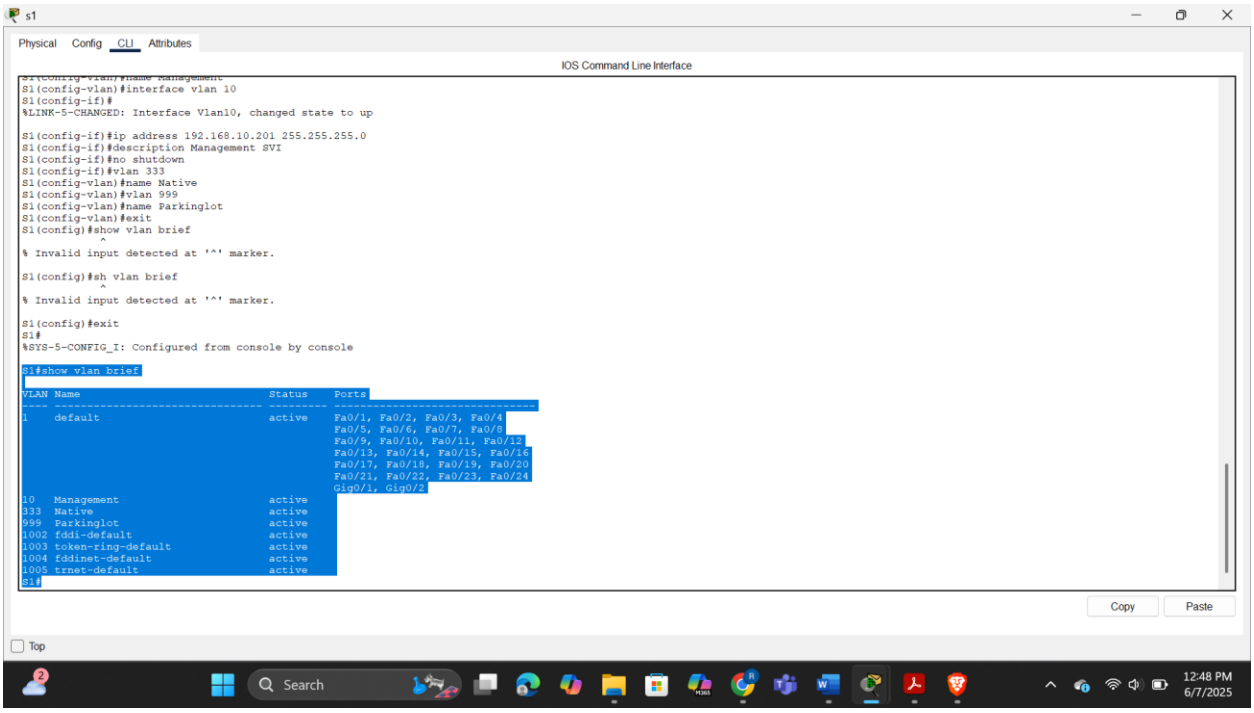


Fig 1.3 s1 vlan interfaces

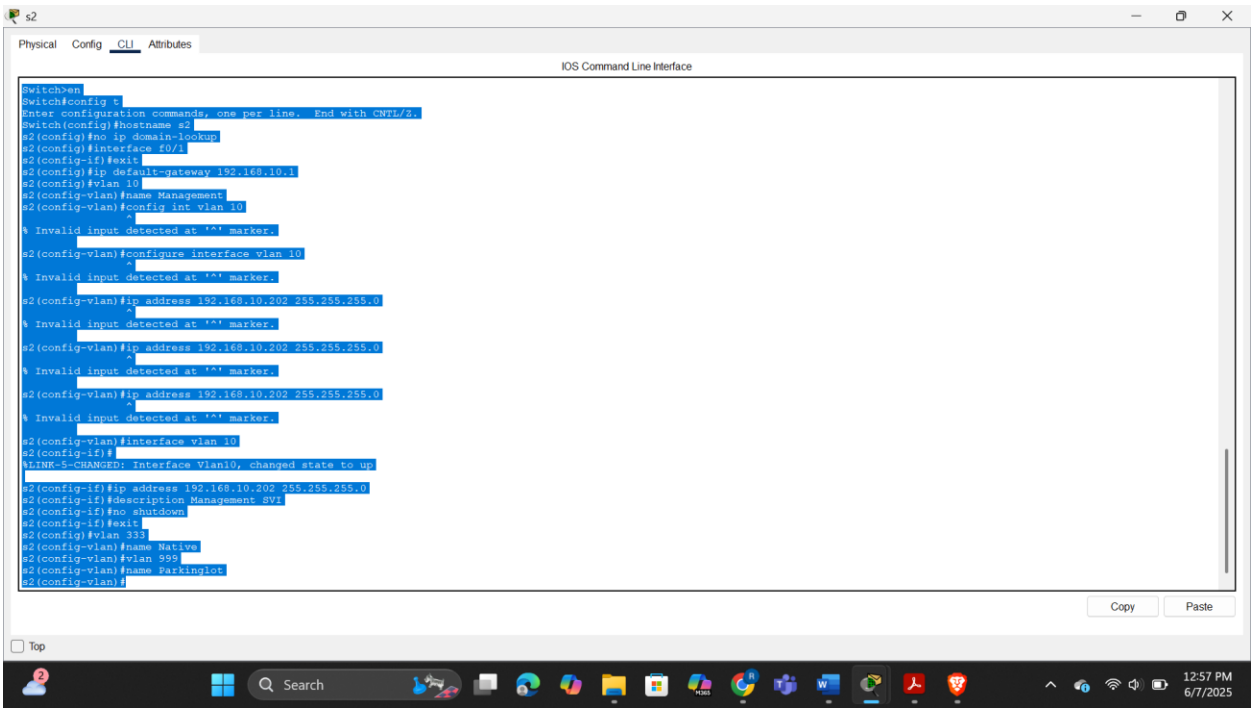


Fig 1.4 configuring s2

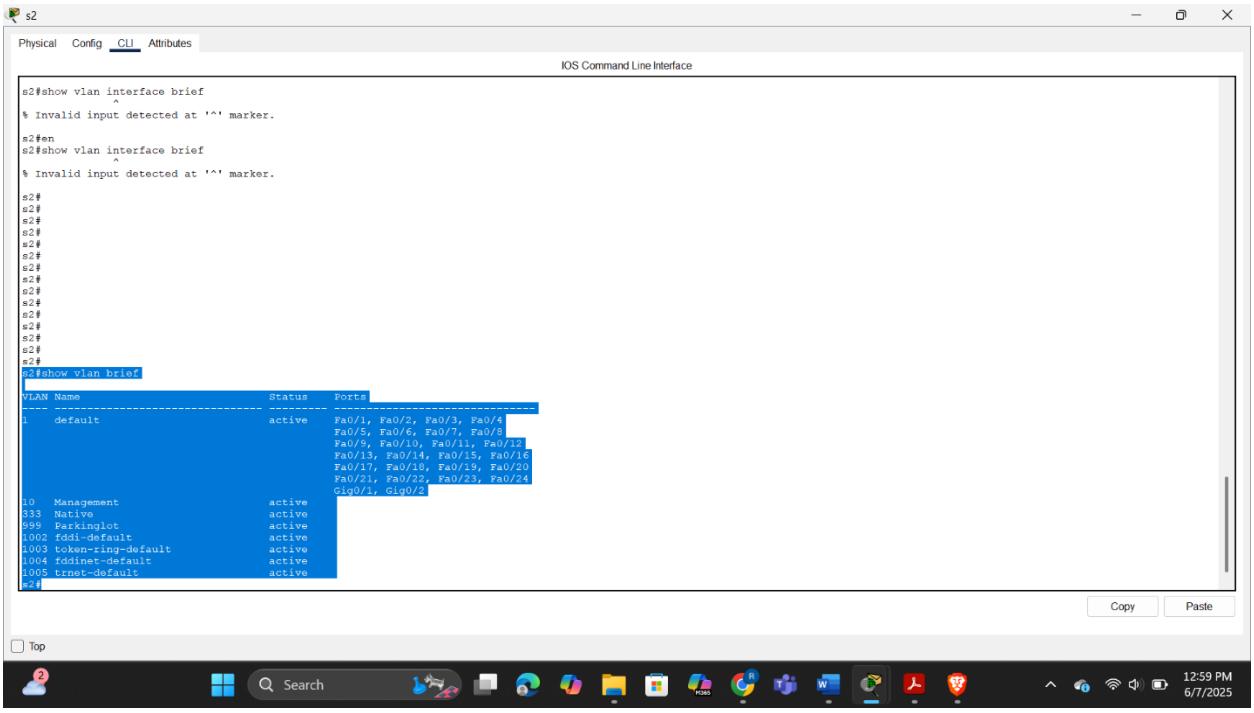


Fig 1.5 s2 vlan interfaces

Configuring Switch Security.

Implement 802.1Q trunking.

To enable VLAN communication across switches, I configured 802.1Q trunking on the inter-switch link (F0/1) between S1 and S2. I set both interfaces to trunk mode and assigned VLAN 333 as the native VLAN. This ensures that untagged frames (often management or legacy traffic) are handled securely on a dedicated VLAN.

To further harden the trunk, I disabled DTP (Dynamic Trunking Protocol) on both trunk interfaces using the nonegotiate option. This forces the trunk configuration to be static, preventing a rogue device from negotiating trunking automatically — a common attack vector.

Afterward, I verified trunking using the show interfaces trunk command and confirmed that VLANs 1, 10, 333, and 999 were allowed and active on the trunk link, and the native VLAN was correctly set to 333.

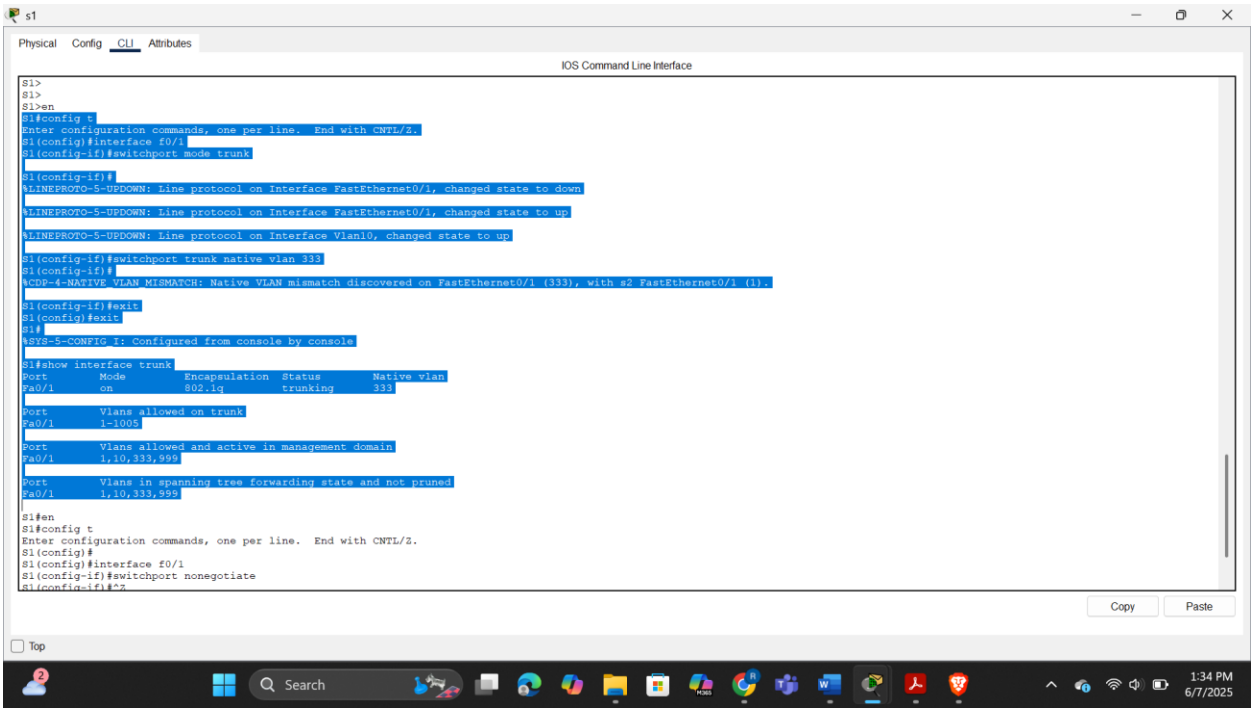


Fig 1.6 implementing 802.1Q trunking to s1

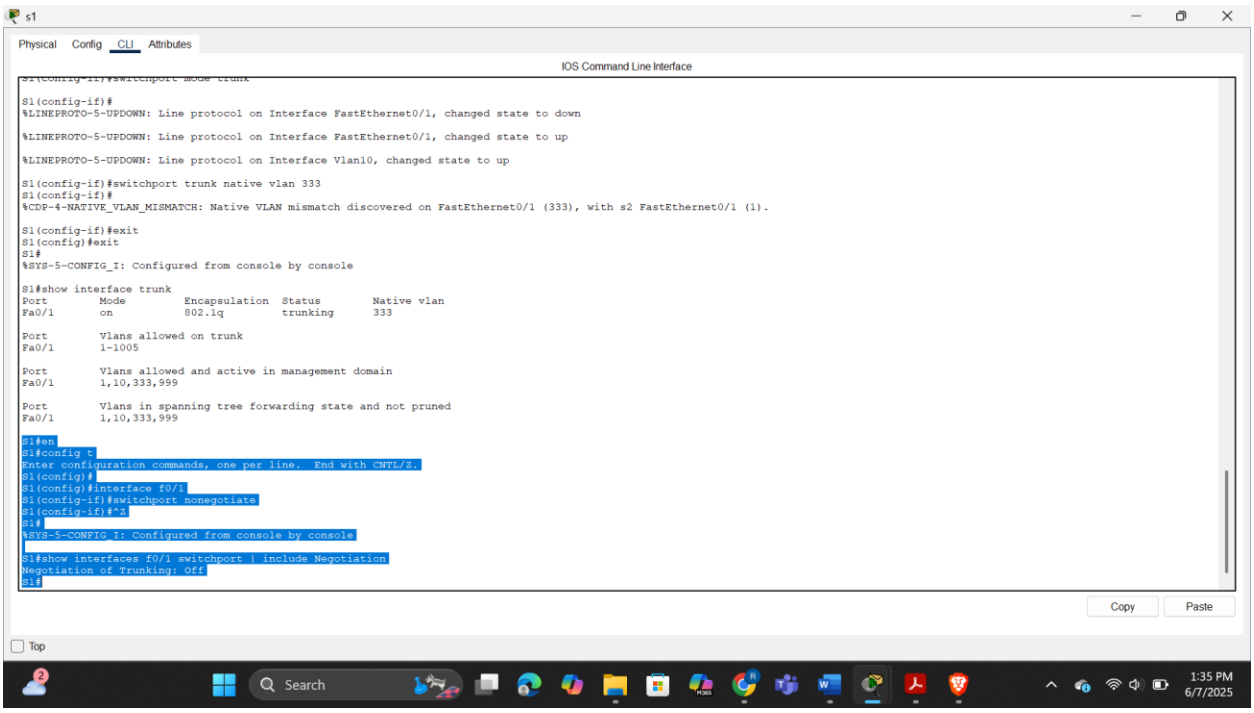


Fig 1.7 disabling dtp on s1

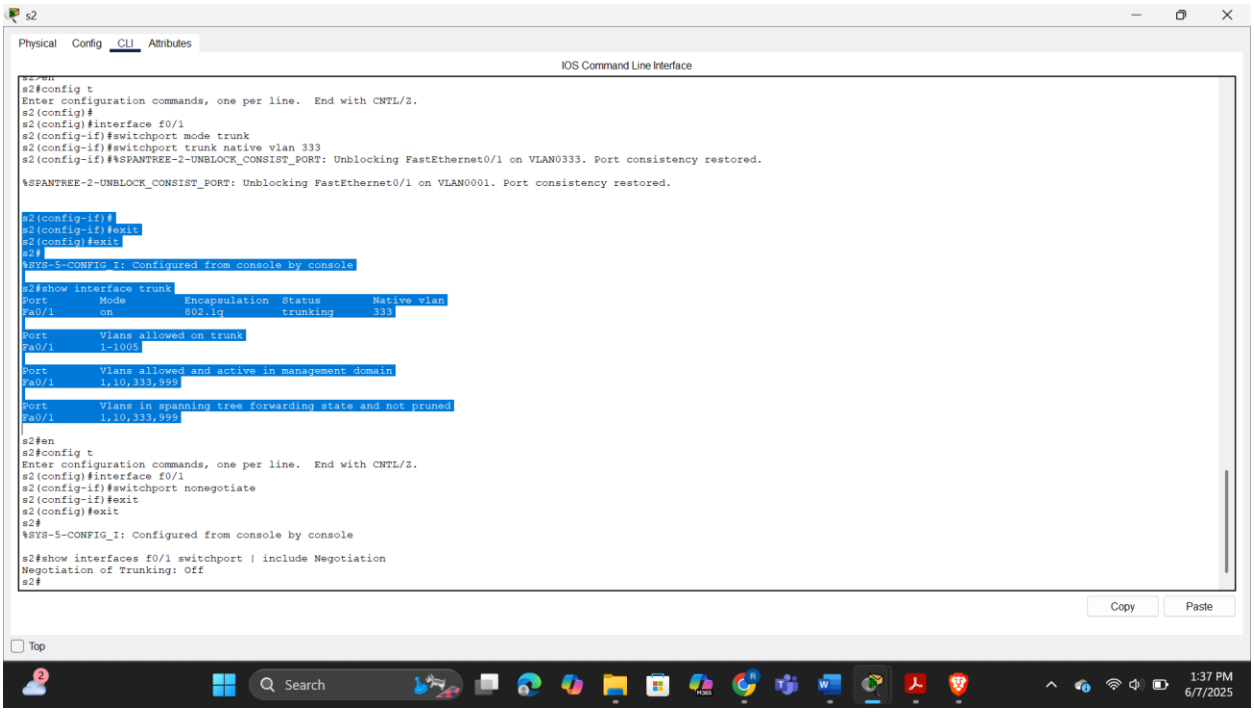


Fig 1.8 implementing 802.1Q trunking to s2

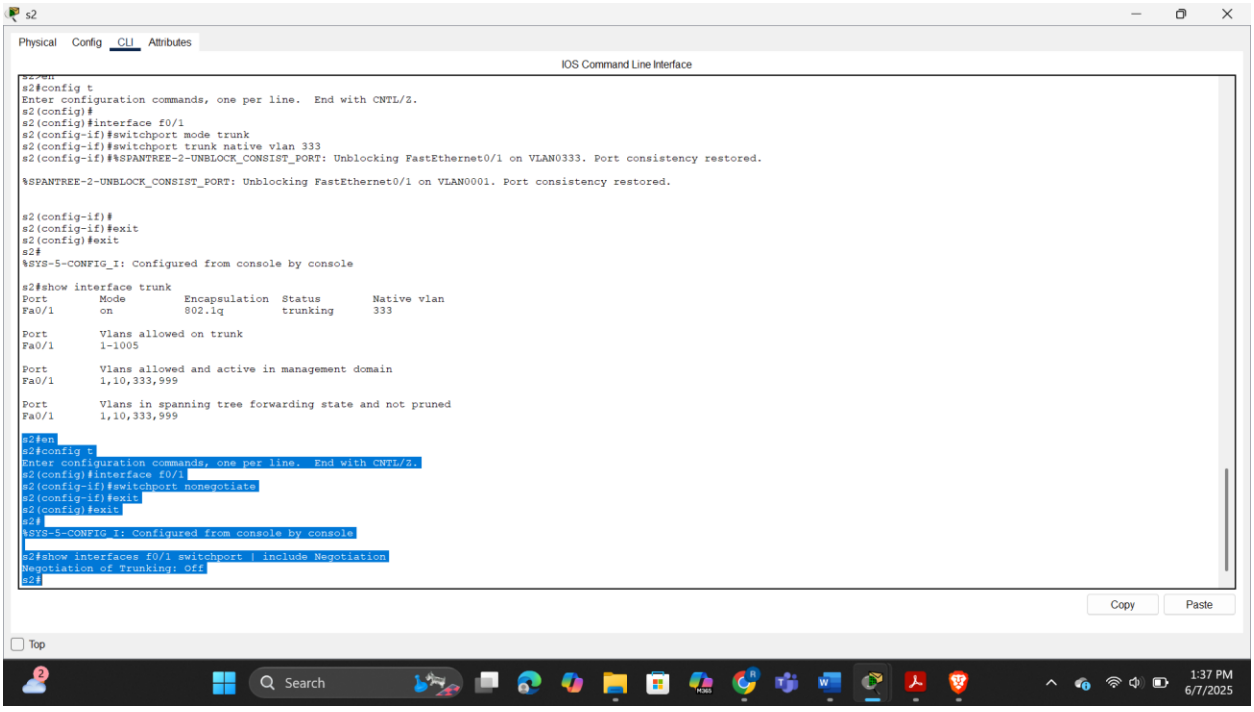


Fig 1.9 disabling dtp on s1

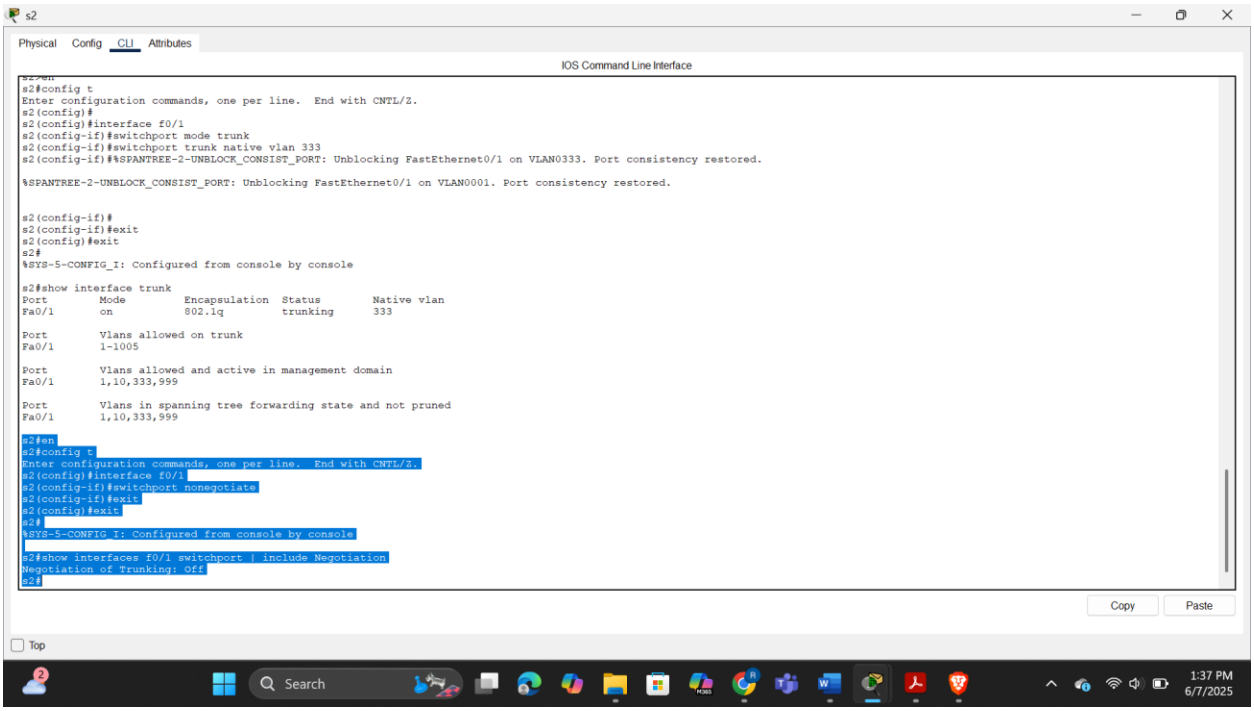


Fig 2.0 Disabling dtp on s2

After completing the trunk implementation, I proceeded to verify the entire VLAN and trunk configuration to ensure that all settings were applied correctly and were functioning as expected. Using the show vlan brief command on both switches, I confirmed the presence of VLAN 10, VLAN 333 (named Native), and VLAN 999 (named ParkingLot). VLAN 10 appeared as active, and the appropriate ports were correctly assigned to it. VLAN 333 was confirmed as the native VLAN, and VLAN 999 was observed to be assigned to unused ports, serving its purpose as the designated parking lot VLAN. I then issued the show interfaces trunk command and verified that 802.1Q trunking was operational between the switches S1 and S2, with VLAN 333 correctly set as the native VLAN and VLAN 10 allowed across the trunk. Finally, the show interfaces status command allowed me to confirm that all interfaces were in the expected state, matching the configuration plans. This verification step gave me confidence that the VLAN segmentation and inter-switch communication were implemented correctly and securely.

Configuring access points

I configured access ports for the end devices. On Switch S1, I set F0/5 and F0/6 as access ports in VLAN 10, used for PC-A and the router connection respectively. On Switch S2, I configured F0/18 as an access port for PC-B, also in VLAN 10. Assigning these ports to the correct VLAN ensures that client devices can obtain IP addresses via DHCP and communicate within the correct broadcast domain.

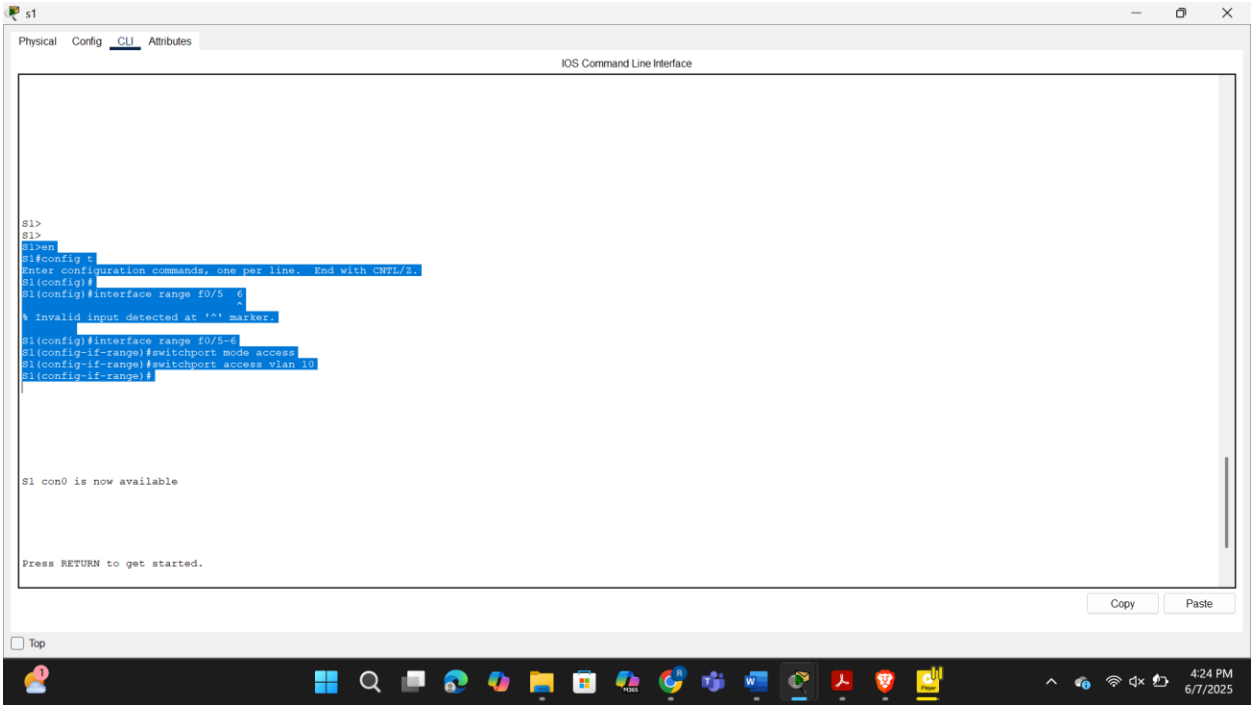


Fig 2.1 Config access point s1

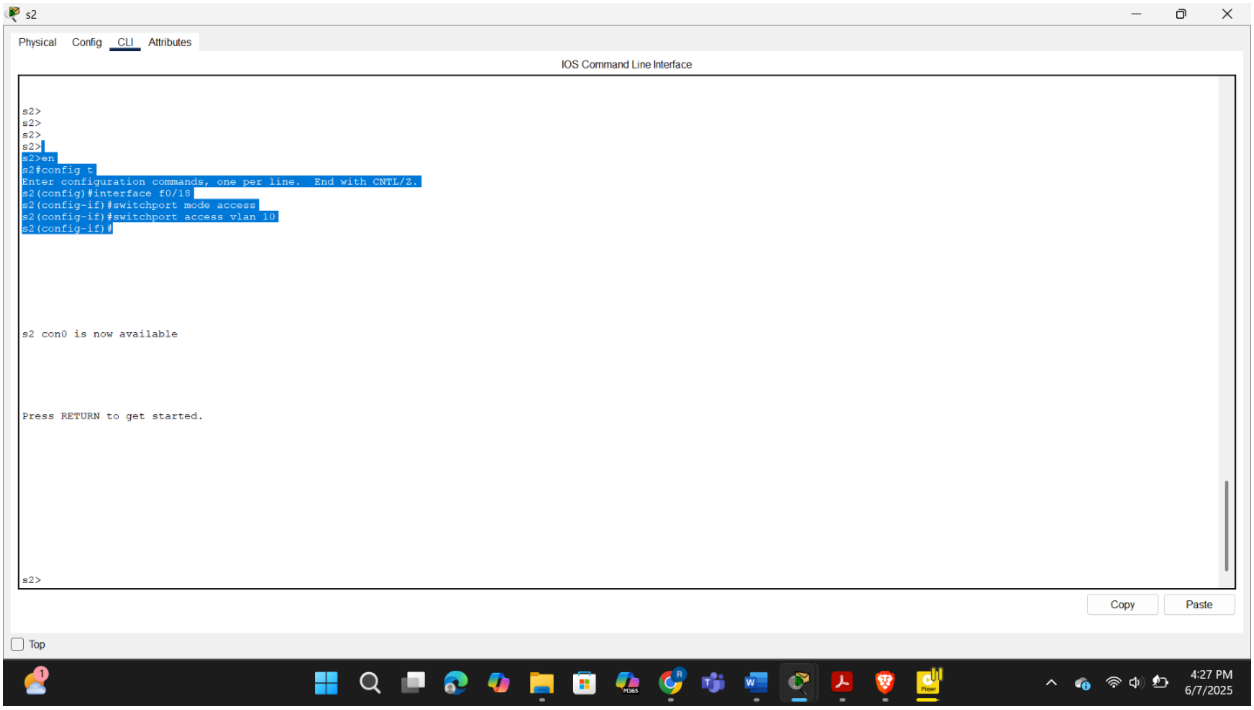


Fig 2.1.1Config access point s2

Securing and disabling unused switchports

To improve security, I identified all unused switchports on both S1 and S2. I moved these ports into VLAN 999 (ParkingLot), which is a non-functional VLAN. This effectively isolates unused ports and prevents them from passing traffic. I then administratively shut down these interfaces to prevent unauthorized access or rogue device connections.

After this, I verified port statuses using the show interfaces status command. All unused ports appeared as "disabled" and assigned to VLAN 999, as expected.

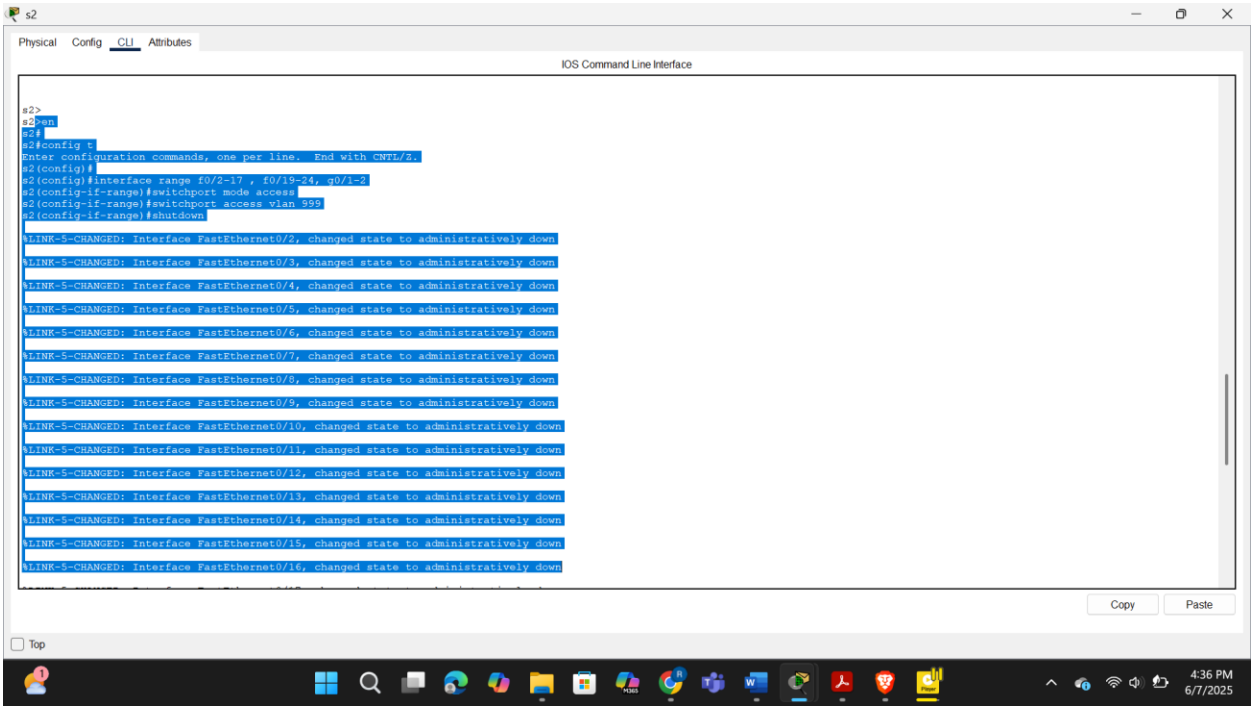


Fig 2.2 Disabling unused sw ports s2

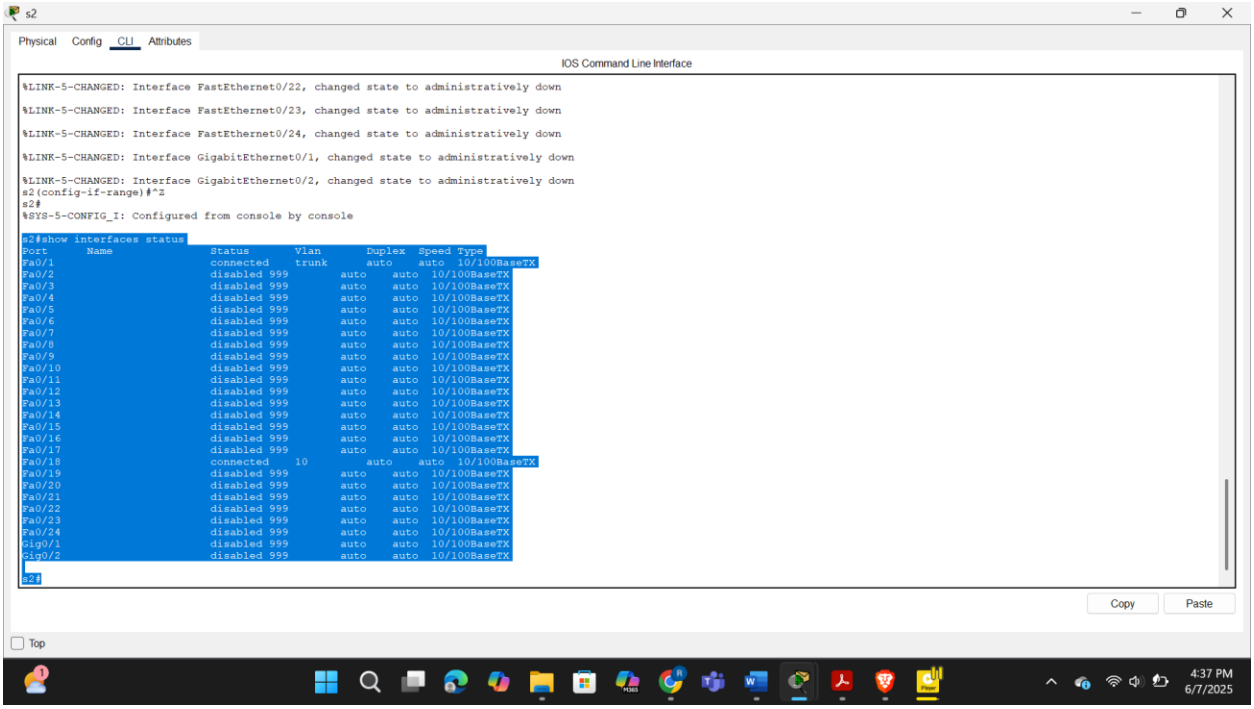


Fig 2.3 Verifying the disabled ports

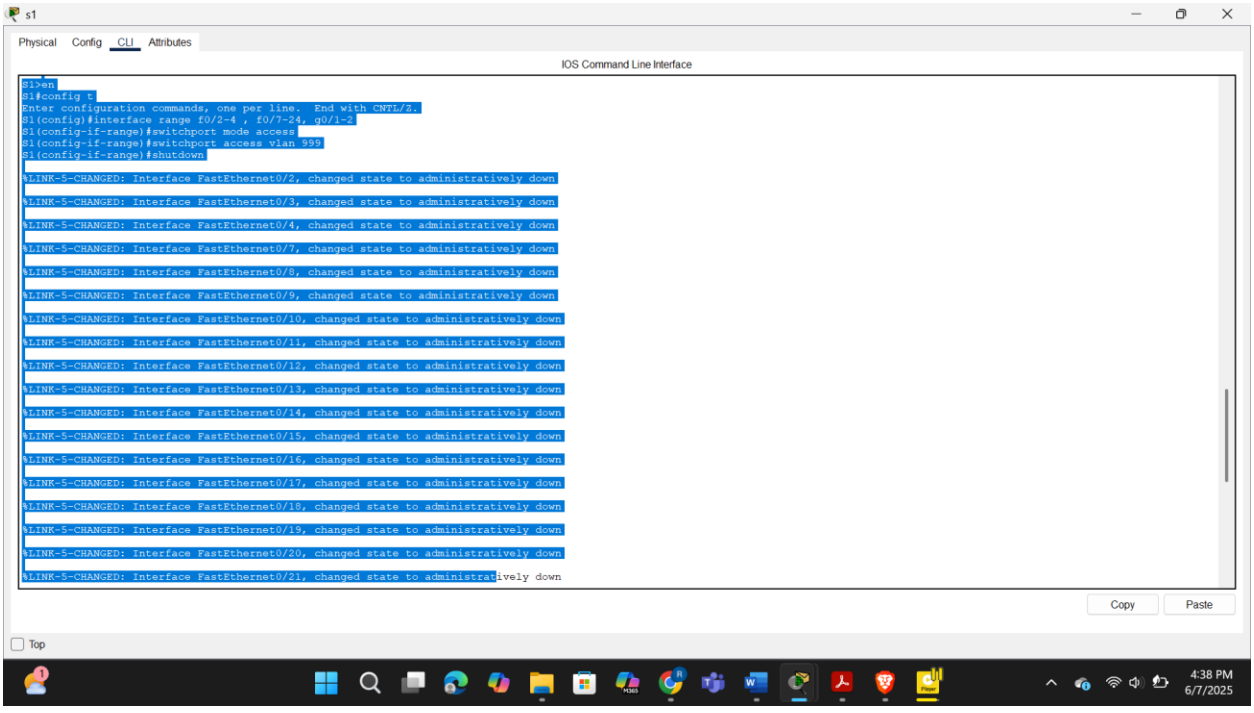


fig 2.4 Disabling unused ports s1

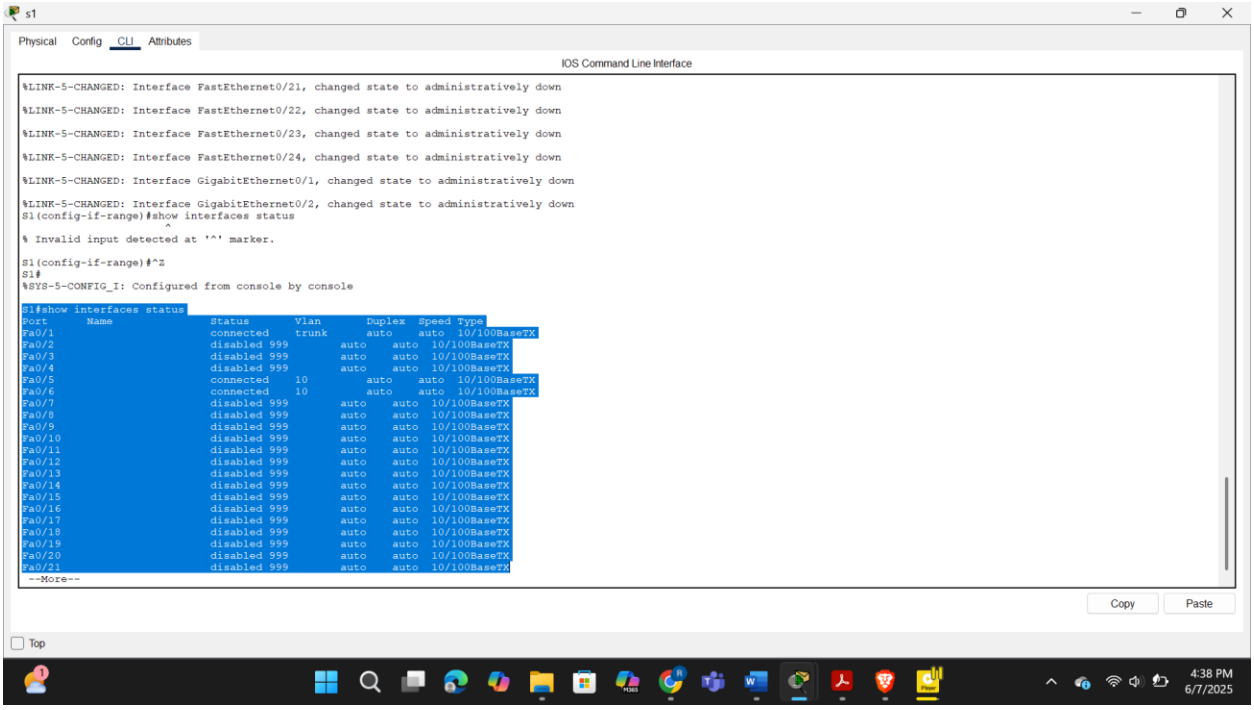


Fig 2.5 Verifying disabled s1 ports

Implementing port security features.

To prevent MAC flooding or unauthorized device access, I enabled port security on F0/6. I configured the port to allow a maximum of 3 MAC addresses, which is helpful for scenarios like VoIP phones + workstations sharing a port.

I set the violation mode to restrict, which silently drops unauthorized traffic but logs the violation. I also enabled MAC address aging — specifically inactivity-based aging, which removes MACs from the table after 60 minutes of no activity.

I verified this using the show port-security command, which confirmed that port security was active, the violation mode was restrict, and the aging policy was in place.

- On S2 (Port F0/18 for PC-B):

On S2, I enabled sticky MAC address learning on F0/18, allowing the switch to dynamically learn MAC addresses and write them into the running configuration.

I limited the port to two MAC addresses, and configured the violation action as protect, meaning any unknown MAC will be silently dropped, without shutting down the port or generating alerts.

This is a stricter form of control suitable for stable endpoint environments. After setup, I validated the configuration and confirmed that the MAC address for PC-B was stored as a sticky entry.

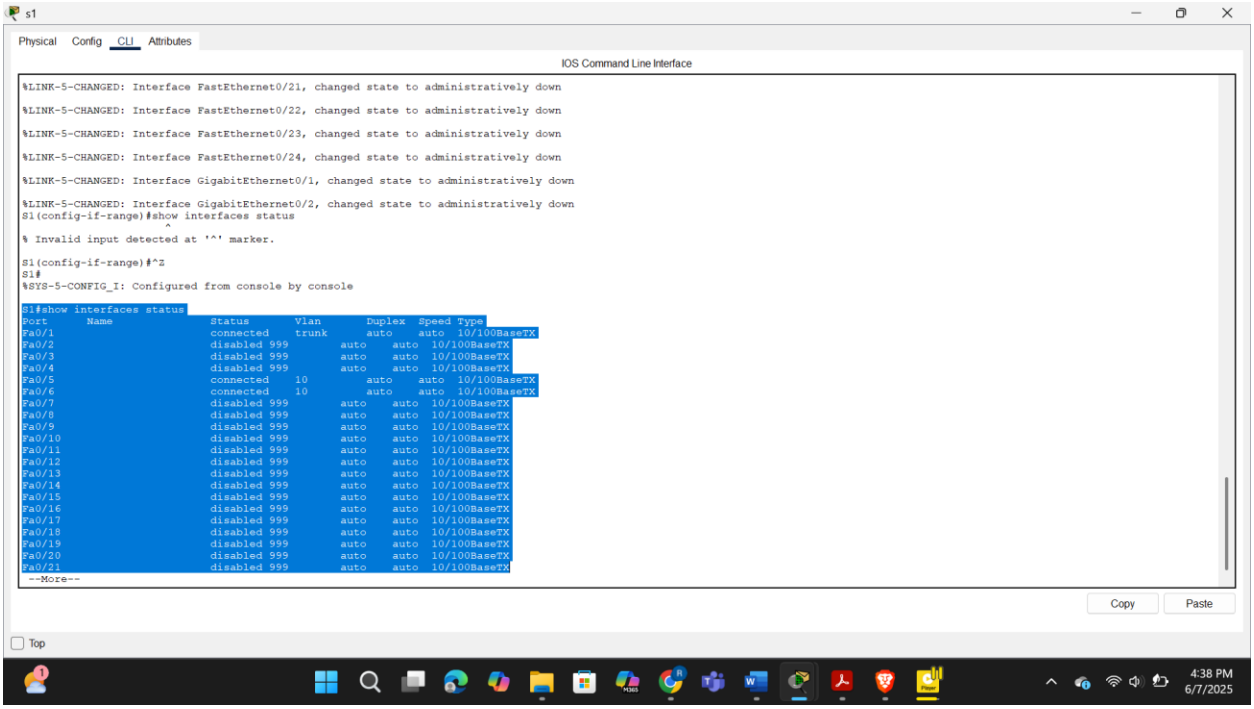


Fig 2.6 Showing port security

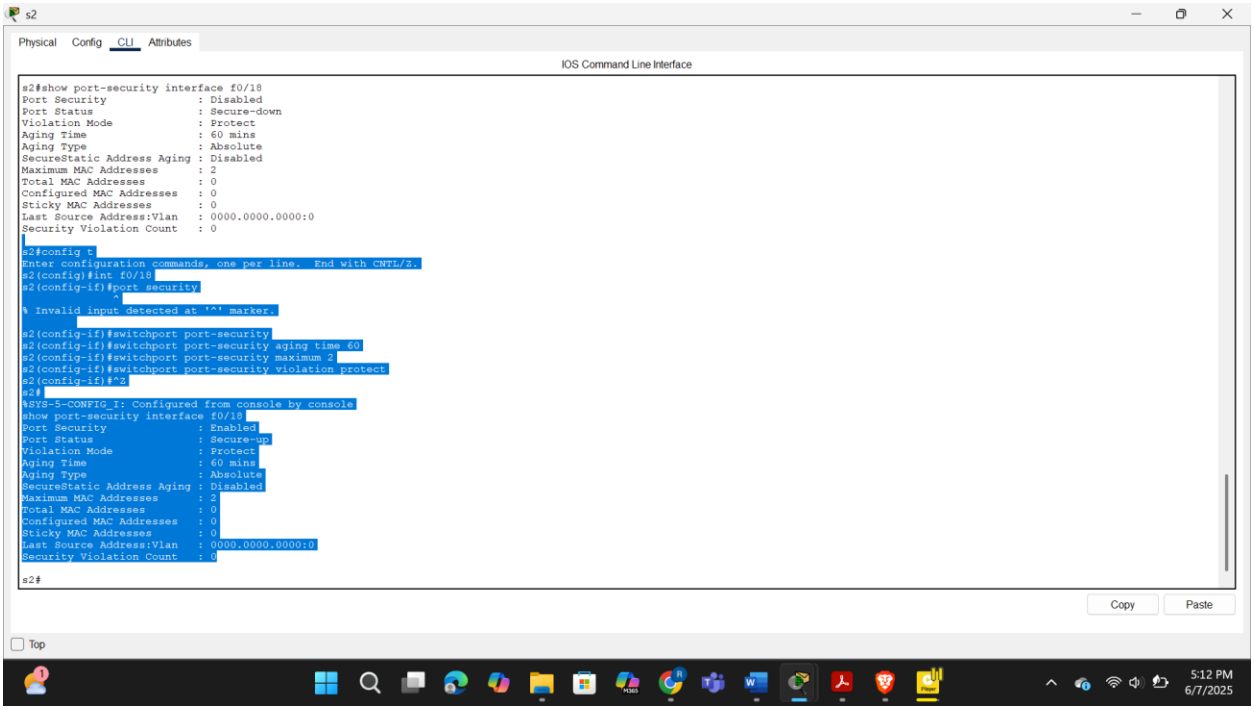


Fig 2.7 Enabling port security and verifying s2

Implementing dhcp snooping security

To defend against rogue DHCP servers, I enabled DHCP snooping globally and specifically on VLAN 10, which is the client VLAN.

I marked the trunk port (F0/1) on S2 as a trusted port, because it connects to other infrastructure devices. The access port F0/18 (connected to PC-B) was left untrusted and limited to 5 DHCP messages per second. This prevents DHCP starvation or exhaustion attacks.

I tested the setup by releasing and renewing PC-B's IP address, which succeeded, indicating that DHCP snooping was allowing legitimate traffic. I confirmed that the MAC and IP binding appeared in the DHCP snooping table, verifying the feature was working correctly.

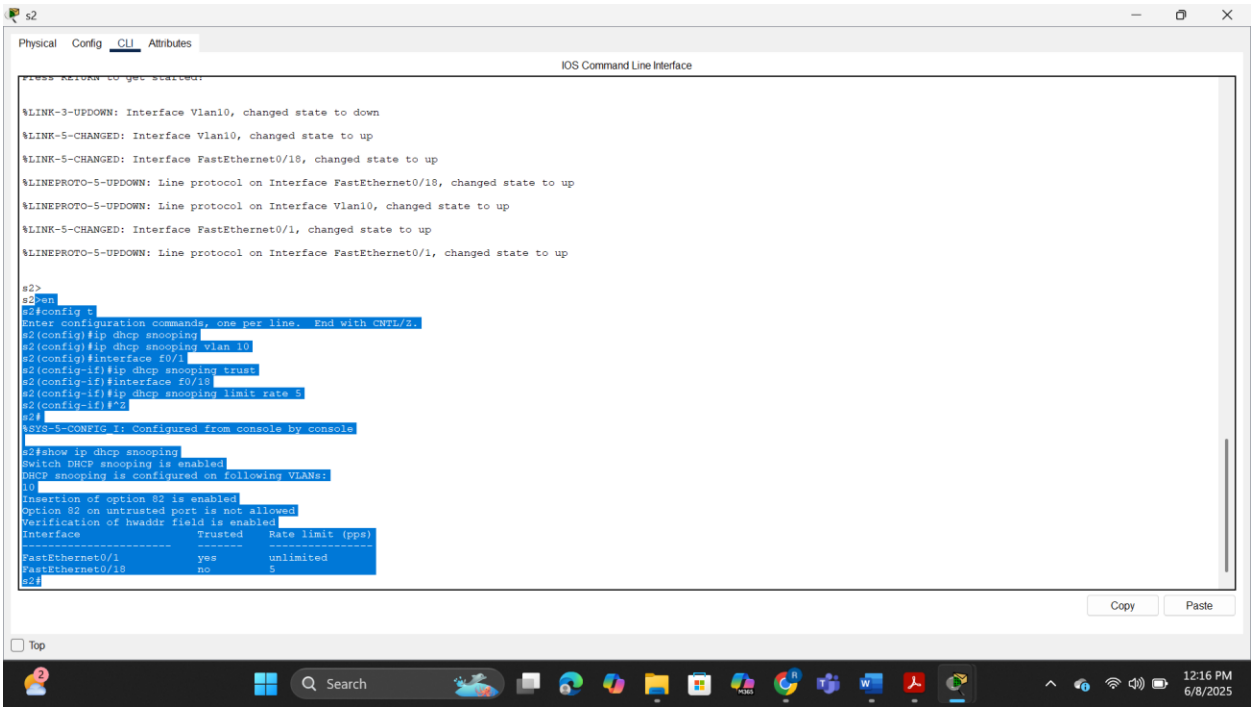


Fig 2.8 implementing dhcp snooping security

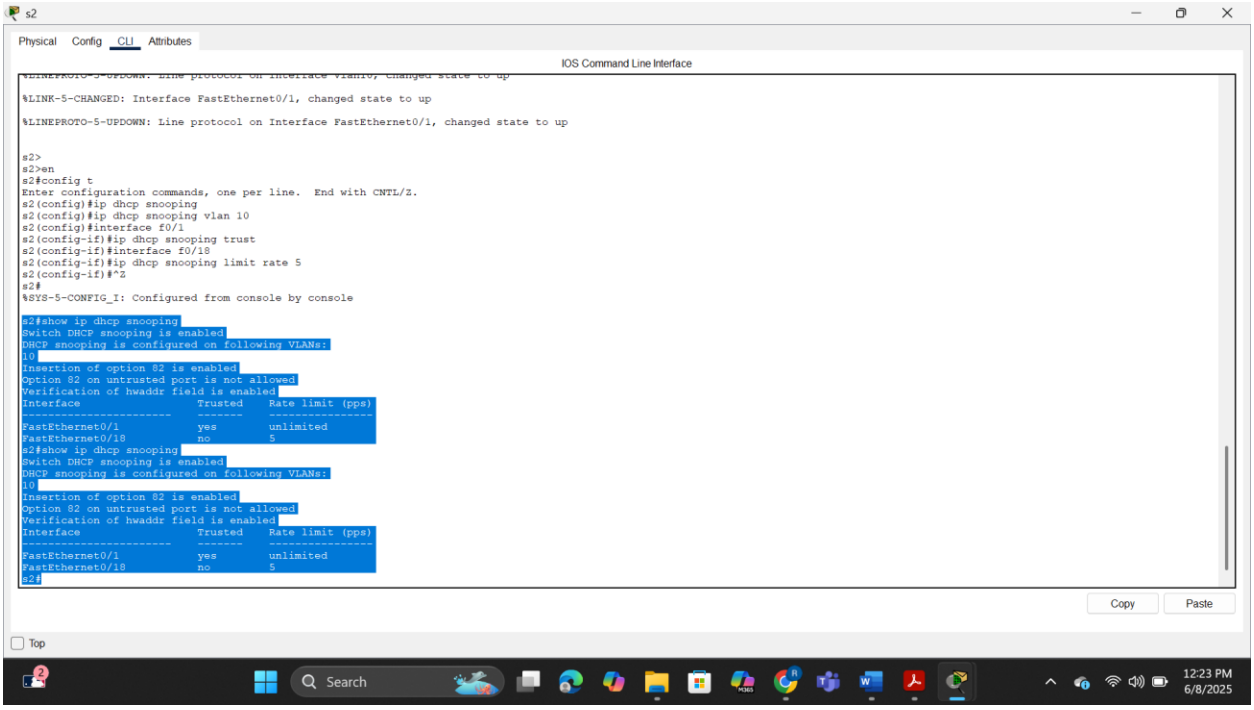


Fig 2.9 verifying implemented dhcp snooping security

Implementing PortFast and BPDU guard.

To reduce startup delay for access ports, I enabled PortFast on the access interfaces in use (F0/5 and F0/6 on S1, F0/18 on S2). This allows ports to immediately enter the forwarding state, rather than wait through STP's learning/listening phases.

I also enabled BPDU Guard on the same access ports. This adds protection against rogue switches — if a BPDU is received on any of these ports, the port will automatically shut down to protect the spanning tree topology.

I verified both features using show spanning-tree interface detail, which showed that PortFast and BPDU Guard were enabled, and the ports were in forwarding state.

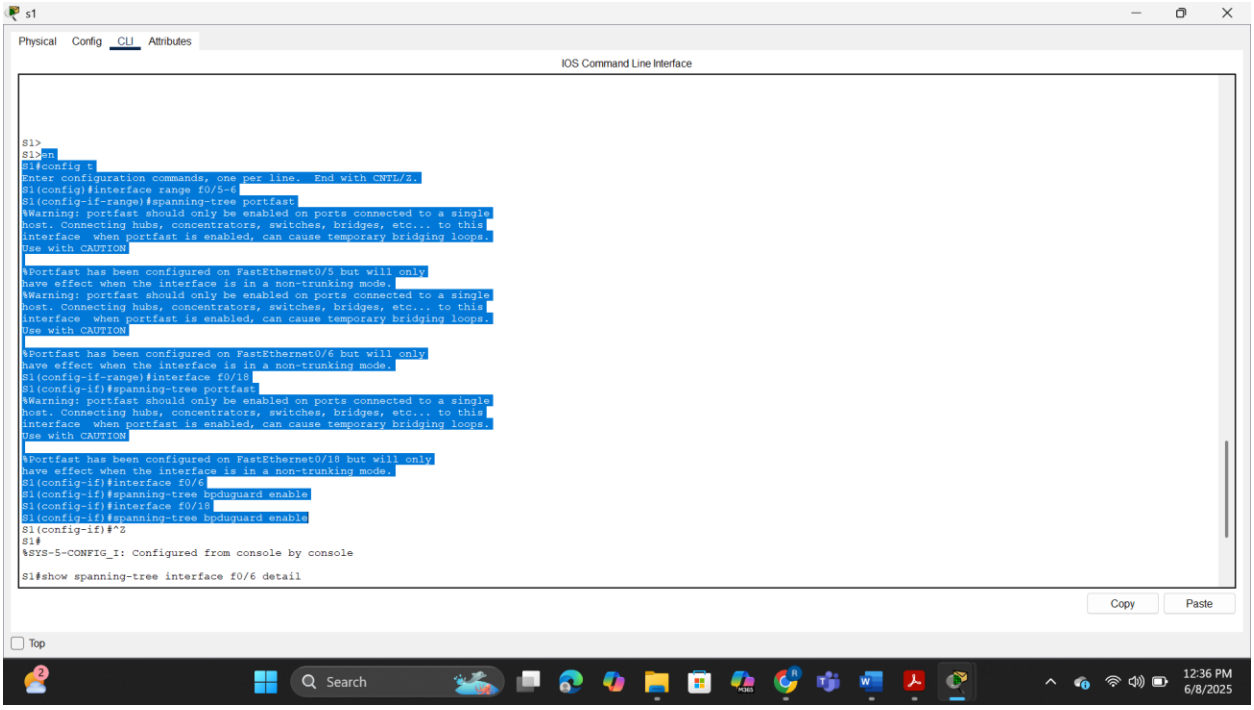


Fig 3.0 Implementing portFast and bpdu guard

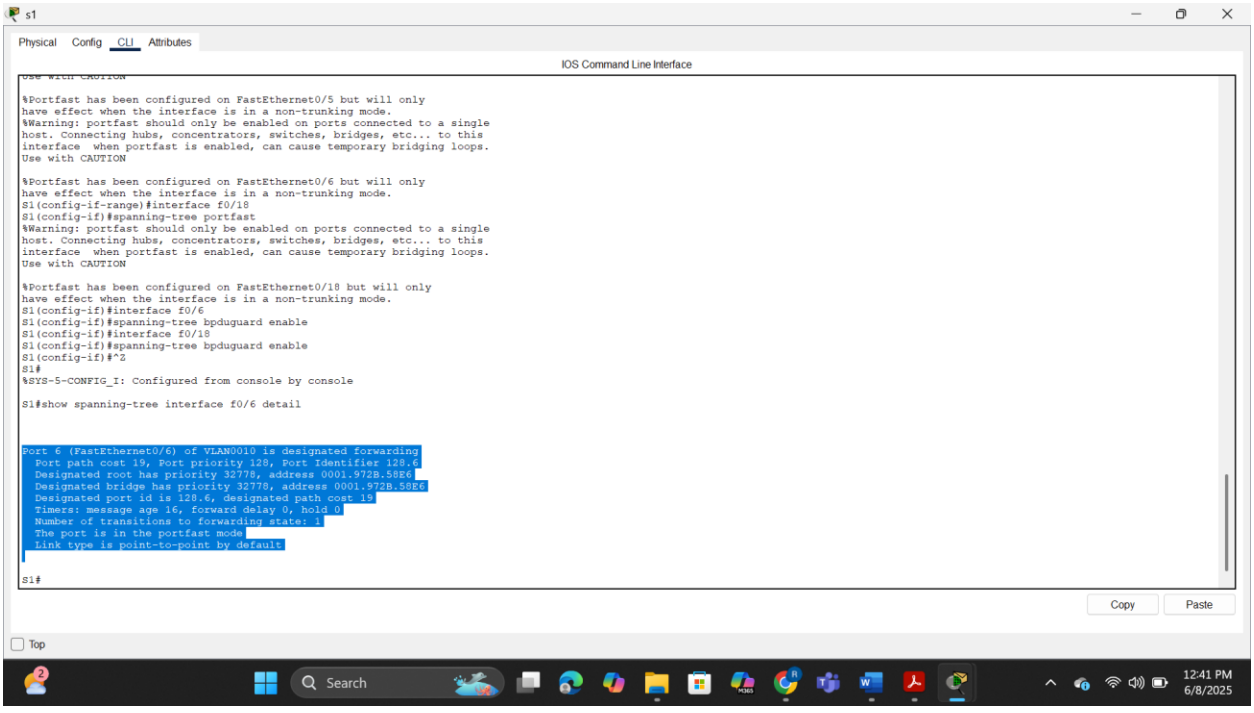


Fig 3.1 Verifying implemented portFast and bpdu guard

Verifying end-to-end connectivity.

Finally, I verified end-to-end IP connectivity by pinging from PC-A to PC-B, from both PCs to the router gateway (192.168.10.1), and from the switches to the loopback interface (10.10.1.1). All pings were successful.

This confirmed that VLANs, trunking, DHCP, and routing were properly configured, and security features did not interfere with legitimate traffic.

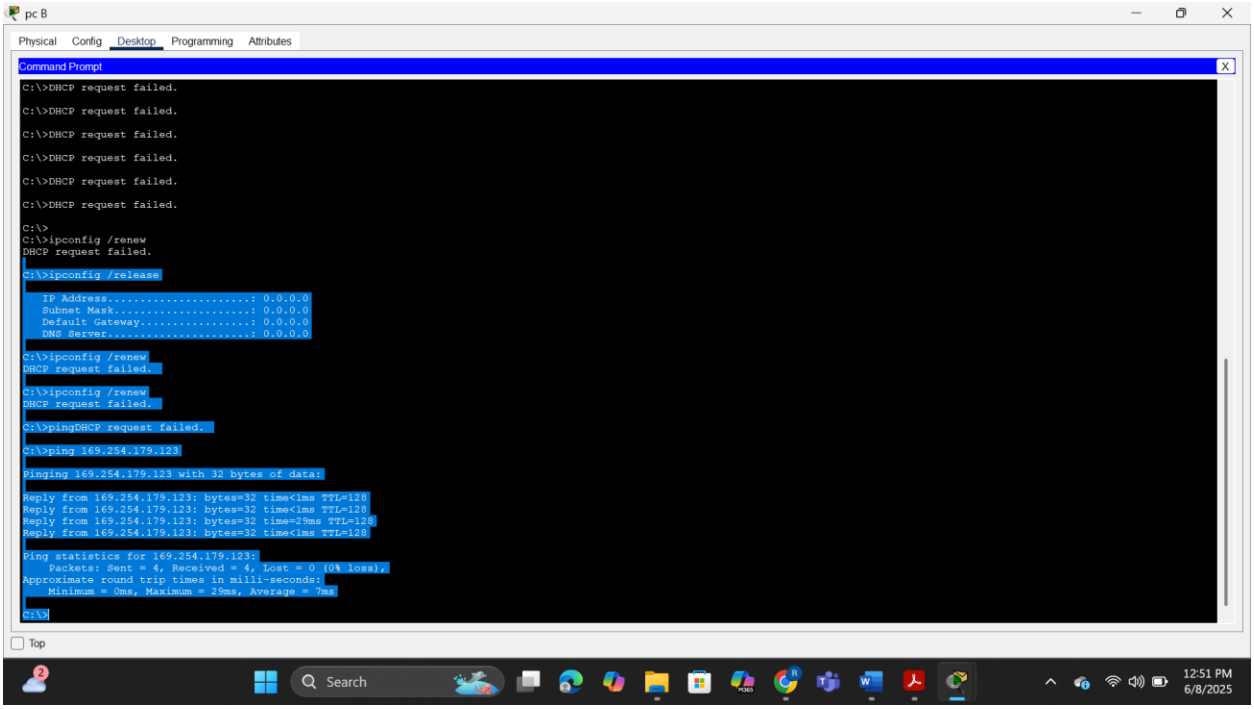


fig 3.2 Pinging pc b from pc a

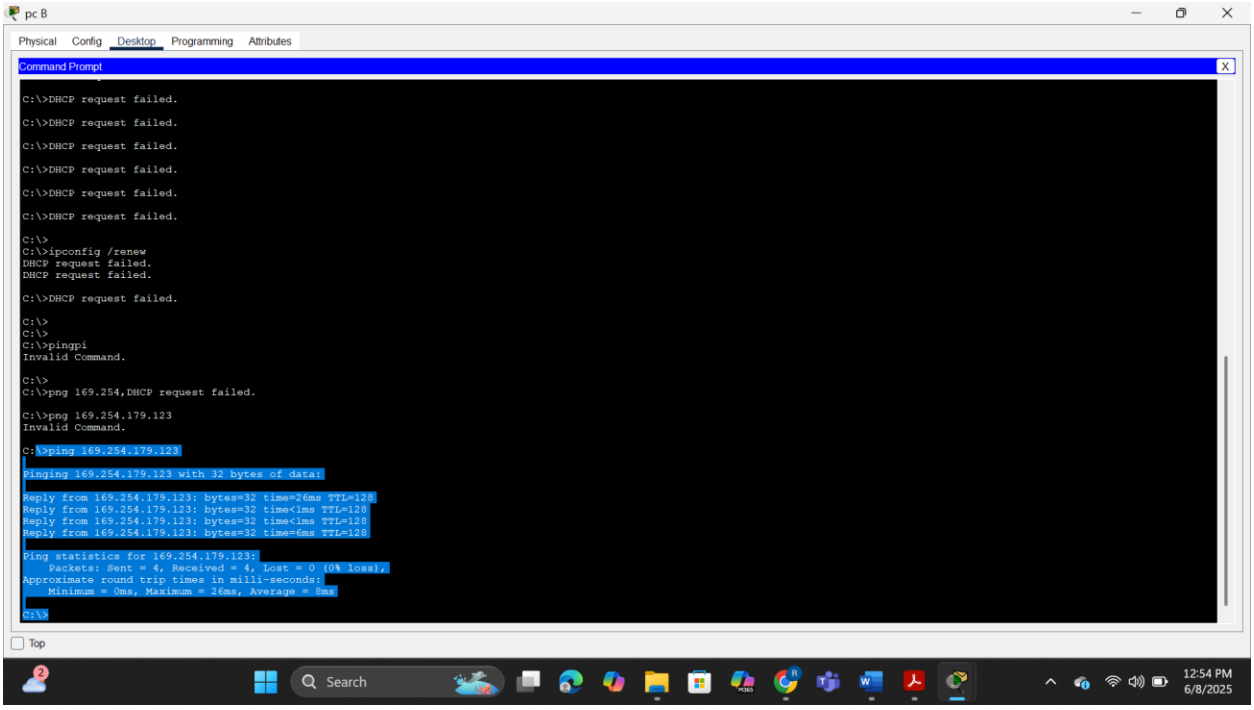


Fig 3.4 Pinging pc b from pc a

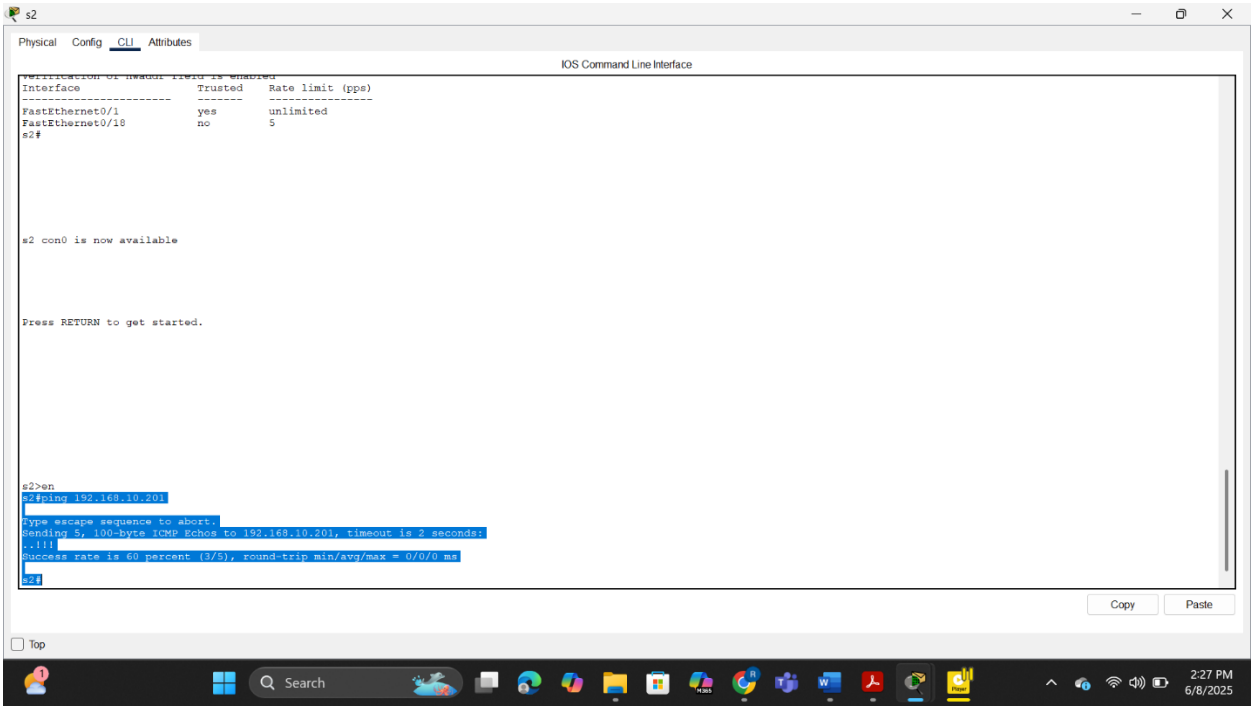


Fig 3.5 Pinging s1 from s2

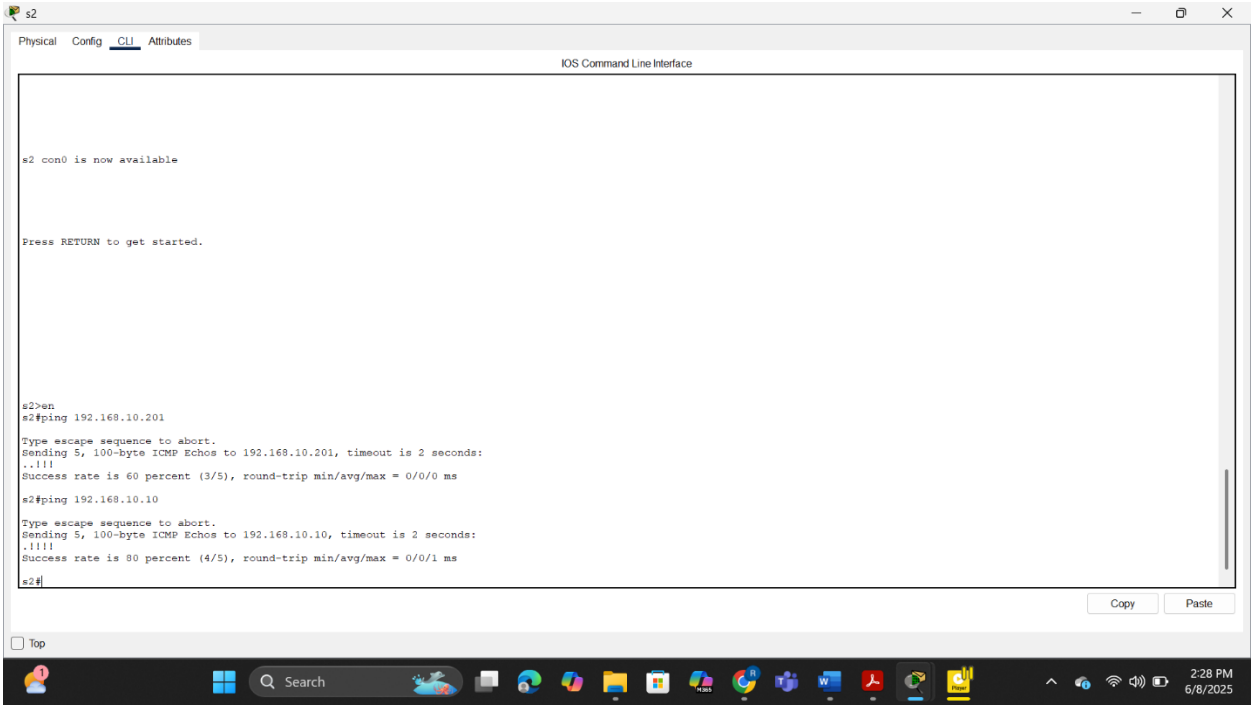


Fig 3.6 Pinging pc a from s2

Reflection

While configuring port security with sticky MAC learning on S2 (specifically on interface F0/18), I noticed that the remaining age for the sticky MAC address did not show a timer value. This was an interesting discovery, and it helped me learn that sticky MAC addresses do not age out by default.

The aging mechanism does not apply unless explicitly configured for non-sticky secure dynamic addresses. Sticky MAC addresses are written into the running configuration and persist until manually removed or until the switch is rebooted — unless the sticky aging type is supported and enabled, which was not the case on this device.

So, I learned that not all switches support aging of sticky MACs, and even when sticky learning is enabled, aging behavior can be device-specific.

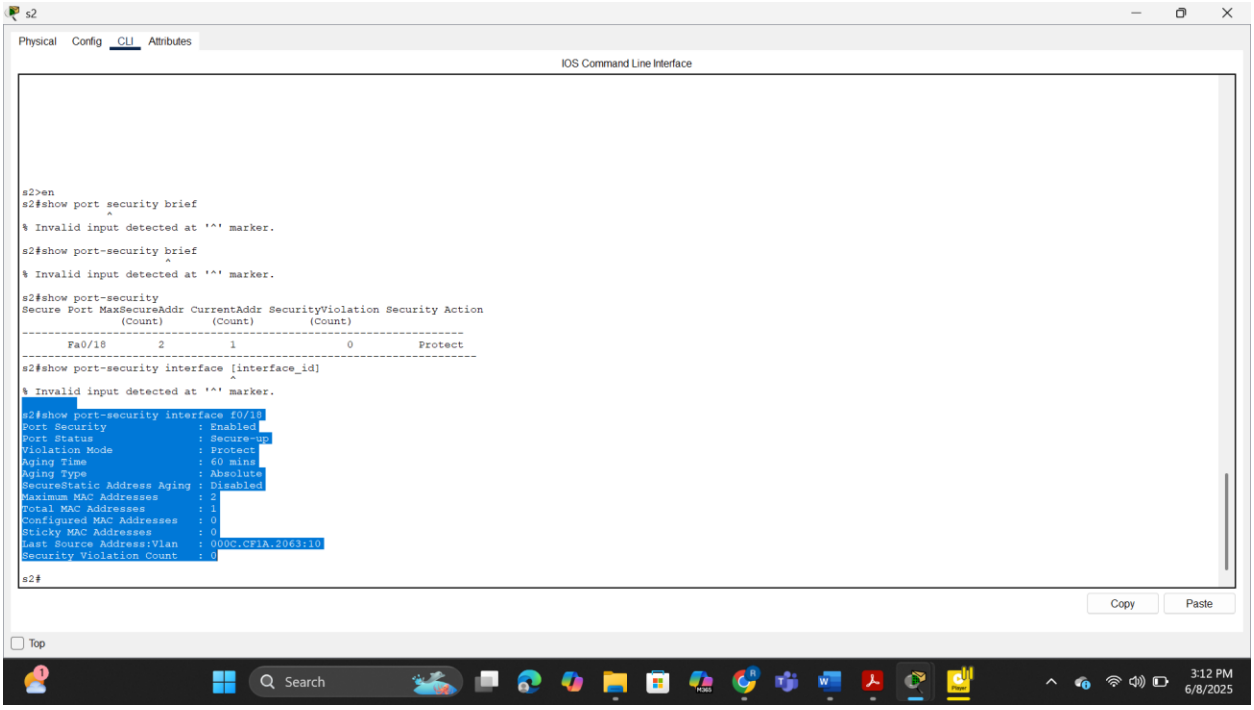


Fig 3.7 Port security interface

Why will PC-B on port F0/18 never get an IP address via DHCP if the running-config is reloaded?

This situation taught me a very valuable lesson about the interaction between port security and DHCP. When I reloaded the configuration on S2 and tried to reconnect PC-B to port F0/18, it couldn't obtain an IP address from the DHCP server.

After analyzing the settings, I realized that port security on F0/18 was configured with only two sticky MAC addresses, and the violation mode was set to protect. This meant any new device — or even the same PC with a changed NIC/MAC — would not be allowed to communicate on that port. Moreover, the protect mode silently drops unauthorized traffic without incrementing the violation counter or generating alerts, which made it less obvious at first.

The key lesson here was that port security can persist MAC bindings and quietly block traffic, especially with the protect violation mode. I must always remember to clear port security or increase the max MAC count before reconnecting or reconfiguring devices after a reload.

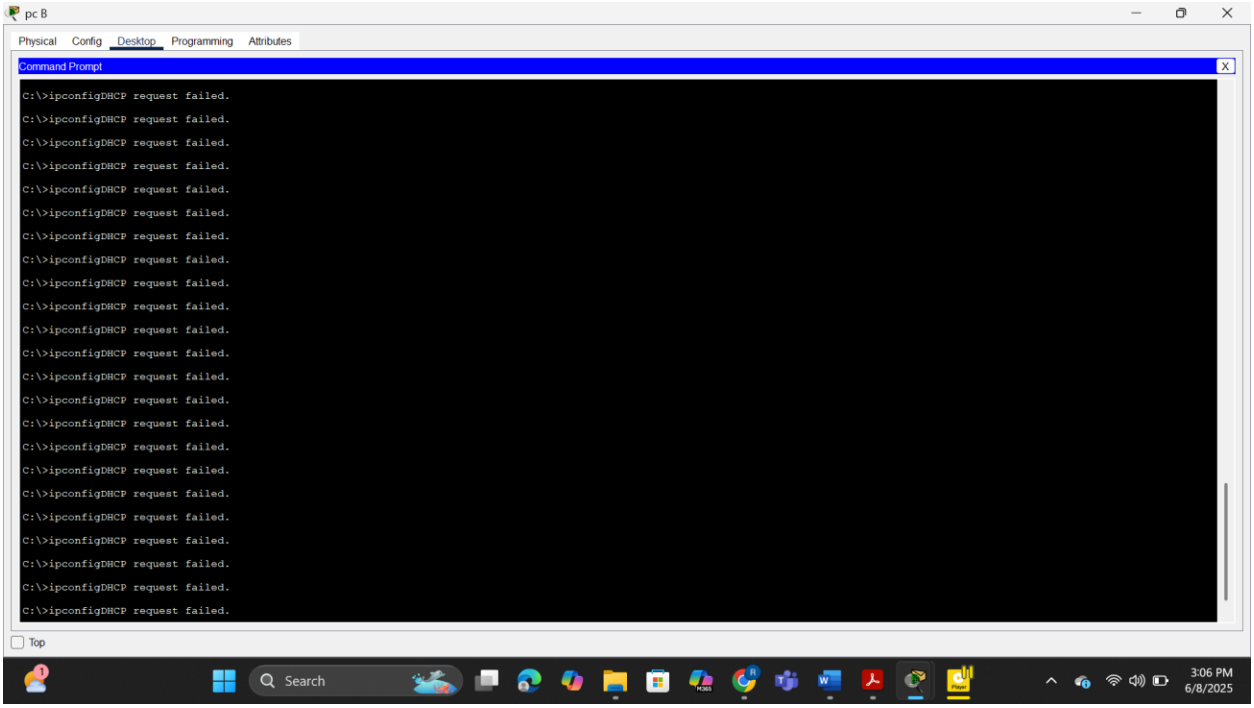


Fig 3.8 Pc b unable to be assigned ip

What is the difference between absolute aging type and inactivity aging type?

This question helped me deepen my understanding of port security aging types. In the lab, I had configured both types:

- Inactivity Aging (used on S1's F0/6): This method removes a secure MAC address only if there is no traffic from the device for the configured aging time (e.g., 60 minutes). It's useful in environments where devices may go idle for extended periods but should retain authorization while active.
- Absolute Aging (the default type): This removes the MAC address after a fixed time, regardless of activity. So even if the device is actively communicating, it will be aged out once the timer expires.

What I learned is that inactivity-based aging is more flexible and user-aware, while absolute aging is more rigid and time-based, potentially leading to service interruptions even for active devices. Choosing between them depends on the network environment and device behavior.

Conclusion

This lab provided a vital opportunity to apply and reinforce key networking skills in a simulated enterprise environment. I successfully configured the router and switches, established secure trunk links, implemented VLAN segmentation, and enforced multiple security measures on switch ports to protect the network from unauthorized access and potential attacks. Through DHCP snooping and port security, I learned how to control IP address allocation and device authentication at the access layer. Additionally,

enabling PortFast and BPDU guard taught me how to safeguard the spanning-tree topology from rogue switches. Overall, this lab sharpened my configuration proficiency, deepened my understanding of secure Layer 2 practices, and prepared me for future challenges in network infrastructure design and protection.