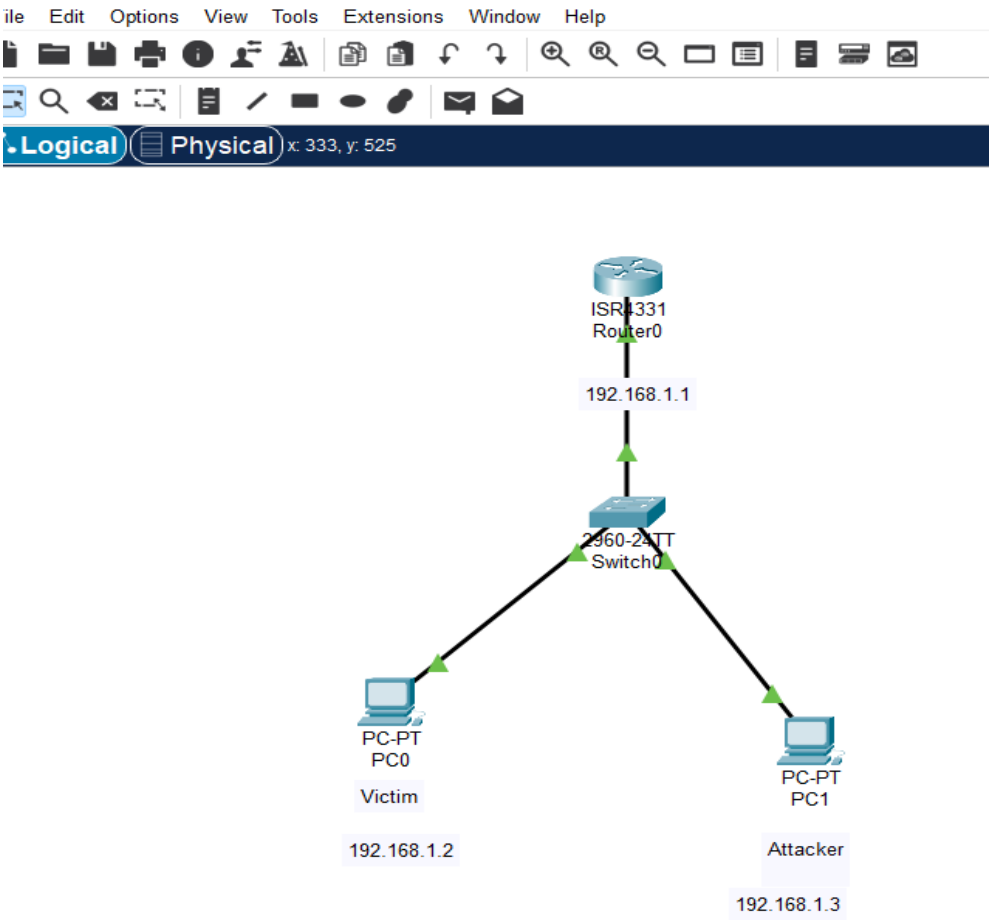
2) Using Packet Tracer, simulate an ARP spoofing attack. Analyze the behavior of devices on the network when they receive a malicious ARP response.

IP Addresses Assigned:

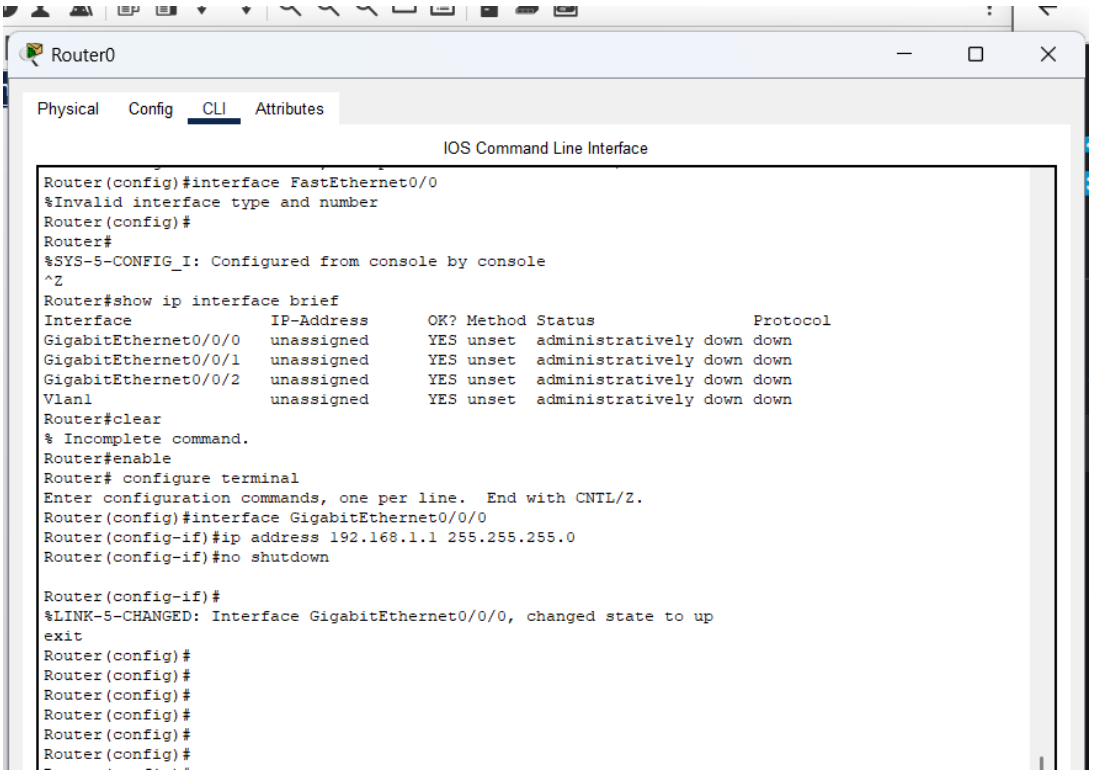
Router: 192.168.1.1

Victim PC: 192.168.1.2

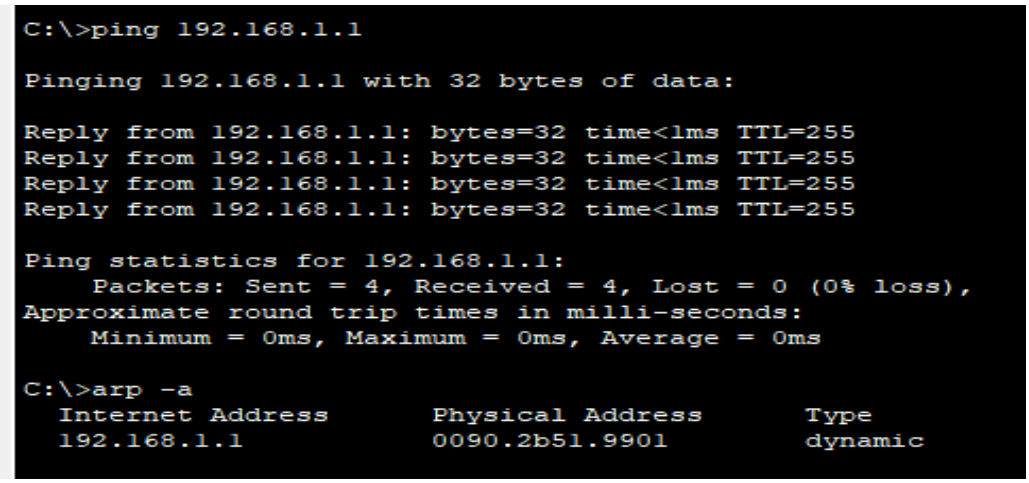
Attacker PC: 192.168.1.3



Router Configuration

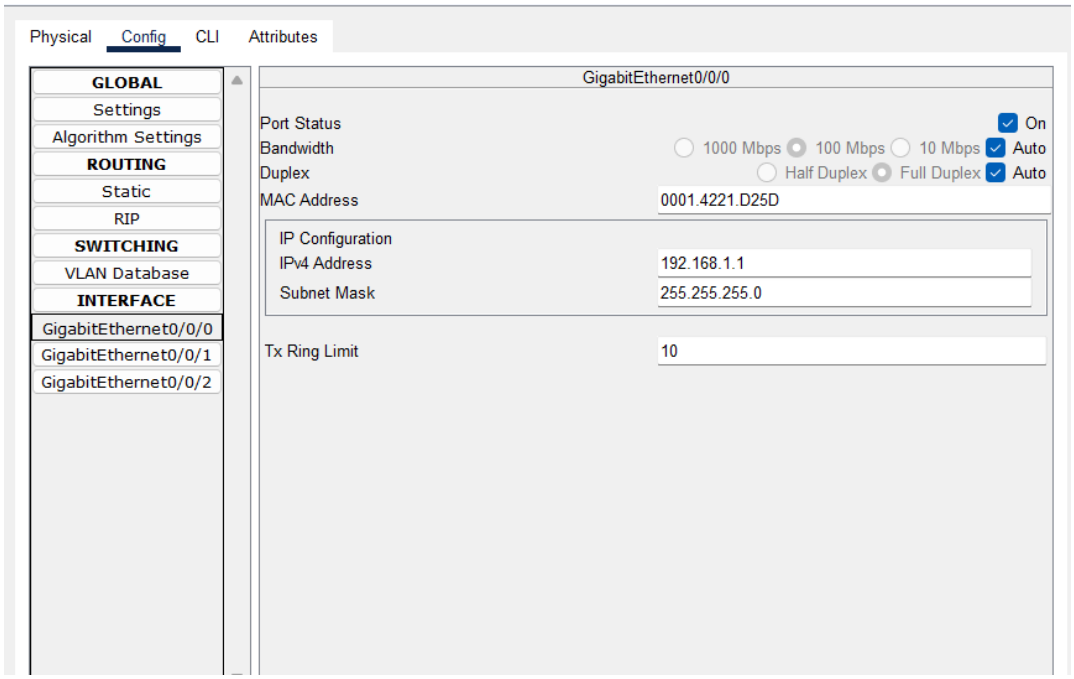


Before ARP Spoofing



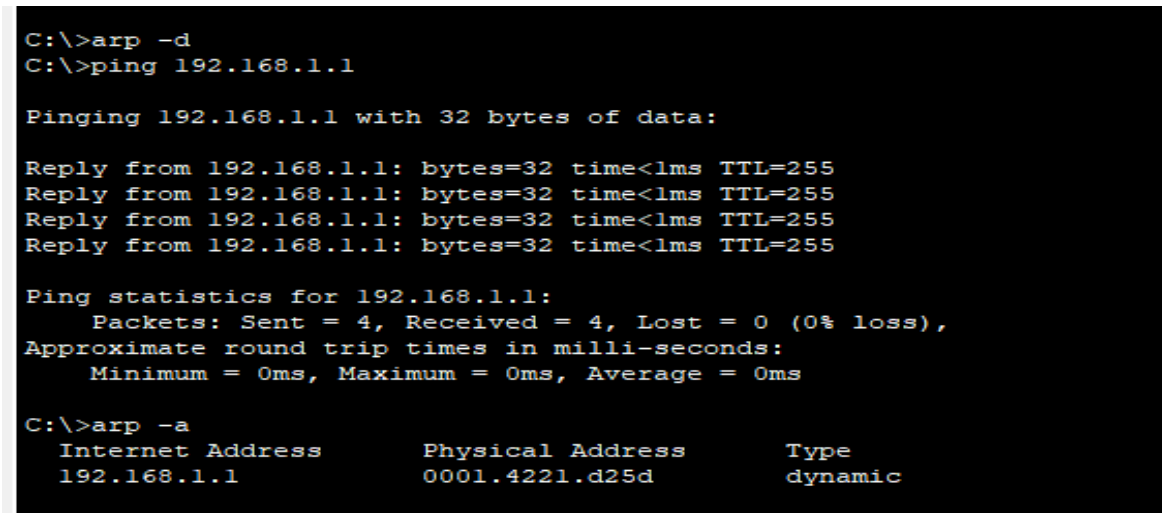
Successfully pinged the router from the victim PC to verify normal communication.

Manually changing the Mac Address



Here I Manually changed the Attacker MAC address to match the Router.

After ARP Spoofing



```
C:\>arp -d
C:\>ping 192.168.1.1

Pinging 192.168.1.1 with 32 bytes of data:

Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255

Ping statistics for 192.168.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>arp -a
  Internet Address      Physical Address      Type
  192.168.1.1           0001.4221.d25d        dynamic
```

Cleared the ARP table on the Victim PC using arp -d.

Pinging the Router again from the Victim PC resulted in an updated ARP table, now associating 192.168.1.1 with the Attacker's MAC.

All packets meant for the Router were now redirected to the Attacker PC instead.

How ARP Spoofing Worked Here

1. Before Spoofing, The Victim PC's ARP table correctly mapped 192.168.1.1 → Router's MAC.
2. I've manually changed the Attacker MAC address to match the Router's.
3. After clearing the ARP cache, the Victim PC sent an ARP request asking, "Who has 192.168.1.1?"
4. Since the Attacker now had the same MAC as the Router, it sent an ARP reply.
5. The Victim PC believed the Attacker and updated its ARP table.
6. Now, all packets from the Victim PC go to the Attacker PC instead of the Router.