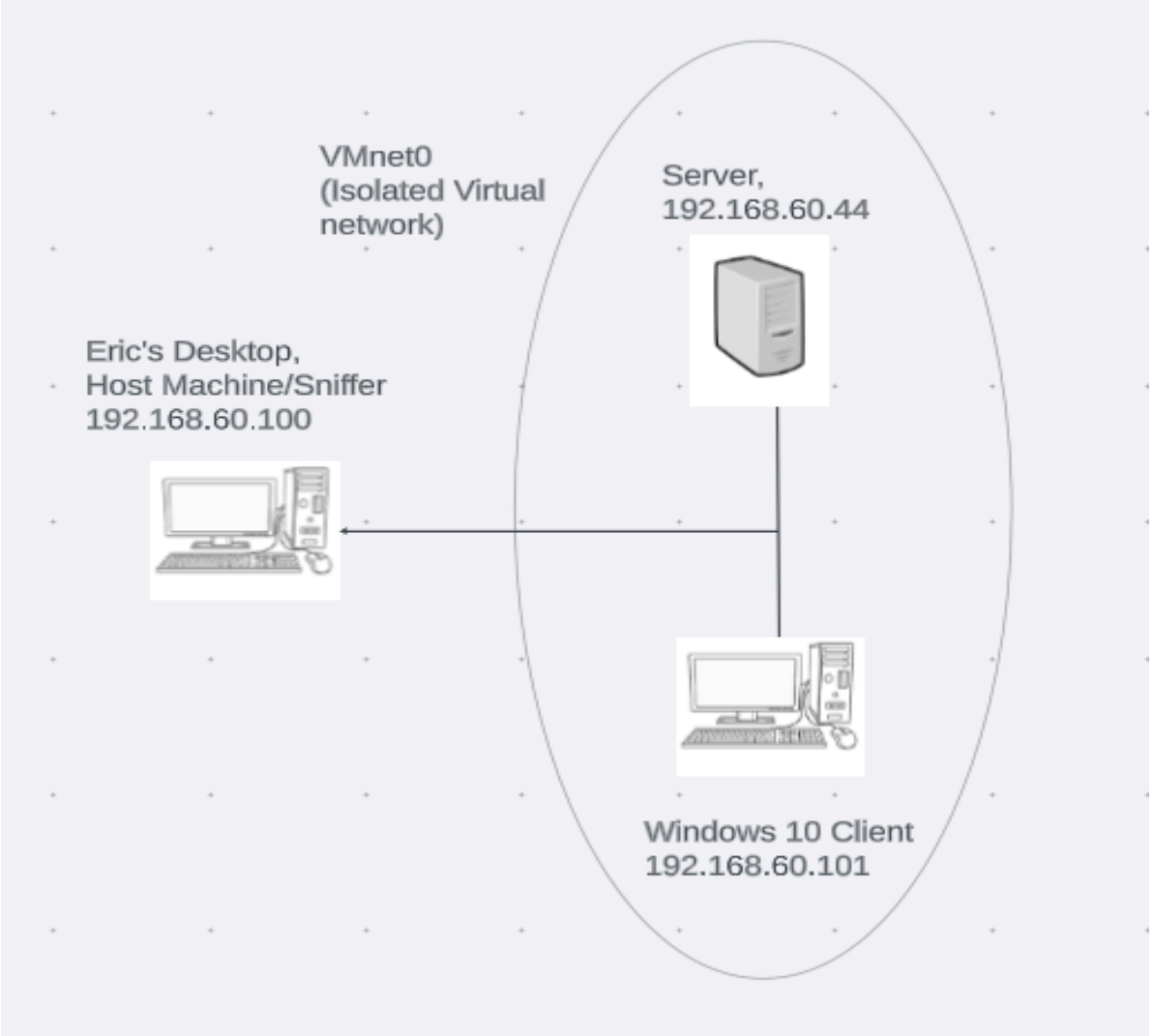
Eric Guzman
Professor Cannistra

10/11/2023
Network Security

Lab 2

Description: In this lab we captured and analyzed network traffic on a secluded network using wireshark. We isolated our Virtual Machines and performed various tasks that created packets that were recorded and able to be looked at.

Topology:



Key Syntax:

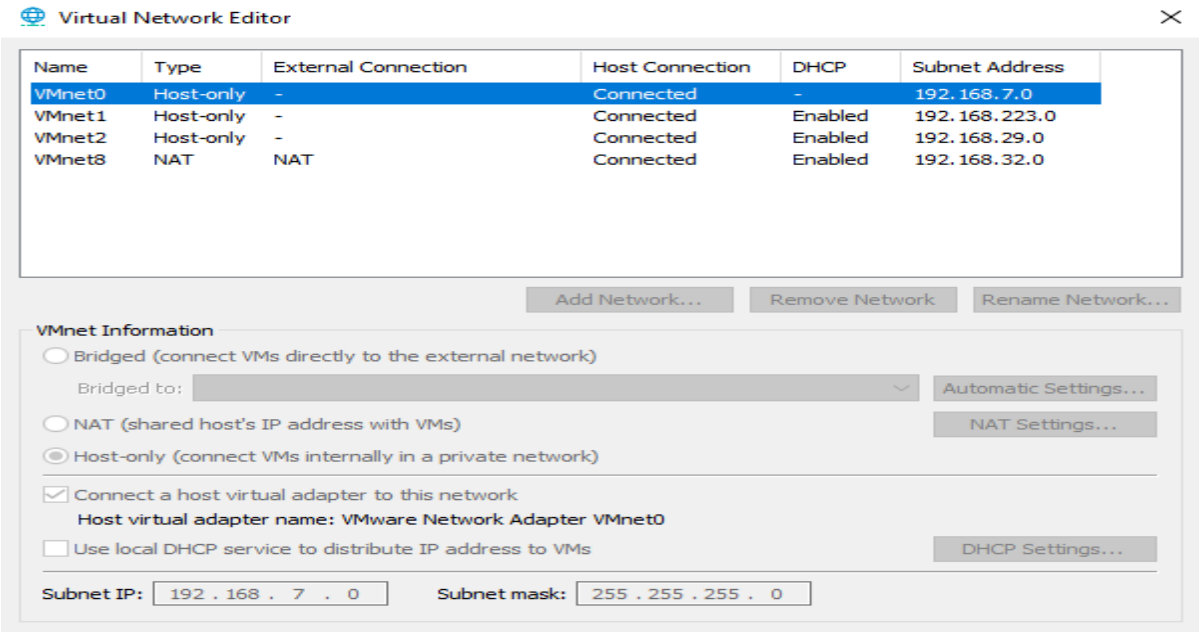
Term	Description
arp -a	Displays the ARP cache
ipconfig	Displays the configurations of all network interfaces
nbtstat	Displays statistics related to NetBIOS
nslookup	Queries DNS servers to get information on a domain name
ping	Tests network connectivity between two devices
tracert	Traces the route that packets take from the source to their destination

Verification:

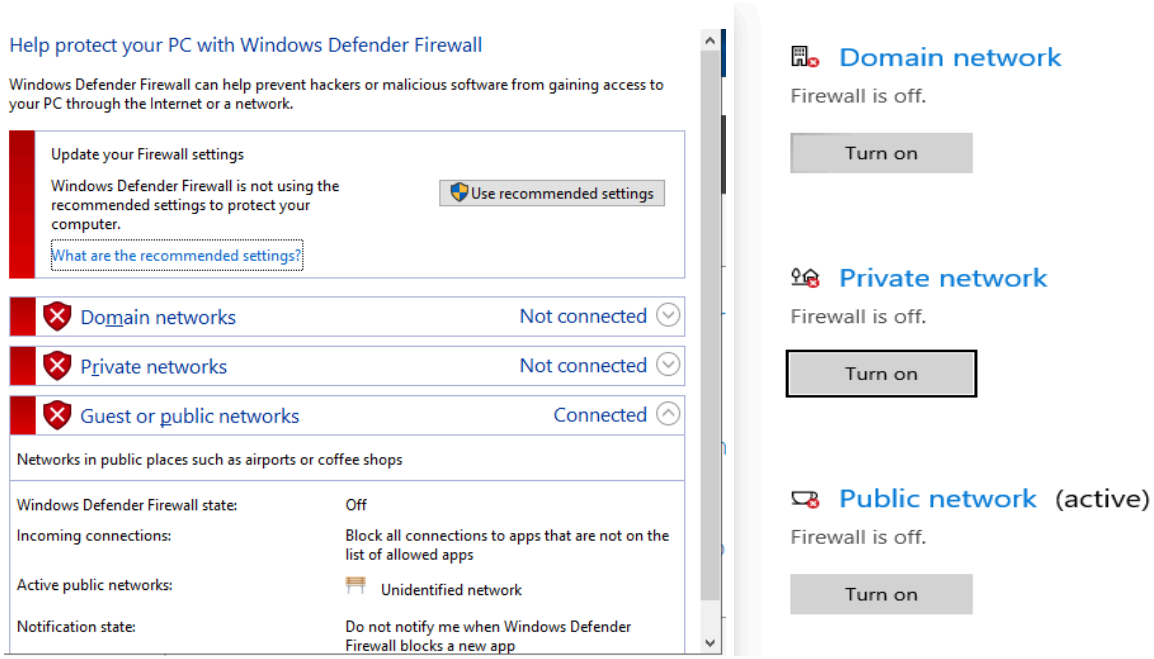
Task One:

I chose to use my host PC as my sniffer.

This screenshot shows my isolated network (VMnet0).



These screenshots shows that I disabled the firewalls on both the server and the windows client.



Task Two:

A. This screenshot shows me performing a DNS lookup.

```
C:\Users\eg0311>nslookup WIN-JGHK481UJHD
DNS request timed out.
    timeout was 2 seconds.
Server:  UnKnown
Address:  172.16.151.200

Name:    WIN-JGHK481UJHD.hacker.testlab
Address:  172.16.151.200
```

B. This screenshot shows me using the ping command.

```
C:\Users\eg0311>ping 172.16.151.200

Pinging 172.16.151.200 with 32 bytes of data:
Reply from 172.16.151.200: bytes=32 time<1ms TTL=128
Reply from 172.16.151.200: bytes=32 time<1ms TTL=128
Reply from 172.16.151.200: bytes=32 time<1ms TTL=128
Reply from 172.16.151.200: bytes=32 time<1ms TTL=128

Ping statistics for 172.16.151.200:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

C. This screenshot shows me using the tracert command to trace the route the packet takes.

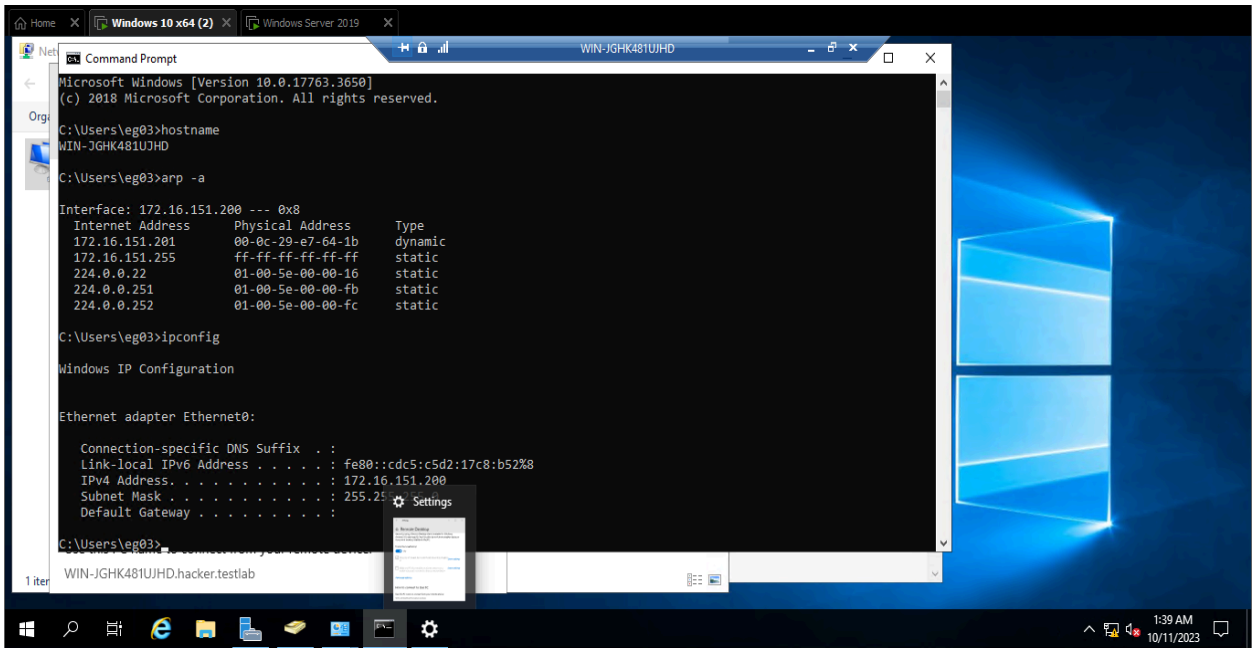
```
C:\Windows\system32>tracert 172.16.151.200

Tracing route to WIN-JGHK481UJHD [172.16.151.200]
over a maximum of 30 hops:

  1     3 ms     9 ms     1 ms  WIN-JGHK481UJHD [172.16.151.200]

Trace complete.
```

D. These screenshots show the ARP cache and the verification of ipconfig on both the server and the client.

```
C:\Users\eg03>arp -a

Interface: 172.16.151.200 --- 0x8
  Internet Address      Physical Address      Type
  172.16.151.201        00-0c-29-e7-64-1b     dynamic
  172.16.151.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static

C:\Users\eg03>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::cdc5:c5d2:17c8:b52%8
   IPv4 Address. . . . . . . . . . . : 172.16.151.200
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

```
C:\Windows\system32>arp -a

Interface: 172.16.151.201 --- 0x5
  Internet Address      Physical Address      Type
  172.16.151.200        00-0c-29-75-3f-59     dynamic
  172.16.151.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static

C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::79b4:f9cf:5e8b:f2d1%5
   IPv4 Address. . . . . . . . . . . : 172.16.151.201
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
```

E. This screenshot shows me successfully logging into the remote desktop of the server on the client VM.



F. This screenshot shows the net command being used to map a network drive from the Windows client to the Windows server's C: drive.

```
C:\Windows\system32>net use Z: \\WIN-JGHK481UJHD\C$
Enter the user name for 'WIN-JGHK481UJHD': HACKER\eg03
Enter the password for WIN-JGHK481UJHD:
The command completed successfully.
```

G. These screenshots show two other forms of traffic being produced from the client to the server.

Copy command:

```
C:\Windows\system32>copy coolfile.txt \\WIN-JGHK481UJHD\c$\Users\eg03\Documents
        1 file(s) copied.
```

Nbtstat command:

```
C:\Windows\system32>nbtstat -a WIN-JGHK481UJHD

Ethernet0:
Node IpAddress: [172.16.151.201] Scope Id: []

           NetBIOS Remote Machine Name Table

       Name               Type         Status
    ---------------------------------------------
    WIN-JGHK481UJHD<00>  UNIQUE      Registered
    HACKER          <00>  GROUP       Registered
    HACKER          <1C>  GROUP       Registered
    WIN-JGHK481UJHD<20>  UNIQUE      Registered
    HACKER          <1B>  UNIQUE      Registered

    MAC Address = 00-0C-29-75-3F-59


Bluetooth Network Connection:
Node IpAddress: [0.0.0.0] Scope Id: []

    Host not found.
```

Task Three:

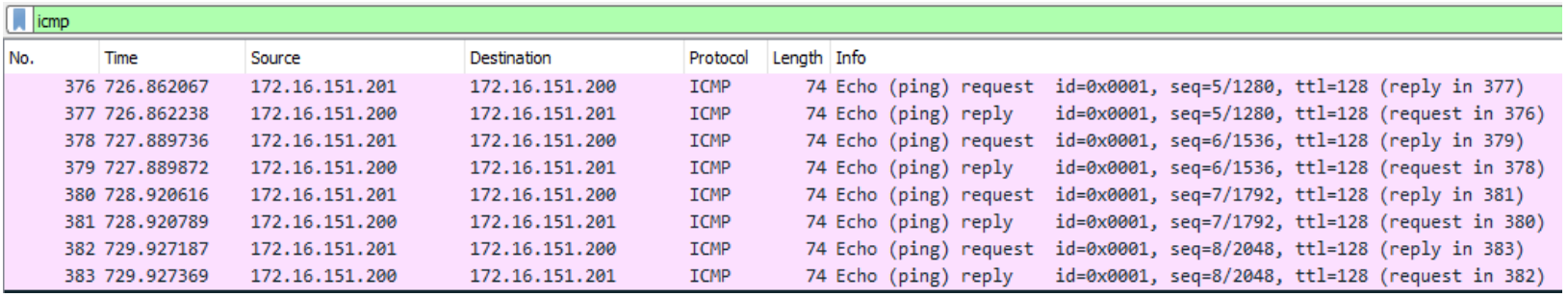
A. DNS lookup/DNS -

udp.port==53 && dns.qry.name contains "WIN-JGHK481UJHD"

No.	Time	Source	Destination	Protocol	Length	Info
338	690.534214	172.16.151.201	172.16.151.200	DNS	90	Standard query 0x0002 A WIN-JGHK481UJHD.hacker.testlab
339	690.534483	172.16.151.200	172.16.151.201	DNS	106	Standard query response 0x0002 A WIN-JGHK481UJHD.hacker.testlab A 172.16.151.200
340	690.534808	172.16.151.201	172.16.151.200	DNS	90	Standard query 0x0003 AAAA WIN-JGHK481UJHD.hacker.testlab
341	690.535064	172.16.151.200	172.16.151.201	DNS	137	Standard query response 0x0003 AAAA WIN-JGHK481UJHD.hacker.testlab SOA WIN-JGHK481UJHD.hacker.testlab

The results were that there was a query from the client to the server and the server sent back a response. This seems pretty standard, nothing really surprised me about this request and reply behavior. You can invade someone's privacy by looking at DNS packets like this. A bad actor could analyze these packets and launch a phishing attack based on the websites someone visits.

B. Ping Command/ICMP -

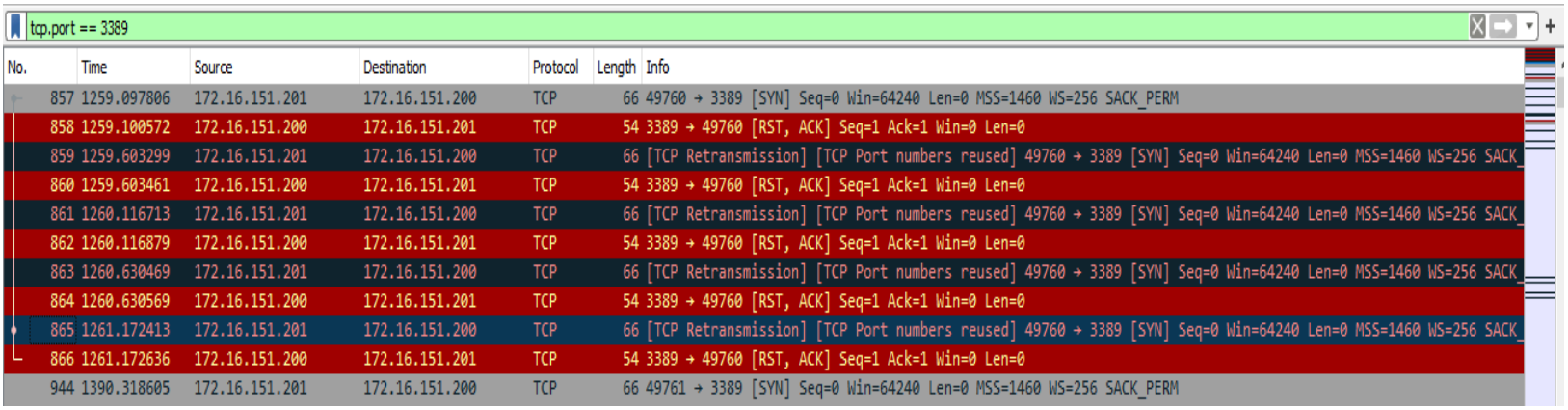


icmp

No.	Time	Source	Destination	Protocol	Length	Info
376	726.862067	172.16.151.201	172.16.151.200	ICMP	74	Echo (ping) request id=0x0001, seq=5/1280, ttl=128 (reply in 377)
377	726.862238	172.16.151.200	172.16.151.201	ICMP	74	Echo (ping) reply id=0x0001, seq=5/1280, ttl=128 (request in 376)
378	727.889736	172.16.151.201	172.16.151.200	ICMP	74	Echo (ping) request id=0x0001, seq=6/1536, ttl=128 (reply in 379)
379	727.889872	172.16.151.200	172.16.151.201	ICMP	74	Echo (ping) reply id=0x0001, seq=6/1536, ttl=128 (request in 378)
380	728.920616	172.16.151.201	172.16.151.200	ICMP	74	Echo (ping) request id=0x0001, seq=7/1792, ttl=128 (reply in 381)
381	728.920789	172.16.151.200	172.16.151.201	ICMP	74	Echo (ping) reply id=0x0001, seq=7/1792, ttl=128 (request in 380)
382	729.927187	172.16.151.201	172.16.151.200	ICMP	74	Echo (ping) request id=0x0001, seq=8/2048, ttl=128 (reply in 383)
383	729.927369	172.16.151.200	172.16.151.201	ICMP	74	Echo (ping) reply id=0x0001, seq=8/2048, ttl=128 (request in 382)

The results were 8 packets being made, 4 requests being sent and 4 replies received. This also seems very standard, nothing out of the ordinary for a ping command. I can use wireshark to measure latency and see how many packets are lost with the ping command. A bad actor could send many ping/ICMP requests to try and do a denial of service attack on a server.

C. Remote Desktop Connection/TCP/RDP



tcp.port == 3389

No.	Time	Source	Destination	Protocol	Length	Info
857	1259.097806	172.16.151.201	172.16.151.200	TCP	66	49760 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
858	1259.100572	172.16.151.200	172.16.151.201	TCP	54	3389 → 49760 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
859	1259.603299	172.16.151.201	172.16.151.200	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 49760 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_
860	1259.603461	172.16.151.200	172.16.151.201	TCP	54	3389 → 49760 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
861	1260.116713	172.16.151.201	172.16.151.200	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 49760 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_
862	1260.116879	172.16.151.200	172.16.151.201	TCP	54	3389 → 49760 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
863	1260.630469	172.16.151.201	172.16.151.200	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 49760 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_
864	1260.630569	172.16.151.200	172.16.151.201	TCP	54	3389 → 49760 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
865	1261.172413	172.16.151.201	172.16.151.200	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 49760 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_
866	1261.172636	172.16.151.200	172.16.151.201	TCP	54	3389 → 49760 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
944	1390.318605	172.16.151.201	172.16.151.200	TCP	66	49761 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

The screenshot I took shows the initiation of a new connection when I started the remote desktop. This actually surprised me as it showed the specific sequence of packets that are involved with establishing the remote desktop connection. The value I can see from looking at these packets in wireshark is that you can monitor the TCP and RDP traffic and look for unauthorized access or issues. A bad actor could potentially gain unauthorized access to a remote system and take hold of the data on that device.

D. Net Command/SMB

smb

No.	Time	Source	Destination	Protocol	Length	Info
52	202.547085	172.16.151.200	172.16.151.255	BROWSER	245	Host Announcement WIN-JGHK481UJHD, Workstation, Server, Domain Controller, Time Source, NT Workstation, DFS server
56	207.577663	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
335	686.080625	172.16.151.201	172.16.151.255	BROWSER	248	Host Announcement WIN10, Workstation, Server, NT Workstation
540	920.057375	172.16.151.200	172.16.151.255	BROWSER	245	Host Announcement WIN-JGHK481UJHD, Workstation, Server, Domain Controller, Time Source, NT Workstation, DFS server
766	1107.694620	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
952	1408.539545	172.16.151.201	172.16.151.255	BROWSER	248	Host Announcement WIN10, Workstation, Server, NT Workstation
2282	1642.554953	172.16.151.200	172.16.151.255	BROWSER	245	Host Announcement WIN-JGHK481UJHD, Workstation, Server, Domain Controller, Time Source, NT Workstation, DFS server
3958	1903.543220	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
4445	2007.708796	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
4479	2013.260952	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
4554	2035.105923	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
4852	2129.667253	172.16.151.201	172.16.151.255	BROWSER	248	Host Announcement WIN10, Workstation, Server, NT Workstation
5796	2363.347930	172.16.151.200	172.16.151.255	BROWSER	245	Host Announcement WIN-JGHK481UJHD, Workstation, Server, Domain Controller, Time Source, NT Workstation, DFS server
5940	2852.347353	172.16.151.201	172.16.151.255	BROWSER	248	Host Announcement WIN10, Workstation, Server, NT Workstation
5978	2907.727437	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
6109	3084.252607	172.16.151.200	172.16.151.255	BROWSER	245	Host Announcement WIN-JGHK481UJHD, Workstation, Server, Domain Controller, Time Source, NT Workstation, DFS server
6353	3565.860573	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request
6464	3571.345584	172.16.151.201	172.16.151.255	BROWSER	248	Host Announcement WIN10, Workstation, Server, NT Workstation
6614	3802.528958	172.16.151.200	172.16.151.255	BROWSER	245	Host Announcement WIN-JGHK481UJHD, Workstation, Server, Domain Controller, Time Source, NT Workstation, DFS server
6618	3807.745975	172.16.151.201	172.16.151.200	SMB	127	Negotiate Protocol Request

The results were that there are requests sent and responses sent back. I found this command/protocol surprising since I had not seen it before in wireshark. Analysis of SMB packets can help look for imperfections and allow someone to optimize file transfer speeds, resource access times, and overall network efficiency. A bad actor could potentially harvest user credentials with these packets, making it important to have these packets be encrypted.

Conclusion:

This lab was very interesting from start to finish. I liked having to set up an isolated network to capture traffic and packets. It was very cool to see the amount of packets that are transferred between just two devices. My computer did crash once during this, completely due to my fault though since I had many apps open on top of the two VMs. Everything worked perfectly after that thankfully.

References:

1. https://www.wireshark.org/docs/wsug_html_chunked/ChWorkDisplayFilterSection.html
2. https://www.ibm.com/docs/en/aix/7.2?topic=management-smb-protocol
3. https://kb.vmware.com/s/article/2043160