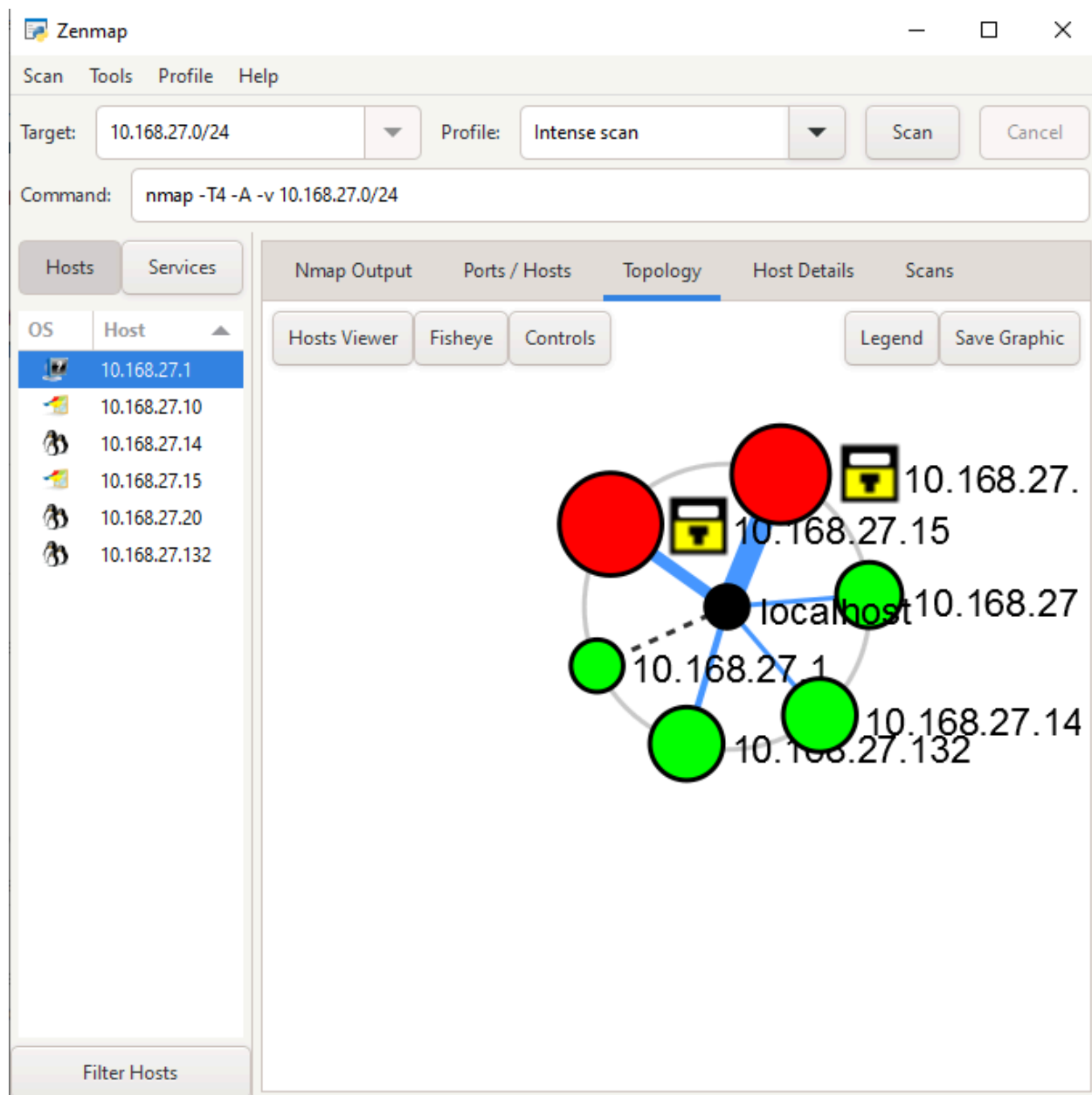
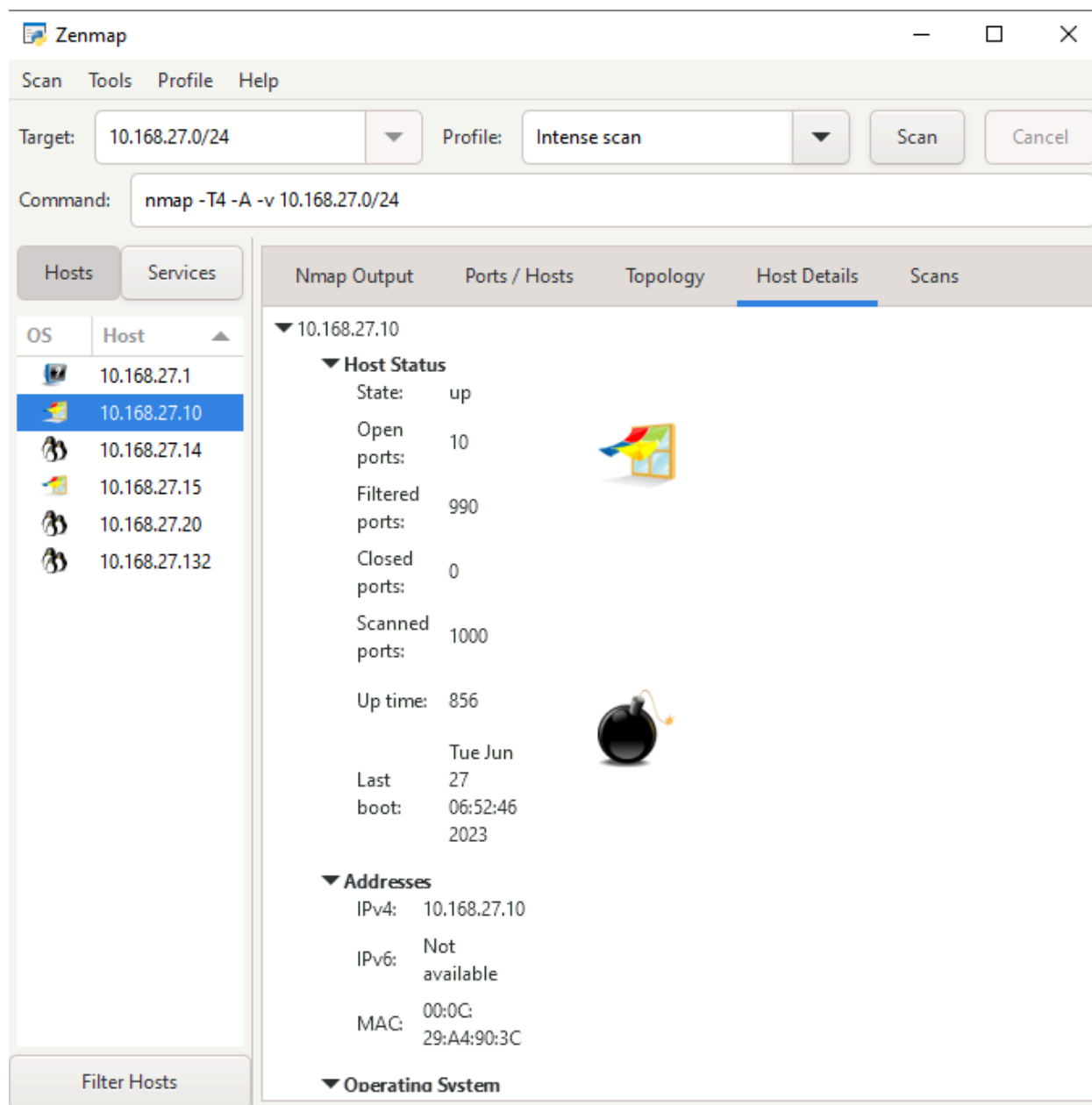


## A. Network Topology



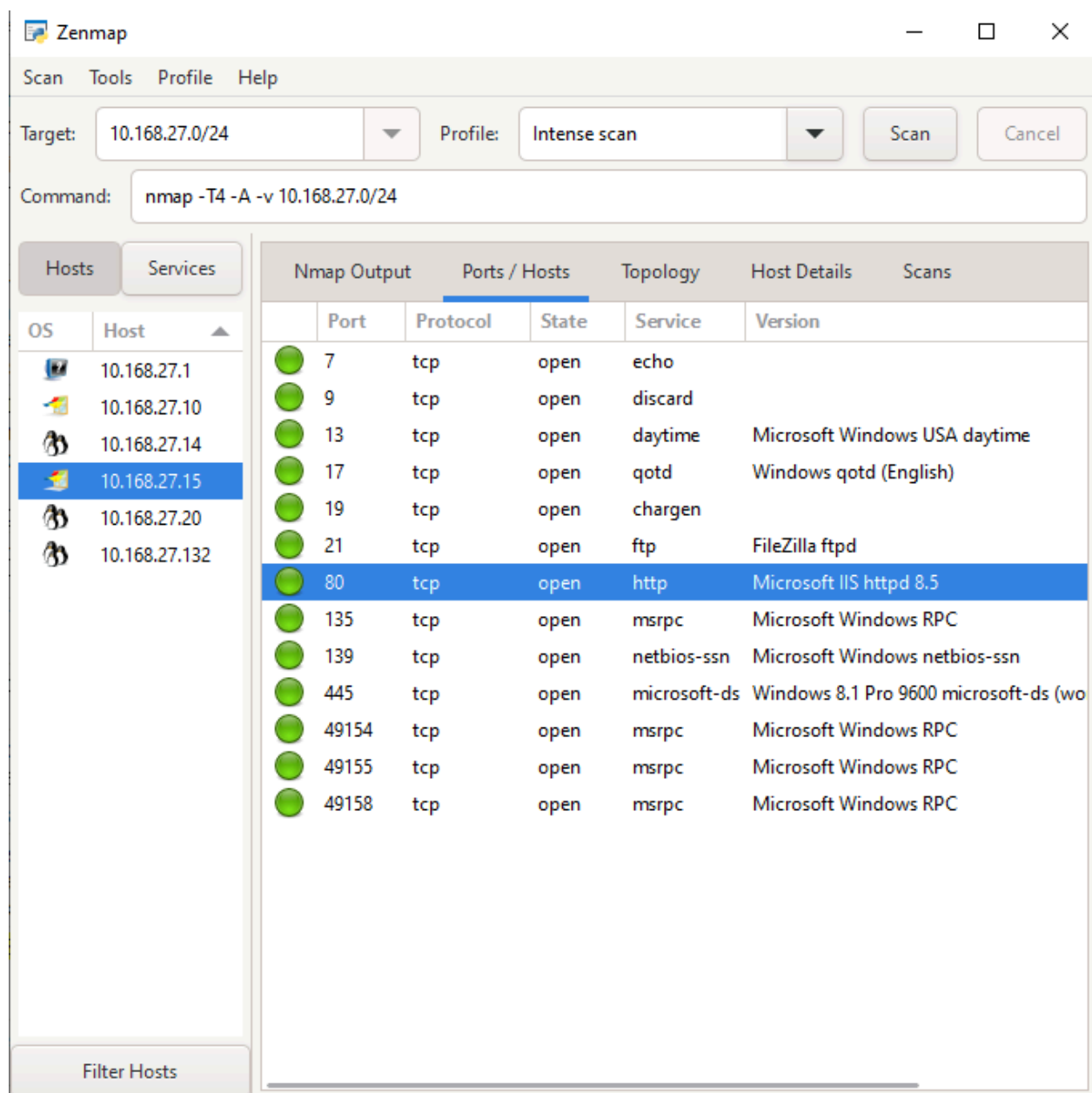
The network uses the STAR topology. There are 6 hosts in total (as seen on the left side): 10.168.27.1 which is running on an unknown operating system, 10.168.127.10 which is running on a Windows OS, 10.168.27.14 which is running on Linux OS, 10.168.27.15 which is also a Windows OS, and the final two 10.168.27.20 and 10.168.27.132 are both Linux OS.

We can dive into more details about the host as needed through the Host Details section and individually click on each host on the left side with Zenmap to further find details such as host uptime, how many ports are open, MAC addresses, and other helpful information.

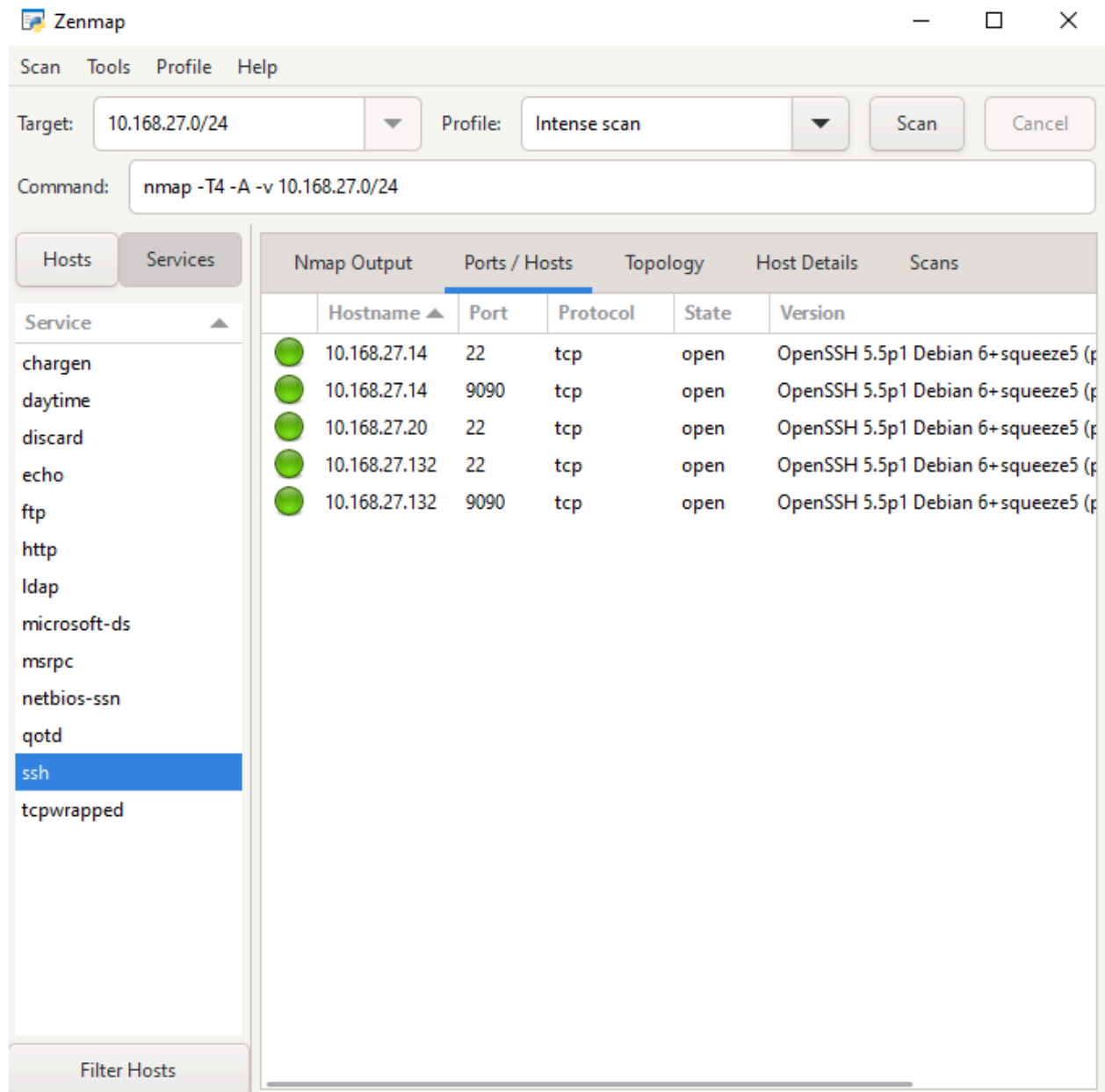


10.168.27.10 has 10 open ports (pictured above), and going through the rest of the hosts we can see that 10.168.27.14 has 2 open ports, host 10.168.27.15 has 13 open ports, 10.168.27.20 has 1 open port, 10.168.27.132 has 2 open ports, and our unknown host 10.168.27.1 has 0 open ports.

## B. Vulnerabilities

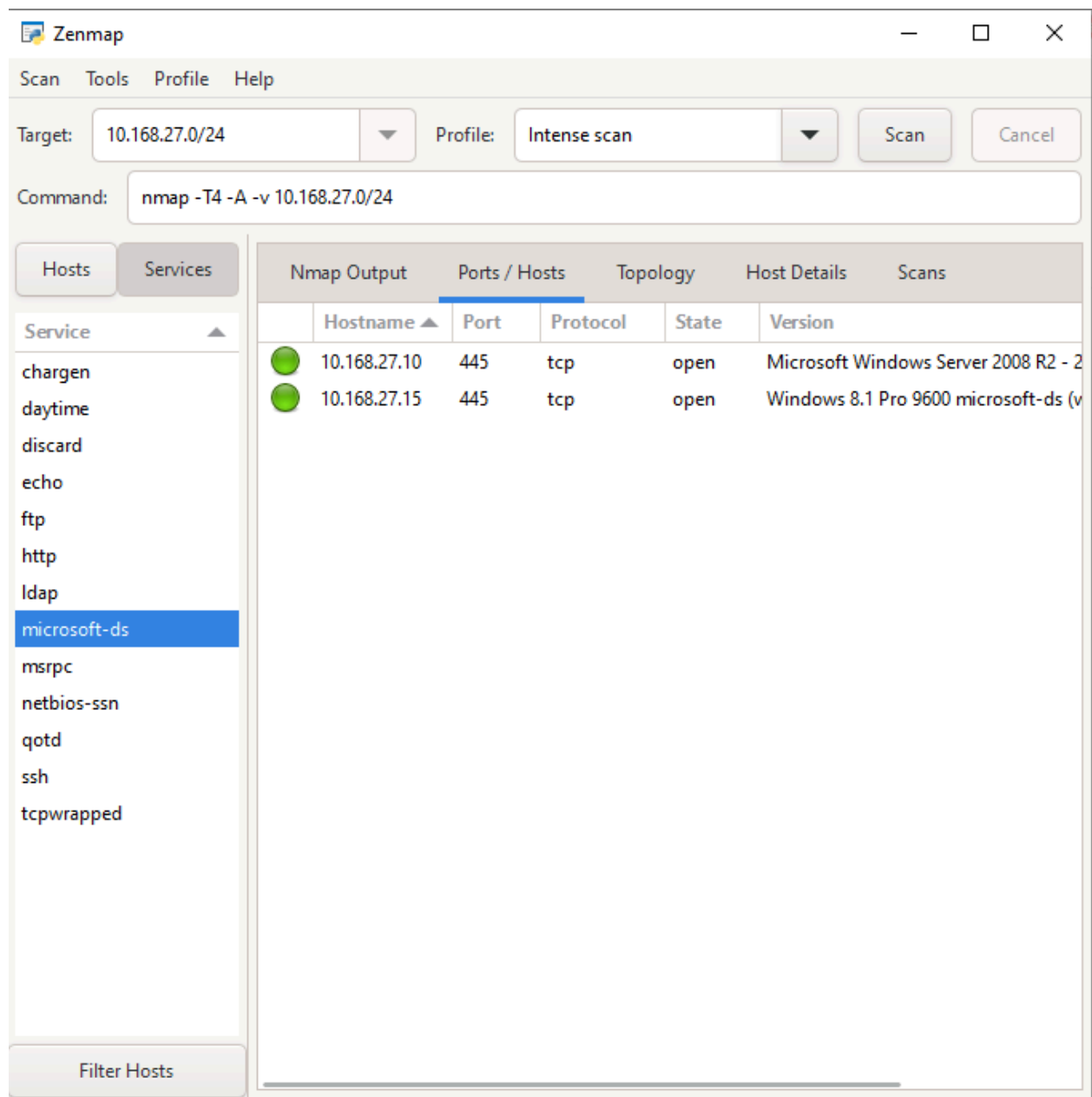


The computer with the IP 10.168.27.15 (pictured above) is using port 80 HTTP which is not a secure protocol as information is not encrypted and allows web traffic to be viewed in the clear. If a hacker is able to perform an on-path attack, this would be a risk to any user using this computer especially if they type in usernames, passwords, credit card information, or any other sensitive data that should be encrypted over the internet.



[CVE-2016-6515](#) The `auth_password` function in `auth-passwd.c` in `sshd` in **OpenSSH** before 7.3 does not limit password lengths for password authentication, which allows remote attackers to cause a denial of service (crypt CPU consumption) via a long string.

The version of OpenSSH 5.5p.1 has several vulnerabilities such as causing a DOS through invalid input validation with CVE-2016-6515 which does not limit password lengths (CVE). There are many vulnerabilities with OpenSSH before version X.X, it is important to make sure that the services that the computers are using are updated using patch management throughout the network environment.



[CVE-2019-0543](#) An elevation of privilege vulnerability exists when Windows improperly handles authentication requests, aka "Microsoft Windows Elevation of Privilege Vulnerability." This affects Windows 7, Windows Server 2012 R2, Windows RT 8.1, Windows Server 2008, Windows Server 2019, Windows Server 2012, Windows 8.1, Windows Server 2016, Windows Server 2008 R2, Windows 10, Windows 10 Servers.

The Microsoft website mentions that these versions of OS running on both Windows computers are at the end-of-life. 8.1 reached end-of-life a few months ago in January. Microsoft Windows Server 2008 R2 reached end-of-life in 2020. There is a risk in using end-of-life products that are no longer getting vendor support. These systems are vulnerable to such things as privilege elevation attacks, CVE-2019-0543 (CVE). As part of its patch management, this company should ensure that they are using products that are still receiving vendor support and upgrade the operating systems on their Windows computers. The NVD mentions that the best way to

prevent this specific CVE is to apply patches from the vendor which on the Microsoft Windows Server 2008 R2 is not possible since the patch was not released until 2022 (2 years after support ended for this specific Window's OS) (NVD).

## C. Wireshark Anomalies (using Pcap1)

### FTP -

Wireshark packet capture showing an FTP session. The packet list shows a series of requests and responses between 49.12.121.47 and 10.168.27.10. The packet details pane shows the structure of an FTP packet, including Ethernet II, IP, and FTP layers. The packet bytes pane shows the raw data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
2138...	690.651507201	49.12.121.47	10.168.27.10	FTP	93	Response: 220 FZ router and firew
2138...	690.651514416	49.12.121.47	10.168.27.10	FTP	60	Response:
2138...	690.654468522	10.168.27.10	49.12.121.47	FTP	70	Request: USER FileZilla
2138...	690.783827486	49.12.121.47	10.168.27.10	FTP	76	Response: 331 Give any password.
2138...	690.783829054	49.12.121.47	10.168.27.10	FTP	60	Response:
2138...	690.786997247	10.168.27.10	49.12.121.47	FTP	67	Request: PASS 3.55.1
2138...	690.916622296	49.12.121.47	10.168.27.10	FTP	68	Response: 230 logged on.
2138...	690.916623530	49.12.121.47	10.168.27.10	FTP	60	Response:
2138...	690.922817898	10.168.27.10	49.12.121.47	FTP	84	Request: IP 10.168.27.10 ba-bgi-c
2138...	691.053128956	49.12.121.47	10.168.27.10	FTP	107	Response: 510 Mismatch. Your IP i
2138...	691.053129566	49.12.121.47	10.168.27.10	FTP	60	Response:
1743	238.443273045	10.16.80.243	10.168.27.1	TCP	74	34306 → 21 [SYN] Seq=0 Win=64240

Frame 213829: 60 bytes on wire (480 bits), 60 bytes captured on interface 0 (eth0) from 49.12.121.47 to 10.168.27.10 on interface 0 (eth0)

Ethernet II, Src: VMware\_b0:0c:1f (00:0c:29:b0:0c:1f), Dst: 10.168.27.10 (08:00:27:10:00:00)

Internet Protocol Version 4, Src: 49.12.121.47, Dst: 10.168.27.10

Transmission Control Protocol, Src Port: 21, Dst Port: 21

File Transfer Protocol (FTP)

[Current working directory: ]

0000 00 15 5d 01 80 06 00 0c 29 b0 0c 1f 08 00 45 2a  
 0010 00 2a d8 35 40 00 2d 06 a5 81 31 0c 79 2f 0a a8  
 0020 1b 0a 00 15 c0 56 7d 6a a9 cd c2 d9 c2 82 50 18  
 0030 01 f6 63 dd 00 00 0d 0a 00 00 00 00

In this packet capture from Wireshark, there is a host that is not part of the network topology from our viewing of Nmap/Zenmap. 49.12.121.47 seems to potentially be spoofing itself as 10.168.27.10 to create an account and password and logging into the FTP server.

### TLSv1 -

The image shows a Wireshark packet capture window titled "Pcap1.pcapng". The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar. A display filter is set to "Apply a display filter ... <Ctrl-/>".

The packet list pane shows a series of 12 packets, all of which are TLSv1 Client Hello messages. The columns are No., Time, Source, Destination, Protocol, Length, and Info. The source IP is 10.16.80.243 and the destination IP is 10.168.27.10. The length of each packet is 583 bytes.

The packet details pane for the selected packet (No. 17082) shows the following layers:

- Frame 203795: 583 bytes on wire (4664 bits), 583 bytes captured (4664 bits) on interface 0
- Ethernet II, Src: Microsof\_01:80:10 (00:15:5d:01:80:10), Dst: 08:00:27:10:16:80
- Internet Protocol Version 4, Src: 10.16.80.243, Dst: 10.168.27.10
- Transmission Control Protocol, Src Port: 51044, Dst Port: 443
- Transport Layer Security

The packet bytes pane shows the raw data of the selected packet, starting with the TLSv1 Client Hello structure: 0000 00 15 5d 01 80 06 00 15 5d 01 80 10 08 00 45 00 0010 02 39 39 2f 40 00 40 06 7e db 0a 10 50 f3 0a a8 0020 1b 0a c7 64 02 7c 03 3b e2 66 1c fe 76 fa 80 18 0030 01 f6 38 a0 00 00 01 01 08 0a d6 fb ff 9f 00 0a 0040 62 22 16 03 01 02 00 01 00 01 fc 03 03 ae fd f4 0050 6e fb 67 6f 6b 56 ee 15 6a c3 70 e2 a9 6b d6 9f 0060 a0 d6 d0 2f ac 7a 7d 31 92 71 0a 26 dc 20 bf 6e 0070 4f 5c 88 3f 20 82 69 78 cb bd 89 a3 7c 69 ce 9c 0080 d9 04 d9 cc a5 07 b2 91 18 bc 8b 6e c7 d0 00 9c 0090 13 02 13 03 13 01 00 33 00 39 00 35 00 2f c0 2c 00a0 c0 30 00 a3 00 9f cc a9 cc a8 cc aa c0 af c0 ad 00b0 c0 a3 c0 9f c0 5d c0 61 c0 57 c0 53 c0 2b c0 2f 00c0 00 a2 00 9e c0 ae c0 ac c0 a2 c0 9e c0 5c c0 60 00d0 c0 56 c0 52 c0 24 c0 28 00 6b 00 6a c0 73 c0 77 00e0 00 c4 00 c3 c0 73 c0 77 00 67 00 40 c0 77 c0 76

The status bar at the bottom indicates: Packets: 219377 · Displayed: 219377 (100.0%) · Marked: 1 (0.0%) · Profile: Default

Using TLSv1 is an out-of-date TLS encryption version as there is TLS1.1, 1.2, and 1.3 versions. Using this out-of-date protocol for encryption over the internet can cause the computer to be at risk.

**HTTP -**

Pcap1.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
2183...	1167.1729138...	10.168.27.10	10.16.80.243	HTTP	622	HTTP/1.1 302 Found
2183...	1167.2390849...	10.16.80.243	10.168.27.10	HTTP	490	GET /dvwa/login.php HTTP/1.1
2183...	1167.2740372...	10.168.27.10	10.16.80.243	HTTP	1836	HTTP/1.1 200 OK (text/html)
2183...	1167.3010476...	10.16.80.243	10.168.27.10	HTTP	438	GET /dvwa/dvwa/css/login.css HTTP/1.1
2183...	1167.3019607...	10.168.27.10	10.16.80.243	HTTP	1216	HTTP/1.1 200 OK (text/css)
2183...	1167.3021433...	10.16.80.243	10.168.27.10	HTTP	442	GET /dvwa/dvwa/images/login_logo. HTTP/1.1
2183...	1167.3032550...	10.168.27.10	10.16.80.243	HTTP	3673	Continuation
2192...	1698.9959670...	10.16.80.243	10.168.27.10	HTTP	419	GET / HTTP/1.1
2192...	1699.0026000...	10.168.27.10	10.16.80.243	HTTP	1045	HTTP/1.1 200 OK (text/html)
2192...	1706.5884007...	10.16.80.243	10.168.27.10	HTTP	476	GET /dvwa/ HTTP/1.1
2192...	1706.5992785...	10.168.27.10	10.16.80.243	HTTP	417	HTTP/1.1 302 Found
2192...	1708.2053269...	10.16.80.243	10.168.27.10	HTTP	485	GET /dvwa/login.php HTTP/1.1

> Frame 219264: 485 bytes on wire (3880 bits), 485 bytes captured (3880 bits) on interface 0  
 > Ethernet II, Src: Microsof\_01:80:10 (00:15:5d:01:80:10), Dst: 10.16.80.243  
 > Internet Protocol Version 4, Src: 10.16.80.243, Dst: 10.168.27.10  
 > Transmission Control Protocol, Src Port: 43906, Dst Port: 80  
 > Hypertext Transfer Protocol

Pcap1.pcapng | Packets: 219377 · Displayed: 219377 (100.0%) · Marked: 1 (0.0%) | Profile: Default

It seems that HTTP is being used to log in. HTTP is sent in-the-clear, so if a on-path attack happens all of the user's credentials (username and password) would be seen by the attacker which would allow them to perform several attacks.

**D. Implications of not addressing the anomalies****FTP -**

FTP is an unsecure protocol that sends information in-the-clear. This allows an attacker that is able to capture the network traffic access to view any file being transferred, or username and passwords being sent. Using some form of secure FTP protocol such as SFTP (SSH FTP) or FTPS (FTP Secure) would encrypt the traffic being sent and disallow an attack from viewing anything sent over the FTP's traffic.

**TLSv1 -**

[CVE-2014-3566](#) The SSL protocol 3.0, as used in OpenSSL through 1.0.1i and other products, uses nondeterministic CBC padding, which makes it easier for man-in-the-middle attackers to obtain cleartext data via a padding-oracle attack, aka the "POODLE" issue.



TLSv1 is able to fall back to SSL 3.0 which has vulnerabilities due to a POODLE downgrade attack (Bhattacharya). The attacker can use an on-path attack and obtain cleartext by downgrading the TLS encryption to use the SSL 3.0 protocol, CVE-2014-3566 (CVE).

#### **HTTP -**

HTTP is an insecure protocol that sends information in-the-clear. This means that if you are on a website that requires you to use your credit card or log in with a username and password, this information is being sent as plaintext and can be read by an attacker that is able to capture your network traffic that is sent over HTTP.

### **E. Recommend solutions for eliminating or minimizing vulnerabilities or anomalies from Wireshark and Nmap.**

#### **FTP -**

Using a secure method of transferring files by using SSH with FTP (SFTP) via port 22 instead of port 21. For SFTP the user would need proper authentication in order to connect and use the FTP all while encrypting the information exchanged between the two systems. The user connecting would need the correct info to connect such as a username, password, and SSH key which would ensure that unauthorized users are unable to connect to the FTP server. (Cyber Defense Magazine)

#### **TLSv1 -**

TLSv1 is vulnerable to a POODLE downgrade attack that makes TLSv1 fall back to SSL 3.0 which opens up vulnerabilities on the outdated SSL encryption. There are newer versions 1.2 and 1.3 of TLS that are not vulnerable to this POODLE attack so simply upgrading the TLS version would remove this specific vulnerability and greatly improve the strength of the encryption used over the web. (Bhattacharya)

#### **HTTP -**

HTTP sends traffic in-the-clear and makes a user vulnerable to several attacks such as on-path attacks, or allowing the attack to redirect your traffic to malicious websites. HTTPS securely sends HTTP traffic over the web with encryption which prevents these types of attacks. HTTPS text would be in Ciphertext (encrypted) instead of using Plaintext (in-the-clear) that HTTP uses which would not allow an attacker to view your web requests. (Cloudflare)

## Works Cited

CVE, [cve.mitre.org/](https://cve.mitre.org/). Accessed 27 June 2023.

NVD, [nvd.nist.gov/](https://nvd.nist.gov/). Accessed 27 June 2023.

Bhattacharya, Anish. "Why Older TLS PROTOCOLS ARE UNSAFE FOR YOUR ORGANIZATION?" *Encryption Consulting*, 17 Nov. 2022, [www.encryptionconsulting.com/why-should-organizations-avoid-older-tls-protocols#:~:text=TLS%201.0%20and%201.1%20are,a%20server%20for%20MITM%20attacks](https://www.encryptionconsulting.com/why-should-organizations-avoid-older-tls-protocols#:~:text=TLS%201.0%20and%201.1%20are,a%20server%20for%20MITM%20attacks).

*Why Is HTTP Not Secure? | HTTP vs. HTTPS | Cloudflare*, [www.cloudflare.com/learning/ssl/why-is-http-not-secure/](https://www.cloudflare.com/learning/ssl/why-is-http-not-secure/). Accessed 27 June 2023.

"Forget FTP: 4 Modern Protocols You Should Use Instead." *Cyber Defense Magazine*, 26 July 2017, [www.cyberdefensemagazine.com/forget-ftp-4-modern-protocols-you-should-use-instead/](https://www.cyberdefensemagazine.com/forget-ftp-4-modern-protocols-you-should-use-instead/).