# Cybersecurity Executive Summary

Violet Figueroa

Table of Contents:

## Introduction

My name is Violet Figueroa. I am a Cybersecurity Analyst. I have been tasked with retrieving and analyzing all information on your NatNetwork digital infrastructure.

## Executive Summary

Your network relies on a flat network infrastructure. All devices are connected and accessible to each other through several access ports. Expanding the infrastructure will degrade response times, making the network unusable (Cisco, 2014).

## Recommendations

Please switch your network design to a Hierarchical design. Your network has a flat network design. Flat networks lack segmentation and are more vulnerable to potential security breaches. A hierarchical design organizes them into layers for better traffic management and scalability. Hierarchical Network Design is now the best practice industry-wide to design networks that are reliable, resilient, scalable, and also cost-effective (Jain, 2022).

# Detailed Findings

- Machine designation: Windows11
    - Device Host Name: WINDOWS11-DESKT
    - IP Address: 10.0.2.1/24
    - MAC address: 08:00:27:CB:20:4A
    - Operating System & version: Windows 11 Home 23H2 (nmap identifies it as window 10 1703)
    - Open ports with associated services:
    - 53/tcp open domain PowerDNS Recursor
    - ARP Ping Scan elapsed time:1.92s elapsed (255 total hosts)

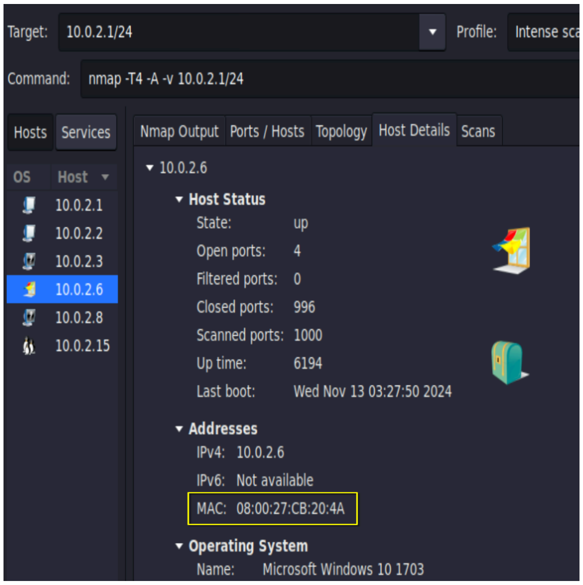
- Machine designation: LinuxServer1
    - Device Host Name: linux-server
    - IP Address: 10.0.2.15/24
    - MAC address: 08:00:27:dd:d8:f8
    - Operating System & version: Ubuntu 22.04.3
    - Open ports with associated services
    - 3306/tcp open mysql MySQL 8.0.37-0ubuntu0.22.04.3
    - ARP Ping Scan elapsed time: 1.95s elapsed (255 total hosts)

- Machine designation: KaliOpenVas
    - Device Host Name (ex. DESKTOP-JE9II55, etc): kali
    - IP Address: 10.0.2.8/24
    - MAC address: 08:00:27:1b:76:b0
    - Operating System & version: Kali Linux 2024.2 (Rolling Release)
    - Open ports with associated services: None found
    - ARP Ping Scan elapsed time: 1.97s elapsed (255 total hosts)

## Information Collection Methodology

I started by turning on each of the virtual machines. I used cmd **ipconfig** (Harwood, 2023) on the Windows 11 machine and learned its IP address. For the Linux Server Machine and Kali Linux machine, I used the cmd **ip a** (Mitchel & Novotny, 2022) to determine each of their IP addresses. The Machine designations were the Virtual Machine names listed in the Virtual Box Menu. From this, I was able to confirm that all 3 machines were in the 10.0.2.[0-255] IP address range.

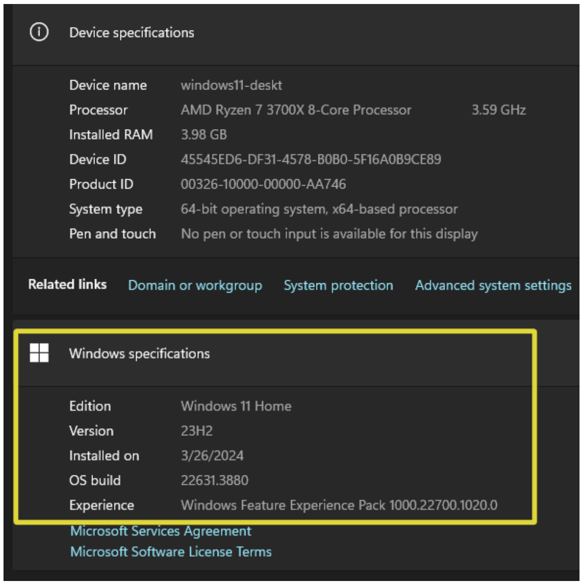
I started by opening Wireshark (Warnicke & Lamping, 2024) and nmap (Lyon, 2009) on my Kali Linux VM. I turned on Wireshark network traffic capture. I ran the following nmap command using Zenmap (Lyon, 2009): **nmap -T4 -A -v 10.0.2.0/24**
This scan searched the virtual network and detected the 3 machines that were in the scope of the assignment. However, it also detected local services running on my devices that were outside the scope of the assignment. After confirming in an assistance request that the scope was only the 3 machines, I ran intensive scans on each of the 3 machines inside the scope 1 at a time.
I started by scanning the the Windows 11 virtual machine using **nmap -T4 -A -v 10.0.2.1/24**. Device Host Name, WINDOWS11-DESKT, was located in the nmap scan in the HTTP header ***"http-title: Welcome | PRTG Network Monitor (WINDOWS11-DESKT)"***



```
Target: 10.0.2.1/24          Profile: Intense sca
Command: nmap -T4 -A -v 10.0.2.1/24

OS  Host        ▾ 10.0.2.6
    10.0.2.1      ▾ Host Status
    10.0.2.2          State:         up
    10.0.2.3          Open ports:    4
    10.0.2.6          Filtered ports: 0
    10.0.2.8          Closed ports:  996
    10.0.2.15         Scanned ports: 1000
                      Up time:       6194
                      Last boot:     Wed Nov 13 03:27:50 2024
                  ▾ Addresses
                      IPv4:  10.0.2.6
                      IPv6:  Not available
                      MAC:  08:00:27:CB:20:4A
                  ▾ Operating System
                      Name:      Microsoft Windows 10 1703
```

Fig. 1: The MAC-Address was listed under the host details tab of the Zenmap GUI, in the addresses section.



Device specifications

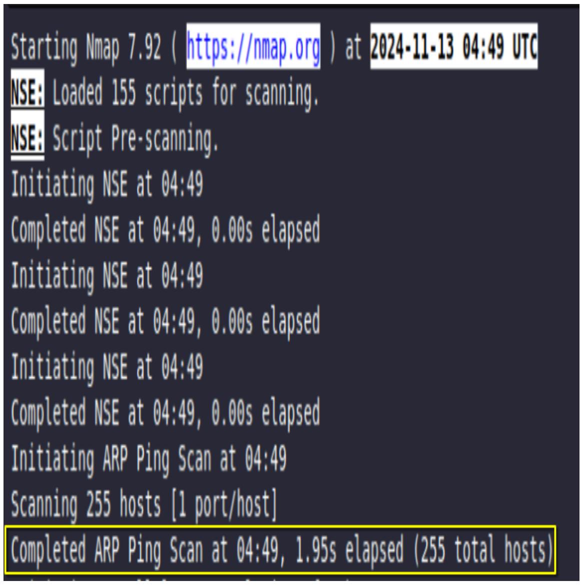
Device name windows11-deskt
Processor AMD Ryzen 7 3700X 8-Core Processor 3.59 GHz
Installed RAM 3.98 GB
Device ID 45545ED6-DF31-4578-B0B0-5F16A0B9CE89
Product ID 00326-10000-00000-AA746
System type 64-bit operating system, x64-based processor
Pen and touch No pen or touch input is available for this display

Related links Domain or workgroup System protection Advanced system settings

Windows specifications

Edition Windows 11 Home
Version 23H2
Installed on 3/26/2024
OS build 22631.3880
Experience Windows Feature Experience Pack 1000.22700.1020.0
Microsoft Services Agreement
Microsoft Software License Terms

Fig 2. The nmap scan host details identify it as Windows 10 1703, but checking the machine device information lists it as Windows 11 Home 23H2.



```
Starting Nmap 7.92 ( https://nmap.org ) at 2024-11-13 04:49 UTC
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 04:49
Completed NSE at 04:49, 0.00s elapsed
Initiating NSE at 04:49
Completed NSE at 04:49, 0.00s elapsed
Initiating NSE at 04:49
Completed NSE at 04:49, 0.00s elapsed
Initiating ARP Ping Scan at 04:49
Scanning 255 hosts [1 port/host]
Completed ARP Ping Scan at 04:49, 1.95s elapsed (255 total hosts)
```

Fig 3. The ARP ping scan information was located at the start of the nmap output.

I used the same methods to acquire the information on the Linux Server.
For the Kali server, I used Linux commands, such as **ip a** (Mitchel & Novotny, 2022). The nmap scans did not show the information as it was the machine running the scans.

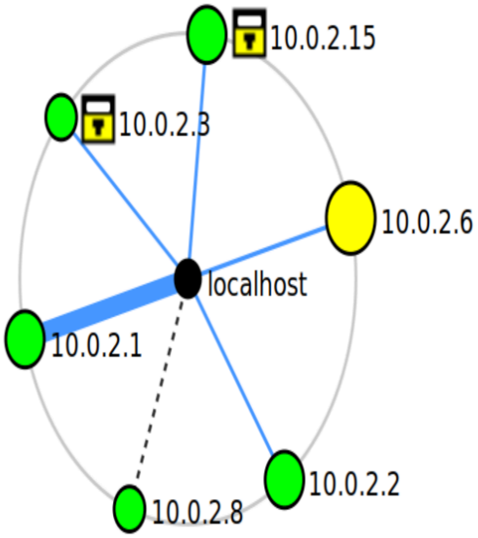
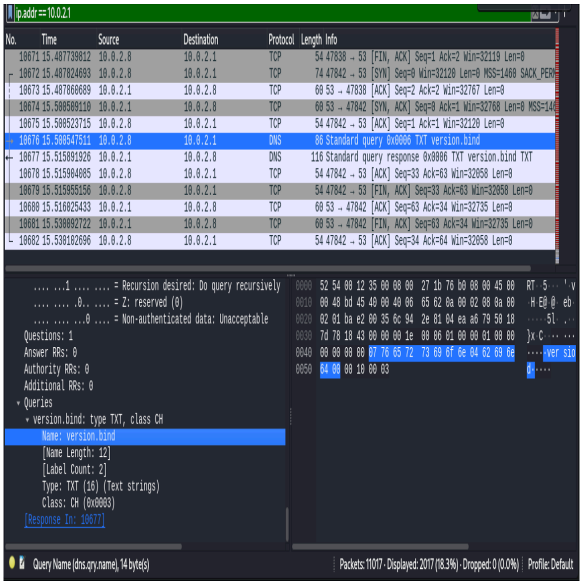


Fig 4. This is the current topology map for the network. All devices on the network have access to each other because it is arranged as a flat network.



```
ip.addr == 10.0.2.1

No.   Time          Source    Destination  Protocol Length Info
10671 15.487739812  10.0.2.8  10.0.2.1     TCP      54 47838 → 53 [FIN, ACK] Seq=1 Ack=2 Win=32119 Len=0
10672 15.487824693  10.0.2.8  10.0.2.1     TCP      74 47842 → 53 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM
10673 15.487860689  10.0.2.1  10.0.2.8     TCP      60 53 → 47838 [ACK] Seq=2 Ack=2 Win=32767 Len=0
10674 15.500509110  10.0.2.1  10.0.2.8     TCP      60 53 → 47842 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=146
10675 15.500523715  10.0.2.8  10.0.2.1     TCP      54 47842 → 53 [ACK] Seq=1 Ack=1 Win=32120 Len=0
10676 15.500547511  10.0.2.8  10.0.2.1     DNS      86 Standard query 0x0006 TXT version.bind
10677 15.515891926  10.0.2.1  10.0.2.8     DNS     116 Standard query response 0x0006 TXT version.bind TXT
10678 15.515904085  10.0.2.8  10.0.2.1     TCP      54 47842 → 53 [ACK] Seq=33 Ack=63 Win=32058 Len=0
10679 15.515955156  10.0.2.8  10.0.2.1     TCP      54 47842 → 53 [FIN, ACK] Seq=33 Ack=63 Win=32058 Len=0
10680 15.516025433  10.0.2.1  10.0.2.8     TCP      60 53 → 47842 [ACK] Seq=63 Ack=34 Win=32735 Len=0
10681 15.530092722  10.0.2.1  10.0.2.8     TCP      60 53 → 47842 [FIN, ACK] Seq=63 Ack=34 Win=32735 Len=0
10682 15.530102696  10.0.2.8  10.0.2.1     TCP      54 47842 → 53 [ACK] Seq=34 Ack=64 Win=32058 Len=0

    .... ...1 .... .... = Recursion desired: Do query recursively
    .... .... .0.. .... = Z: reserved (0)
    .... .... ...0 .... = Non-authenticated data: Unacceptable
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 0
▾ Queries
  ▾ version.bind: type TXT, class CH
      Name: version.bind
      [Name Length: 12]
      [Label Count: 2]
      Type: TXT (16) (Text strings)
      Class: CH (0x0003)
  [Response In: 10677]

0000  52 54 00 12 35 00 08 00  27 1b 76 b0 08 00 45 00   RT··5···'·v
0010  00 48 bd 45 40 00 40 06  65 62 0a 00 02 08 0a 00   ·H·E@·@·eb·
0020  02 01 ba e2 00 35 6c 94  2e 81 04 ea a6 79 50 18   ·····5l·.··
0030  7d 78 18 43 00 00 00 1e  00 06 01 00 00 01 00 00   }x·C··· ···
0040  00 00 00 00 07 76 65 72  73 69 6f 6e 04 62 69 6e   ·····ver sio
0050  64 00 00 10 00 03                                  d·····

Query Name (dns.qry.name), 14 byte(s)   Packets: 11017 · Displayed: 2017 (18.3%) · Dropped: 0 (0.0%)   Profile: Default
```

Fig 5. The screenshot of Wireshark here shows that the Windows nmap scan was primarily done using TCP at the Transport Layer. The query of the Domain Name System (DNS) was done at the Application Layer. (Warnicke & Lamping, 2024)

# References

1. Cisco. (2014, May 9). *Cisco Networking Academy Connecting Networks Companion Guide: Hierarchical Network Design > Objectives*. Cisco Press. Retrieved November 15, 2024, from https://www.ciscopress.com/articles/article.asp?p=2202410
2. Jain, S. (2022, November 17). *Hierarchical Network Design*. GeeksforGeeks. Retrieved November 15, 2024, from https://www.geeksforgeeks.org/hierarchical-network-design/
3. Harwood, R. (2023, February 3). *ipconfig*. Microsoft Learn. Retrieved November 15, 2024, from https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/ipconfig
4. Mitchel, J., & Novotny, J. (2022, June 16). *How to Use the ip Command in Linux*. Linode. Retrieved November 15, 2024, from https://www.linode.com/docs/guides/how-to-use-the-linux-ip-command/
5. Warnicke, E., & Lamping, U. (2024, 08 28). *Wireshark User's Guide*. Wireshark. Retrieved November 15, 2024, from https://www.wireshark.org/docs/wsug_html/
6. Lyon, G. “. (2009, January 1). *Nmap Network Scanning*. Nmap Network Scanning. Retrieved November 15, 2024, from https://nmap.org/book/toc.html