

Colby Heilner
Professor Torres
9/15
IT 120
Lab 4

Part A: NDG Security+ v3

For this lab, please follow the below instructions:

- (Links to an external site.)perform a hardware inventory of all devices within this network (except the Kali box).

-Manual in an Excel spreadsheet, or scripted

Okay first I looked at all the machines I would have to inventory. Because I get the OS from netlab I can sort them early and know which of my scripts I will be able to use.

I like to start with a winver is possible it will open this below



Now my scripts for the rest of the WIN boxes. This is simple but I will improve on it as I learn more commands. "I tried to use wmic but it had a lot of bad formatting issues"

```
File Edit Format View Help
@ECHO OFF
set output_file=hardwarescan.txt

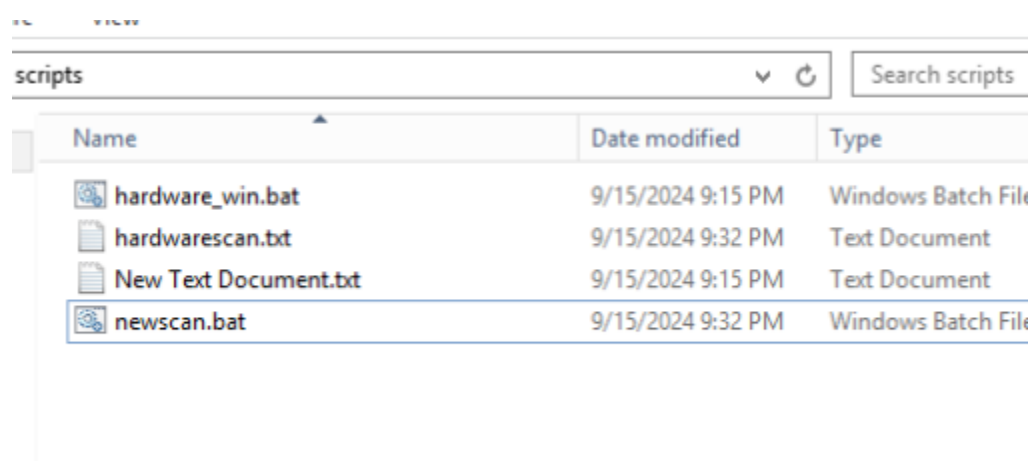
systeminfo > %output_file%

fsutil volume diskfree c: >> %output_file%

ipconfig /all >> %output_file%

tasklist >> %output_file%
```

I ran these and they worked here is SS of the folder with my output and script.



Name	Date modified	Type
hardware_win.bat	9/15/2024 9:15 PM	Windows Batch File
hardwarescan.txt	9/15/2024 9:32 PM	Text Document
New Text Document.txt	9/15/2024 9:15 PM	Text Document
newscan.bat	9/15/2024 9:32 PM	Windows Batch File

For Linux I had to make bash scripts they start with #!/bin/bash

Here is my Linux scan. I also show here the file that it created when it ran.

```
root@Ubuntu:/home/student/Documents# pwd
/home/student/Documents
root@Ubuntu:/home/student/Documents# ls
hardwareinfo.txt  linuxHardware
root@Ubuntu:/home/student/Documents# cat linuxHardware
#!/bin/bash

OUTPUT_FILE="hardwareinfo.txt"

> "$OUTPUT_FILE"

df -h >> "$OUTPUT_FILE"

free -l >> "$OUTPUT_FILE"

lscpu >> "$OUTPUT_FILE"

uname -a >> "$OUTPUT_FILE"

lsb_release -a >> "$OUTPUT_FILE"

ip addr >> "$OUTPUT_FILE"

lspci >> "$OUTPUT_FILE"

lshw -c display 2>/dev/null >> "$OUTPUT_FILE"

ps aux >> "$OUTPUT_FILE"

echo "Scan Done"

root@Ubuntu:/home/student/Documents#
```

Part B: NDG Security+ v3

- perform a software inventory of all devices within this network (except the Kali box).
 - manual in an Excel spreadsheet, or scripted

For windows I will be looking at the Win 16 box. Sadly, windows don't like to give you a lot about application installed and sometimes a lot of them can slip through commands. here is a simple PowerShell command to grab applications.

```
PS C:\Users\Administrator> get-wmiobject -class win32_product | select-object name, version
name                                     version
----                                     -
VMware Tools                           10.0.0.3000743
Microsoft Visual C++ 2008 Redistributable - x64 9.0.30729.6161 9.0.30729.6161
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.4148 9.0.30729.4148
```

For Linux we can get all the good stuff using

```
root@Ubuntu:/home/student/Documents# dpkg -l > installedapps.txt
root@Ubuntu:/home/student/Documents# ls
hardwareinfo.txt  installedapps.txt  linuxHardware
```

Here are some Wireshark versions from my **installedapps.txt**

```
ii  wireshark                        1.6.7-1
er - GTK+ version
ii  wireshark-common                1.6.7-1
er - common files
ii  wodim                          9:1.1.11-2ubuntu2
```

A lot of these scripts were pulled from my newly created GitHub for Identify in NIST.

Part C: NDG Security+ v3

- Research if the software is the most current version
- Research any vulnerabilities in the software or hardware

Taking an easy one from Linux we can look at Wireshark. VER 1.6.7

🚩 CVE-2013-4074 Detail

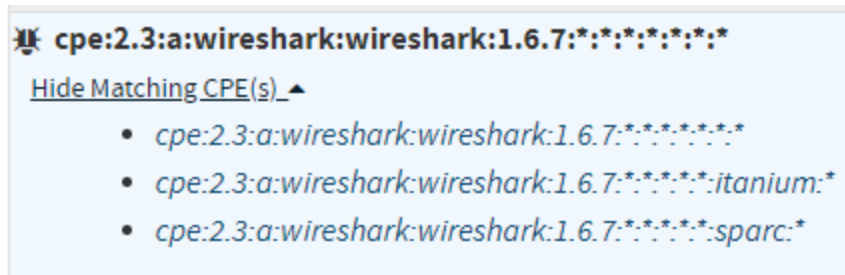
MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Description

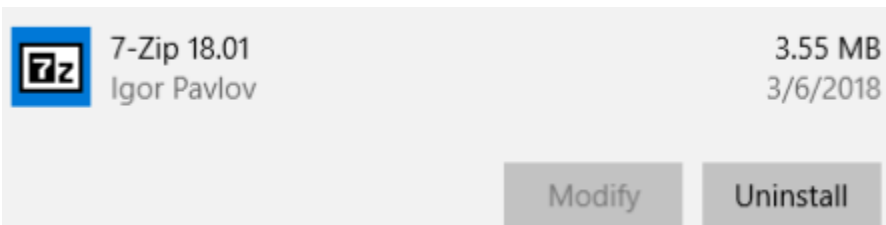
The dissect_capwap_data function in epan/dissectors/packet-capwap.c in the CAPWAP dissector in Wireshark 1.6.x before 1.6.16 and 1.8.x before 1.8.8 incorrectly uses a -1 data value to represent an error condition, which allows remote attackers to cause a denial of service (application crash) via a crafted packet.

Looking at the affected versions we can see this one is at risk!



Now let's look at something on the windows side of things.

Here we have 7-zip program VER 18.01



Here is a vulnerability to any versions before this one, making ours vulnerable by default!

The “and before” is key here.

Description

Incorrect initialization logic of RAR decoder objects in 7-Zip 18.03 and before can lead to usage of uninitialized memory, allowing remote attackers to cause a denial of service (segmentation fault) or execute arbitrary code via a crafted RAR archive.

Metrics

CVSS Version 4.0
CVSS Version 3.x
CVSS Version 2.0

NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displayed.

CVSS 3.x Severity and Vector Strings:

NIST: NVD
Base Score: 7.8 HIGH
Vector: CVSS:3.0/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

Part D: 5 points of extra credit

- List any technical, physical, administrative, or compensating controls that can be used for this exercise

For this lab I would say that having automated software updates run on systems could be an administrative control. This could negate any possibility of old software being abused by threat actors.

For compensating, if we know our hardware or software are not up to date. Then maybe we will have advanced monitoring in place. Splunk could be a good

example. I never used it, but it provides detailed log analysis, allowing you to act when something outdated gets exploited. Because it will.