

Logan Flowers & Garrett Fraley

IT 359-001

Final Project Report

NetworkRecon.sh

Introduction & Purpose:

The first thing performed when doing any penetration testing, specifically in HackTheBox, is an NMAP scan to find open ports and to find anything that you may be able to utilize to gain access. Honestly, that can become really annoying and tends to take a long period of time for what you receive back. NetworkRecon.sh is a script that we wrote to provide an alternative option for performing IP scans that provides more concise feedback at a much faster rate. This tool also has additional options that allow for more in-depth scans as necessary.

Technical Implementation:

We started with a basic set of components that we wanted our script to perform:

- Alerts for suspicious open ports/services
- Bash script
- Give customization options/arguments to allow different scan options.
- Use Nmap `-sp` and possibly `arp -a` to perform scans to determine other machines on the network.

- Use Nmap -p to scan for ports that are less likely to be secure, like telnet, HTTP, or FTP.
 - Perform service fingerprinting based on results to determine which versions of services are being used on questionable ports.
 - Perform device fingerprinting to determine what OS machines are using or what other types of devices are on the network.
 - Perform a nmap scan on each IP found or provide option to scan specific IPs to determine if any device has vulnerabilities.
 - Use netcat to perform basic network performance/bandwidth scans
 - Compare results to a security database to determine any additional vulnerabilities.
 - Automatically generate a security report for the network, with highlights for anything that is suspicious or vulnerable.
- save the script's output in a file

We then decided to focus on the priorities of things that we figured would be the most helpful.

We chose to focus primarily on nmap, netcat, and creating a formatted output file that would provide helpful information at a glance and perform at a quicker pace than a standard nmap scan.

We utilized paired programming to produce a better end product, programming within a HackTheBox virtual machine to allow for testing while programming. This worked well for us because we were able to test this on HackTheBox hosted machines, providing quick feedback for any changes we needed to make without performing extensive scanning on machines we do not own. This also ensured that our scanner worked well in different scenarios, since each box is different and generally has different ports/vulnerabilities for teaching reasons.

Tools used:

- All programming was saved within the github repository, with programming being performed in a .sh file on a HackTheBox virtual machine.
- We utilized ChatGPT to clean up our code formatting and assist in debugging when necessary.
- All testing was performed on HackTheBox virtual machines.

References for Programming

- KodeKloud Notes — Pipefail and Bash streams.
<https://notes.kodekloud.com/docs/Advanced-Bash-Scripting/Streams/Pipefail>
- kvz.io — Best Practices for Writing Bash Scripts.
<https://kvz.io/blog/bash-best-practices.html>
- Nmap Reference Guide (official).
<https://nmap.org/book/man.html>
- Linux.com — Linux Security Fundamentals: Introduction to nmap.
<https://www.linux.com/news/linux-security-fundamentals-part-6-introduction-nmap/>

Justification & Analysis

Our goal with this project was to create a clean and concise script that will essentially scan an IP and, in return, provide straightforward information about the device to help you determine entry points. Ideally, we wanted this script to run quickly and give an automated response that can be quickly looked at to find information that could be useful for performing penetration testing. We also wanted to provide a text file that would contain this information that

allows for a quick reference, either for write-ups or just as a reminder if you must step away from a project for some time.

As we programmed, we were able to determine portions of our program that would not work well for this project and were not worth spending the time to get to work. Initially, we had masscan, arp, and a security database comparison tool in our list of desired utilities for this project. Masscan was something we already knew we probably wouldn't use, since the tool is meant to essentially perform a nmap scan of the whole internet. Performing a masscan scan on a network can cause issues with ISPs, and we did not want to risk any form of repercussions from testing. Arp did not work well in testing with HackTheBox and we did not want to test on another network, so we decided to not utilize arp for this tool. The database comparison tool is still a good idea, and I wish we were able to implement it because it would be pretty helpful, but realistically we did not have time to implement this with this tool and ultimately decided to leave this off as well.

Conclusion

NetworkRecon.sh successfully implements a fast, single-host reconnaissance workflow that reduces the time between starting a scan to the full output compared to a full nmap scan while preserving the ability to expand into deeper checks as necessary. The script provides a single consolidated report suitable for writeups and repeatable testing. Future improvements that could be made include CVE mapping, LAN discovery options, and database comparison. Overall, we think that this project was incredibly helpful in both increasing our knowledge of penetration testing but also in allowing us to research and learn more about bash scripting. Hopefully this

tool will allow others to quickly perform scans of networks and assist in creating writeups for penetration testing applications.