CS378: Ethical Hacking The University of Texas at Austin

Group 6: Network Mapping Tool

Joel Henning, Jeffrey Li, Tipparat Umrod, Xinyi Wang

Introduction

The network assessment methodology can be roughly defined by the following steps: network discovery, host discovery, service discovery, host enumeration, service enumeration, network topology mapping, vulnerability testing, reporting, and remediation. One of the most crucial steps in almost any network environment is footprinting. Network footprinting is loosely defined as a means or process of accumulating data and defining a baseline of operation (in respects to vulnerability prevention/assessment). Some of the information obtained includes open/closed ports, banners, operating system details, system version/architecture, much of which can provide invaluable information and reveal potential system vulnerabilities. It can also improve the ease at which they can be exploited.

The primary development and use of our tool serves as a means to automate this important subset of the network assessment methodology. We noticed that many of the more modern and basic command lines tools used widely in industry practices (i.e. nmap, snmpwalk, finger) simply end up dumping all of the information to console. In practice, and depending on the thoroughness or options selected for the scan, the amount of data collected can be rather tedious to debug or analyze. Therefore, the primary purpose of our tool is to not only automating the process, but also to export the data to a series of a more readable and user-friendly formats such as PDF sortable tables.

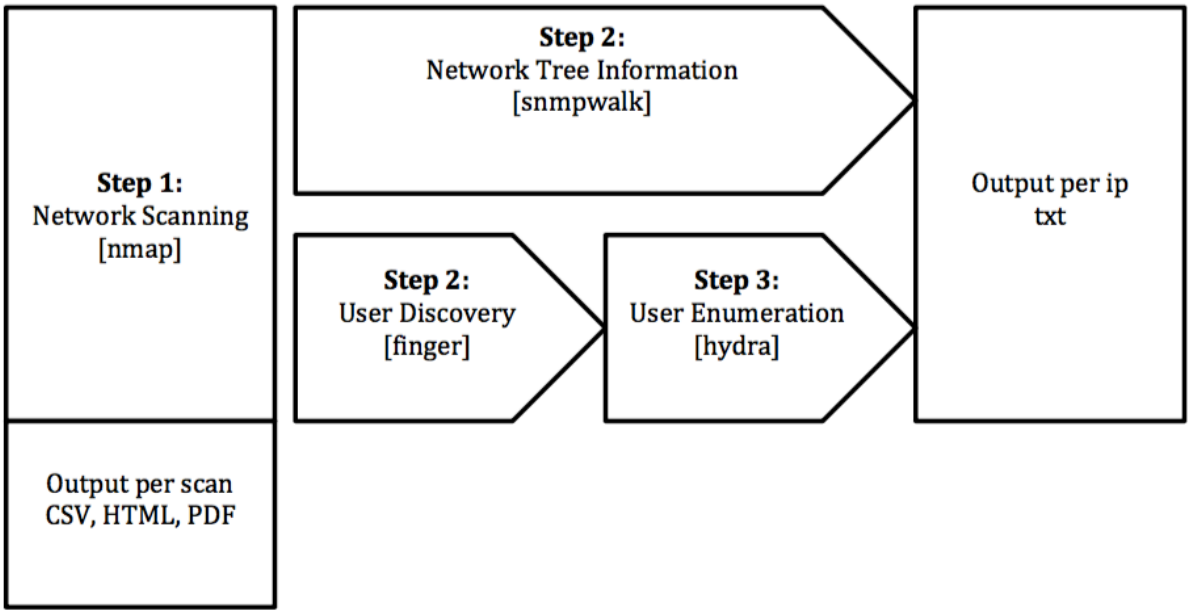
Tools Used:

- **Security Scanner(s):**
  - Nmap: Nmap (Network Mapper) is a security scanner used to discover hosts and services on a computer network.
- **Command-Line Tools:**
  - snmpwalk: snmpwalk is an SNMP application that uses SNMP GETNEXT requests to query a network entity for a tree of information.
  - finger: The finger displays information about the system users.
  - hydra: Hydra's the "tool of choice" for brute-force cracking a remote authentication service. It can perform rapid dictionary attacks against more than 50 protocols, including telnet, ftp, http, https, smb, several databases, and more.
  - ssh: The ssh client is is a program for logging into a remote machine and for executing commands on a remote machine. It is intended to replace rlogin and rsh, and provide secure encrypted communications between two untrusted hosts over an insecure network.
- **Export Tools:**
  - ReportLab: ReportLab offers open source, cross platform libraries written in Python for PDF creation, as well as an XML parser and text preprocessor.
  - pyPDF: pyPDF is a pure python library built as a PDF toolkit. It is capable of extracting document information, splitting documents page by page, merging documents page by page, cropping pages, merging multiple pages into a single page, and encrypting and decrypting PDF files.

Running the Tools:

$ python combine.py -h #for help with other option
$ python combine.py <ip_range> #basic command

Flow Chart Depicting the tool



Comprehensive nmap scan

To accomplish our tool, we focused on three different parts of the footprinting process. We first began with a comprehensive nmap scan that checks the most common ports for each box on the network as well as does OS discovery to see what operating systems are being run on each of the boxes. This is the first step so that the general network topology can be discovered as well as provide any knowledge of any services running that may be potentially exploitable.

SNMP

Once the nmap scan is complete, we pipe the results to a simple output file where it can be easily parsed. The next step involves performing an snmpwalk on the network to discover the name of the router(s) and to see if there are any tftp servers running. In order to perform snmpwalk on the router(s) successfully, brute forcing community string from password.lst was implemented. Depending on the brand (usually Cisco) and version of the routers, we can potentially grab its configuration file providing even more additional information. It is important to note the significance of these files as they contain software commands used to customize the functionality of the routing device as well as identify/provide other administrative information.

Finger

Following this step, finger is run on any live box that has port 79 (finger port) open. What the 'finger' command will do is see if there are any users that are logged into that specific box. If so, it can provide the user's login information including their real name, terminal name, home directory, shell information, and administrative access privileges.

User enumeration

As the last step, any users that are obtained from 'finger' are then used to try and 'ssh' into any boxes that have a live/open port 22 (ssh port). Informally, to do this, the tool takes the obtained users and uses hydra to try and brute-force the password using the 'password.lst' from the infamous John the Ripper password cracker. The command is:

hydra -l root -P /usr/share/john/password.lst ssh://[ip]

Brute forcing ssh is the next step for our tool because it is a good way to see if any of the user names that were found with the finger tool can be used to get into any boxes. Upon sshing into a machine on the network an attacker would then try and install malicious software or put back doors in place. That's why this would be the logical next step after mapping out the network. It is also possible to provide a list of usernames or an even more robust list of passwords to try and break into ssh.

Development/Responsibilities

Since our tasks are highly modularized, each individual member were assigned specific tasks:

Tipparat Umrod (snmpwalk)

After running the initial nmap scan, an output file and dictionary is generated. run_services()then takes in the resulting dictionary and executes an instruction set based on the services requested. In this case, if the Device type is a router (which can potentially have an snmp service available), the script will then run an snmpwalk command with the community string set to publicfor that particular IP address, since public is a default community string. This can be demonstrated as follows:

snmpwalk -v1 -c public <ip_address>

If the console output from snmpwalk is Timeout, then the community string will be brute-forced using John the Ripper's password.lst.

snmpwalk -v1 -c <guessing_string> <ip_address>

If successful, the snmpwalk output will be saved to an output file.

Joel Henning (finger/user enumeration/hydra cracker)

After the nmap scan is completed, finger will be run on any box that has port 79 open. fingerwill return the username of any users logged in on that specific box. The command below is run by the program with a list of the IP addresses:

finger -l @<ip_address>

Any usernames that are returned by the finger command are then run with hydra to try and brute-force ssh login credentials for those users. The password list used by the program is password.lst from John the Ripper which is the smaller of John's password files. Hydra is a brute forcing tool that can be used on any live service that requires a username and password login. This is to try and get access to any boxes that happen to have weak credentials to ssh in. The command to run hydra is:

hydra -l <username> -P <password_list> <service>://<ip_address>

Jeffrey Li (nmap parser/export to bootstrap table)

Step 1 required manually parsing the nmap output. Some of the most important headers (MAC Address, device type, port services, operating system details, etc) were selected from the scan, while some were intentionally left out. The headers and their associated values were then added to a dictionary, which was then converted and written to a .csv file. The file was then taken in as an argument to the `csv_to_html' script and converted into an HTML bootstrap table (provides column/feature sorting).

Xinyi Wang (export to PDF)

After the initial nmap scan (which is exported to a generic text file), the export script takes in this file as an file input stream and writes it to a PDF. To create a PDF file using our PDF converter tool, simply do the following (the script only requires a filename as input):

$ python conver2PDF.py [-o] YOUR_NMAP_RESULT_FILE_NAME

The program will parse the input from the nmap result file and output it to a pdf file calledscanOutput.pdf. Your nmap result file will be ideally pure text file as the input read it in and take it as strings. The choice in the bracket is optional.

Complications

During the development and testing phases of our network tool, we ran across a series of rather tedious problems. Due to the fact that each team member was using a different version of nmap and on different systems, the inherent formatting of the scans' outputs weren't always consistent. This required us to manually parse for specific headers of the scan as well as possible unique or edge cases such as timeouts or error warnings. When working with thorough and complete nmap scans, we also noticed that our bootstrap tables were consistently producing extra columns. Many of our export tools were developed using comma delimited input files, which made the nmap output file particularly tricky to work with due to the fact that there were extra commas scattered throughout the result set. Cisco v. 500, 1000, 1500. This required a multi-pass approach that checked and removed/replaced the extra columns from the csv file and added them back in the final formatting.

There were other issues regarding the export tools we used. Most of the initial tools we considered using that were better documented and had more community support were only supported by python3. With ReportLab, we had significant problems in getting the PDF to export correctly and the formatting made it difficult to control text position, sizing and other features that would have provided better usability for users.

One of the biggest inherent development flaws of our tool is its inability to actively support the vast capabilities and options provided by nmap. As mentioned above, due to the varying output formats of nmap, parsing was significantly more difficult. Therefore, we actively selected some of the more important headers and hardcode them in our parser.

One issue with using the hydra tool is that brute forcing in general can take a very long time. Even with multiple threads all running at the same time the tool can take a long time. For the demonstration the environment had to be set up so that the boxes with ssh could be broken quickly with easy passwords, however in a real network environment the hydra command could take a very long time to complete. Brute forcing is also very loud so our tool would not be the best for use in a penetration test because it could be easily detected.

Recommendations

One of the primary reasons for our tool is to facilitate the network mapping process by making the initial vulnerability assessment and network analysis easier. Through this development and automation process, we were forced to remove some of the functionalities of some of the command line tools. Therefore, we'd like to expand on the capabilities provided by these tools and utilize the various options/arguments provided by nmap. To tackle the expansive capabilities of nmap, the next step would be creating a GUI interface that allows users to selectively choose nmap options as well as headers/values to export. As a tool that serves to enhance usability, we understand that simply exporting and re-dumping the results is counter-intuitive.

PDF Export

There are a lot of different tools available that provide relatively simple libraries to use in regards to PDF creation. However, many of these were only available to python3. Therefore, an essential requirement for the future would to refactor our current code to be compatible with python3 as well as python2. However, in regards to our current implementation, we could avoid using "draw strings" during the PDF creation that prevents words to be searchable and indexable. We could also provide the feature that allows for user preferences or a variation of a table-of-contents that only reports the results that satisfy a particular requirement. Right now, we only have the option to highlight operating systems. Rather than dumping the nmap scan with little processing to a PDF, we'd like to be able to reformat the input and provide PDF indexing to allow for greater usability.

SNMP

For our snmpwalk service, we could utilize 'snmpget' to retrieve data from a remote host and obtain specific information such as the host's name and authentication information. Due to the nature of this task, it could be easily paired and done concurrently with our current user enumeration and brute-force cracking to reduce the overall runtime. Snmp also provides the level of access of the object (such as read or read/write), therefore we could brute the community strings for READ-WRITE(default = private) as well as READ-ONLY (default=public) to assess and determine any other vulnerabilities.

Other

Finally, as a point-and-shoot tool, we'd like to incorporate other options and other command line tools such as metasploit, which has built-in support for Nessus integration. As a tool that heavily relies on network landscaping and vulnerability assessment, using Nessus could provide invaluable information in regards to other potential vulnerabilities on any hosts. The vulnerability assessment and report created by Nessus could then be combined with our current report information.