Farzad Mohammadi & Ben Finkelstein

CSC 4610

Tisha Gaines

4/12/21

Port Scanning and Malware Protection

Port scanning is a technique commonly used by IT professionals to identify open ports on a network in order to maximize the sending and receiving of packets, but it also has a history of being a technique used by hackers in order to gain remote access to hosts on a network. The process of port scanning begins with identifying hosts on the network via IP addresses. An article by *avast.com* on port scanning highlights some common ports used for topics we had discussed in class, such as Port 20, used for FTP and data transfer, or Port 53, which is used for the DNS information for website IP addresses. The port scan method utilizes both TCP and UDP protocols to identify the open ports.

This technique, however, has been used for a number of years by hackers to identify open ports for infiltration and malware attacks. Over time, commonly used ports have become much more secure, but some can still be attacked by hackers. Intrusion detection and firewalls can be set up to prevent these attacks, but these are not always 100% effective in stopping the cyber-criminals. Port scanning can be a useful tool to identify which open ports may be at risk of infiltration and allow the administrator to take steps necessary to prevent this.

This port scan program utilizes a number of different components. It begins with setting up a socket connection to send network data to a port. This is done to determine the current status of the port. The action is completed over a large thread pool of ports and repeated utilizing

loops. The procedure of scanning multiple ports is executed asynchronously using concurrent.futures which provides a high-level interface that interacts with pools of threads and processes. The program allows the scan to be completed in any input IP addresses with an inputted timeout for socket connection.

The scan reveals five different details. It reveals each open port with its service name, date, and time. The duration of the entire test is revealed at the end of the scan. There is not much else which can be revealed for the intended purposes of this program. It is merely a piece of the process of malware protection for an entire system.

There were a number of solutions and situations encountered while typing up this program. It was simple to execute a socket connection and scan a single open port as socket connections have been completed throughout the semester. It took a while to develop loops to search for multiple ports. We researched and found concurrent.futures to simplify the implementation of loops and the large pool of ports to analyze. It was assistful in other parts of the program.

The number of registered ports and well-known ports are 49,151 which were learned through class lectures. The large number of ports took time and testing to decrease the value. The importance of minimizing the pool of ports to scan is time and necessity. We did not want the program to search for 20,000 ports which are never open in any location. The python module, concurrent.futures, significantly sped up the process of looking through all of the ports. We tested our localhosts, webhosts, and other hosts to discover our highest values with timeout values set between 1 and 5. This assisted in breaking down the number of analyzed ports to 30,000 since the open ports were in proximity.

One of the details of the scanned ports was the ports' service name. The getservbyport() python function from the socket module was employed to acquire the service names. We got the service name of mainly known ports like port 80, 135, 445 which are html, epmap, and microsoft-ds. We could only acquire the service names of a few number of ports above 5,000. The reasoning behind this function not being able to acquire all of the ports' service names is that there isn't one available for each port. The only ports with names are those that are readily available and known.

A challenge which occurred was running the program on Apple Macbook where there would be an error due to the threadpoolsize. The error was "OSError: [Errno 24] Too many open files" . We adjusted the threadpoolsize multiple times and it functioned correctly at threadpoolsize of 256. We could not solve this problem and figure out the reason for the error in time so we left it be. The program's procedure and procedure to create this program were simple and emphasized the importance of cybersecurity as noted before.

Port scanning will most likely remain relevant and more advanced as we progress as a more technologically advanced and interconnected society. However, as port scans and firewall protections get more advanced, so too will the cyber-criminals' techniques to identify open ports. It will be up to IT professionals and computer scientists to develop better methods to identify and protect open and at-risk ports beyond a basic port scan.

Citations:

"What Is Port Scanning?" *Avast*, 2021, www.avast.com/en-us/business/resources/what-is-port-scanning.

"Malicious Port Scanning and How to Recognize It." *ExtraHop*, 2021, www.extrahop.com/resources/attacks/malicious-port-scanning/.

"What Is a Port Scan and How Does It Work?" *Fortinet*, 2021, www.fortinet.com/resources/cyberglossary/what-is-port-scan.