Lab08

Network Trace analysis and Attacks: First Part

[You must not attack any network without authorization! There are also severe legal consequences for unauthorized interception of network data.]

Introduction

This project will introduce you to common network protocols, the basics behind analyzing network traces from both offensive and defensive perspectives, and several local network attacks.

Objectives

• Gain exposure to core network protocols and concepts.

• Understand offensive techniques used to attack local network traffic.

• Learn to apply manual and automated traffic analysis to detect security problems

8.1 Exploring Network Traces (15 points)

Security analysts and attackers both frequently study network traffic to search for vulnerabilities and to characterize network behavior. In this section, you will examine a packet trace from a sample network we set up for this assignment. You will search for specific vulnerable behaviors and extract relevant details using the Wireshark network analyzer (http://www.wireshark.org). Get the network trace ***8_1.pcap*** from **here** and examine it using Wireshark. Provide concise answers to the following questions using submission format.

8.1.1 MAC, IP address (5 points)

Multiple hosts sent packets on the local network.

1. What are their MAC addresses?
2. What are their IP addresses?

What to submit: Submit a text file named 8.1.1_mac.txt that contains the MAC addresses of hosts, and a text file named 8.1.1_ip.txt contains the hosts' IP addresses. Write one address per line in the same order for both MAC address and IP address.

8.1.2 FTP server (5 points)

One of the clients connects to an FTP server during the trace.

1. What is the DNS hostname of the server it connects to?
2. Is the connection using Active or Passive FTP?

What to submit: Submit a text file named 8.1.2_dns.txt containing DNS hostname, and a text file named 8.1.2_ftp.txt containing whether it is Active or Passive FTP.

8.1.3 HTTPS connection (5 points)

The trace shows that at least one of the clients makes HTTPS connections to sites other than Facebook. Pick one of these connections and answer the following:

1. What is the domain name of the site the client is connecting to?

What to submit: Submit a text file named 8.1.3_domain.txt containing your answer.

2. During the TLS handshake, the client provides a list of supported cipher suites. List the cipher suites. Refer to the website with a list of known cipher suites table http://www.thesprawl.org/research/tls-and-ssl-cipher-suites/. Double check whether cipher suite name matches from the given page.

What to submit: Submit a text file named 8.1.3_client.txt where each line contains each cipher suite's name. Refer to the website with a list of known cipher suites table http://www.thesprawl.org/research/tls-and-ssl-cipher-suites/. Double check whether cipher suite name matches from the given page.

3. What cipher suite does the server choose for the connection?

What to submit: Submit a text file named 8.1.3_server.txt containing the corresponding cipher name.

8.1.4 Facebook traffic analysis (5 points)

One of the clients makes a number of requests to Facebook.

1. Even though logins are processed over HTTPS, what is insecure about the way the browser is authenticated to Facebook?

What to submit: Submit a text file named 8.1.4_insecurity.txt containing your answer.

2. How would this let an attacker impersonate the user on Facebook?

What to submit: Submit a text file named 8.1.4_impersonate.txt containing your answer.

3. How can users protect themselves against this type of attack?

What to submit: Submit a text file named 8.1.4_protect.txt containing your answer.

4. What did the user do while on the Facebook site?

What to submit: Submit a text file named 8.1.4_user.txt containing your answer.

8.2 Anomaly Detection (35 points)

In 8.1, you manually explored a network trace. Now, you will programmatically analyze trace data to detect suspicious behavior. Specifically, you will be attempting to identity port scanning. Port scanning is a technique used to find network hosts that have services listening on one or more target ports. It can be used offensively to locate vulnerable systems in preparation for an attack, or defensively for research or network administration. In one port scan technique, known as a SYN scan, the scanner sends TCP SYN packets (the first packet in the TCP handshake) and watches for hosts that respond with SYN+ACK packets (the second handshake step).
Since most hosts are not prepared to receive connections on any given port, typically, during a port scan, a much smaller number of hosts will respond with SYN+ACK packets than originally received SYN packets. By observing this effect in a packet trace, you can identify source addresses that may be attempting a port scan.

Your task is to develop a Python program that analyzes a PCAP file in order to detect possible SYN scans. You should use a library for packet manipulation and dissection: either dpkt or scapy. Both are available in most package repositories. You can find more information about dpkt at https://code.google.com/p/dpkt/ and view documentation by running pydoc dpkt, pydoc dpkt.ip, etc.; there's also a helpful tutorial here: http://jon.oberheide.org/blog/2008/10/15/dpkt-tutorial-2-parsing-a-pcap-file/. To learn about scapy, visit http://www.secdev.org/projects/scapy/.

Your program will take one argument, the name of the PCAP file to be analyzed, e.g.: python2.7 detector.py capture.pcap

The output should be the set of IP addresses (one per line) that sent more than 3 times as many SYN packets as the number of SYN+ACK packets they received. Your program should silently ignore packets that are malformed or that are not using Ethernet, IP, and TCP.

A sample PCAP file captured from a real network can be downloaded at ftp://ftp.bro-ids.org/enterprise-traces/hdr-traces05/lbl-internal.20041004-1305.port002.dump.anon. (You can examine the packets manually by opening this file in Wireshark.) For this input, your program's output should be these lines, in any order:

128.3.23.2
128.3.23.5
128.3.23.117
128.3.23.158
128.3.164.248

128.3.164.249

What to submit Submit a Python program that accomplishes the task specified above, as a file named 8.2.py. You should assume that dpkt 1.8 and scapy 2.3.1 are available, and you may use standard Python system libraries, but your program should otherwise be self-contained. We will grade your detector using a variety of different PCAP files.