Project 3: Network Security

This project is due on **April 1, 2015** at **6 p.m.** and counts for 8% of your course grade. Late submissions will be penalized by 10% plus an additional 10% every 5 hours until received. Late work will not be accepted after 20 hours past the deadline. If you have a conflict due to travel, interviews, etc., please plan accordingly and turn in your project early.

Even though working alone is permitted, you are strongly encouraged to work in **teams of two**. Some hardware does not work with some of the tools needed for this project such as Aircrack (Part 2). Build your teams such that at least one member of the team can run the required tools. **Start early!** You can use Piazza for finding partners.

Each team must submit its answers electronically under the `mp3` folder of **one** of the team members in the SVN repository. Details on the filename and submission guideline for each part of the project is listed under that part. In addition to the solutions, submit a file named **partners.txt** with the netids of your team members. Put each netid on a separate line. If you worked alone, include only your netid.

The code and other answers your group submits must be entirely your own work, and you are bound by the Honor Code. You may consult with other students about the conceptualization of the project and the meaning of the questions, but you may not look at any part of someone else's solution or collaborate with anyone outside your group. You may consult published references, provided that you appropriately cite them (e.g., with program comments), as you would in an academic paper.

"You can't defend. You can't prevent. The only thing you can do is detect and respond."
– Bruce Schneier

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.

Read this First

This project asks you to perform attacks, with our permission, against a target network that we are providing for this purpose. Attempting the same kinds of attacks against other networks without authorization is prohibited by law and university policies and may result in *fines, expulsion, and jail time*. **You must not attack any network without authorization!** There are also severe legal consequences for unauthorized interception of network data under the Electronic Communications Privacy Act and other statutes. Per the course ethics policy, you are required to respect the privacy and property rights of others at all times, *or else you will fail the course.* See "Ethics, Law, and University Policies" on the course website.

Part 1. Exploring Network Traces

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 from https://subversion.ews.illinois.edu/svn/sp15-cs461/_shared/mp3/p1.pcap and examine it using Wireshark. Provide concise answers to the following questions. Each response should require at most 2–3 sentences.

1. Multiple hosts sent packets on the local network. What are their MAC and IP addresses?

   What to submit: Submit a text file named `pcap-1.txt` where each line contains the MAC address of a host, followed by a white space, followed by the host's IP address.

2. What type of network does this appear to be (e.g., a large corporation, an ISP backbone, etc.)? Point to evidence from the trace that supports this.

   What to submit: Submit a text file named `pcap-2.txt` containing your answer.

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

   (a) What is the DNS hostname of the server it connects to?
   What to submit: Submit a text file named `pcap-3-a.txt` containing your answer.

   (b) Is the connection using Active or Passive FTP?
   What to submit: Submit a text file named `pcap-3-b.txt` containing your answer.

   (c) Based on the packet capture, what's one major vulnerability of the FTP protocol?
   What to submit: Submit a text file named `pcap-3-c.txt` containing your answer.

   (d) Name at least two network protocols that can be used in place of FTP to provide secure file transfer.
   What to submit: Submit a text file named `pcap-3-d.txt`. List one protocol per line.

4. 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:

   (a) What is the domain name of the site the client is connecting to?
   What to submit: Submit a text file named `pcap-4-a.txt` containing your answer.

   (b) Is there any way the HTTPS server can protect against the leak of information in (a)?
   What to submit: Submit a text file named `pcap-4-b.txt` containing your answer.

   (c) During the TLS handshake, the client provides a list of supported cipher suites. List the cipher suites and name the crypto algorithms used for each.
   What to submit: Submit a text file named `pcap-4-c.txt` where each line contains the cipher suite, followed by a white space, followed by the name of the crypto algorithm used for it.

(d) Are any of these cipher suites worrisome from a security or privacy perspective? Why?
What to submit: Submit a text file named `pcap-4-d.txt` containing your answer.

(e) What cipher suite does the server choose for the connection?
What to submit: Submit a text file named `pcap-4-e.txt` containing your answer.

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

(a) 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 `pcap-5-a.txt` containing your answer.

(b) How would this let an attacker impersonate the user on Facebook?
What to submit: Submit a text file named `pcap-5-b.txt` containing your answer.

(c) How can users protect themselves against this type of attack?
What to submit: Submit a text file named `pcap-5-c.txt` containing your answer.

(d) What did the user do while on the Facebook site?
What to submit: Submit a text file named `pcap-5-d.txt` containing your answer.

Part 2. Network Attacks

In this part of the project, you will experiment with network attacks by cracking the password for a WEP-encrypted WiFi network, decrypting an HTTPS connection, and recovering a simulated victim's username and password.

In the basement area of Siebel Center, near SC 0219, there is WiFi network named `cs461` that is protected with WEP, a common but insecure crypto protocol. We've created this network specifically for you to attack, and you have permission to do so. There is also one or more clients wirelessly connected to this network that makes a connection to a password-protected HTTPS server every few seconds. Your goal is to find the password of a client and log in to this website.

The tools and techniques you use are up to you, but here are some suggestions to get you started:

1. First, you will need to crack the WEP encryption key for the network. There are many online tutorials and automated tools available to help you perform this task. We recommend installing Kali Linux (available from http://www.kali.org) and using Aircrack-ng (https://www.aircrack-ng.org/).

   The WEP cracking process usually involves generating network traffic to speed up the collection of data. For this project, we've made sure that there's sufficient traffic on the network for data collection. When following an Aircrack tutorial, please skip any steps that involve generating traffic.

2. Once you've cracked the WEP key, join the network and examine the traffic. We recommend using Kali and Wireshark. Determine the IP addresses of the client and server, and carefully investigate any services running on these machines. Nmap (http://nmap.org) is a powerful tool for probing remote hosts.

3. In order to discover the client's password, you'll need to decrypt the HTTPS traffic. Wireshark can do this for TLS connections that don't use forward secrecy, if you can provide the server's private key. You may also want read up on the HTTP Basic Authentication method, which is specified in RFC 2617.

What to submit Submit (1) a text file named `attack-1.txt` that contains the WEP key for the network; (2) a text file named `attack-2.txt` that contains the IP addresses of the server (first line) and a client; (3) a text file named `attack-3.txt` that contains a list of the network services running on each machine; one service per line; (4) a text file named `attack-4.txt` that contains the username and password for the HTTPS site the client loads; (5) a text file named `attack-5.txt` that contains the secret contents of the website the client tries to load; (6) a text file named `attack-6.txt` that contains a paragraph describing the steps and the tools you used to carry out the attacks; (7) a text file named `attack-7.txt` that contains the maximum number of years in jail that you could face under 18 USC § 2511 for intercepting traffic on an encrypted WiFi network without permission.

Part 3. Anomaly Detection

In Part 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 `detector.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.

Part 4. Bonus: Analyzing a suspicious PCAP file

(a) `04:11 *devlol* The TTT is attacking our IRC server. While we wait for a bite on this phising attack, analyze the network logs so I can reconfigure the firewall to keep them out.`

We have provided you the PCAP file, `p4-1.pcap`, at https://subversion.ews.illinois.edu/svn/sp15-cs461/_shared/mp3/p4-1.pcap. This file consist of some logs we captured. You'll need to analyze the log and give us the following information:

- IP address of the infected machine.
- MAC address of the infected machine.
- Domain name of the compromised website.

What to submit Submit a file named `bonus-1.txt` where the first, second, and third lines list, respectively, the IP address, the MAC address, and the domain name.

(b) `06:42 *devlol* Fish on! Our spear phishing attack worked. You know what? I've got network logs here from around the time the TTT compromised my system. I bet their exploit payload is in there somewhere. If you can find it, I'll launch it against their server. Let's give these jerks a taste of their own medicine!`

We have provided a PCAP log file, `p4-2.pcap`, at https://subversion.ews.illinois.edu/svn/sp15-cs461/_shared/mp3/p4-2.pcap. There are some malicious activity in the log file. We are specifically asking for the name of three files (payloads), one with *.dll and the two other with *.module.

What to submit Submit a file named `bonus-2.txt` with the full name of all three files (including the file extension). Put each name on a separate line.