Lab 5: Network Security

Submission policy. This lab is due on **Tuesday, April 18, 2017** at **11:59PM** via websubmit, following the submission checklist below. Late submissions will be penalized according to course policy. Your writeup MUST include the following information:

1. List of collaborators (on all parts of the project, not just the writeup)
2. List of references used (online material, course nodes, textbooks, wikipedia, etc.)
3. Number of late days used on this assignment
4. Total number of late days used thus far in the entire semester

If any of this information is missing, at least 20% of the points for the assignment will automatically be deducted from your assignment. See also discussion on plagiarism and the collaboration policy on the course syllabus.

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.

Grading

Each part of this assignment is worth 50 points for a total of 150 points. You may choose to entirely ignore one part (or do segments of all three of them) and still get a full score of 100 points. However, if you do get more than 100 points, the additional score will count as a bonus.

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 the "Ethics" section on the course website.

Part 1. Network Attacks

In this part of the project, you will experiment with network attacks by cracking the password for a WEP-encrypted WiFi network, performing a man-in-the-middle attack on an HTTPS connection made over that network, and recovering a simulated victim's username and password.

In **MCS B20** (all students registered for the course will have swipe access starting April 7th), there is WiFi network named "glaDOS" that is protected with WEP, a common but insecure crypto protocol. We've created this network specifically for you to attack as described below, and you have permission to do so. There is also a client wirelessly connected to this network that makes a connection to a password-protected HTTPS website every few seconds. The client is configured to ignore certificate errors (including self-signed certs), simulating a user who habitually clicks through browser security warnings. Your goal is to recover the username and password for this site.

Most of the online tutorials and tools for this project assume you are running linux and that you have a wifi card which can be run in monitor mode. We have setup computers with wifi cards in MCS B20 that you will be able to live boot in KALI linux. **Instructions for booting into Kali linux will be provided during discussion on April 7th. It is vital that you attend that discussion section.** Note that while you are allowed to do this on your own computer, it is far easier to use the machines in the lab. Most newer computers do not support putting their wifi card in monitoring mode.

If you would rather use your own machine, you will need to boot KALI linux off a USB thumb drive (http://www.kali.org/). Here is a video and set of links to do this: http://www.irongeek.com/i.php?page=videos/kali-linux-live-boot-usb-flash-drive-jeremy-druin-webpwnized. Here is a guide for connecting to a wireless network from the linux commandline http://puppylinux.org/wikka/HowToWifiFromCommandWEP.

First, you will need to attack the wireless network in order to find the WEP encryption key and join the network. There are many online tutorials and automated tools available to help you perform this task. We recommend Aircrack-ng, available at http://www.aircrack-ng.org/. Here is a decent tutorial to attacking WEP protected wireless networks http://www.aircrack-ng.org/doku.php?id=

simple_wep_crack. Once you crack the WEP key you should be able to log into non-password protected page for the HTTPS server.

Once connected to the network, you will need to use ARP spoofing to redirect the client's traffic through a machine you control, then perform a man-in-the-middle attack on the HTTPS connection. Again, there are a variety of tutorials and tools available online. For ARP spoofing we recommend dsniff http://en.wikipedia.org/wiki/DSniff which provides a utility arpspoof. Multiple people ARP snoofing at the same time may cause problems, if it does so, take turns.
We recommend sslsniff, available at http://www.thoughtcrime.org/software/sslsniff/. If you use sslsniff you will need to generate a bogus certificate or download one (https://github.com/freelan-developers/freelan-all/wiki/Sample-certificate-files). Note that the username and password will be base64 encoded, so if you don't see anything in the log file that sslsniff creates, that appears to be a username:password, look for a base64 encoded string. You should check that you have the right password by attempting to log in to the password protected page on HTTPS server.

We've intentionally provided few details about how to accomplish these goals. However, you should be able to find abundant information by searching for the tools and attacks. As always, we recommend starting early!

What to submit The pdf file you submit must contain the following:

(1) a paragraph describing how you setup your computer to perform the attack (5 points);

(2) the WEP key for the network (10 points);

(3) the password for the HTTPS site the client loads (10 points);

(4) the secret code on the HTTPS site once you log in (10 points);

(5) a paragraph describing the steps and the tools you used to carry out the attacks (10 points);

(6) the maximum jail time you could face under 18 USC § 2511 for intercepting traffic on an encrypted WiFi network without permission (5 points).

Part 2. 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).

Download the network trace at http://www.cs.bu.edu/~goldbe/teaching/CS558S17/lab5/lab5-2.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 sitting at the local network are sending packets. What are their MAC and IP addresses?

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.

3. 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 questions. Your answers should include references by number to corresponding Wireshark frames.

   (a) What is the domain name of the site the client is connecting to?

   (b) Is there any way the HTTPS server can protect against the leak of information in (a), namely the domain name of the site the client was connecting to?

   (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.

   (d) Based on what you have been taught and any other information you can find, are any of these cipher suites worrisome from a security or privacy perspective? Why?

   (e) What cipher suite does the server choose for the connection?

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

   (a) Even though logins are processed over HTTPS, what else is insecure about the way the browser is authenticated to Facebook?

   (b) How would this let an attacker impersonate the user on Facebook?

   (c) How can users protect themselves in general against this type of attack?

   (d) What did the user do while on the Facebook site?

What to submit The pdf file you submit must contain your answer for all of the above questions.

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 identify *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. For more information on port scanning see here http://en.wikipedia.org/wiki/Port_scanner.

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 using the `dpkt` packet manipulation library in order to detect possible `SYN` scans. The `dpkt` library is available in most package repositories or at https://code.google.com/p/dpkt/. You can find 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/.

Your program will take one command-line parameter, the path 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 at least 3 times more `SYN` packets than than 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 http://www.cs.bu.edu/~goldbe/teaching/CS558S17/lab5/lab5-3.pcap. (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` 1.6 is 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.

Submission Checklist

Upload to websubmit an archive file (`tar/.tar.gz/.zip/.rar`) named `yourname.lab4.extension`. The archive should contain **only** the following files:

- A pdf file named `yourname.lab4.pdf` with your answers for parts 1 & 2. Do not forget to include your name, late days, references and collaborators.
- A python module for scan detection named `detector.py` for part 3. Include your name in the comments.