Project 3: Network Security

This project is due on **Thursday, October 24** 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 20.5 hours past the deadline. If you have a conflict due to travel, interviews, etc., please plan accordingly and turn in your project early.

This is a group project; you will work in **teams of two** and submit one project per team. Please find a partner as soon as possible. If have trouble forming a team, post to Piazza's partner search forum.

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.

Solutions must be submitted electronically via CTools, following the submission checklist below. Please coordinate carefully with your partner to make sure at least one of you submits on time.

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).

Download the network trace at https://www.eecs.umich.edu/courses/eecs388/static/proj2-1.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?
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. One of the clients connects to an FTP server during the trace.
   (a) What is the DNS hostname of the server it connects to?
   (b) Is the connection using Active or Passive FTP?
   (c) Based on the packet capture, what's one major vulnerability of the FTP protocol?
   (d) Name at least two network protocols that can be used in place of FTP to provide secure file transfer.
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?
   (b) Is there any way the HTTPS server can protect against the leak of information in (a)?
   (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) 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?
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?
   (b) How would this let an attacker impersonate the user on Facebook?
   (c) How can users protect themselves against this type of attack?
   (d) What did the user do while on the Facebook site?

What to submit Submit a text file named `pcap.txt` containing your answers.

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, 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 the atrium of the Beyster building, there is WiFi network named "eecs388" 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.

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/.

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. We recommend sslsniff, available at http://www.thoughtcrime.org/software/sslsniff/.

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 Submit a text file named `attack.txt` that contains the following: (1) the WEP key for the network; (2) the password for the HTTPS site the client loads; (3) a paragraph describing the steps and the tools you used to carry out the attacks; (4) the maximum jail time 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 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 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` 1.6 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 CTools a gzipped tarball (`.tar.gz`) named `project3.`*`uniqname1`*`.`*`uniqname2`*`.tar.gz`. The tarball should contain only the files below:

Part 1: Exploring Network Traces

`pcap.txt`	A plain text file containing your answers to the questions in Part 1.

Part 2: Network Attacks

`attack.txt`	A plain text file with the contents specified in Part 2.

Part 3: Anomaly Detection

`detector.py`	Your Python program for `SYN` scan detection.