Project 3: Network Security

This project is due on **Friday, November 1, 2024** at **11:59 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 24 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 you have trouble forming a team, post to Piazza's partner search forum. The final will cover project material, so you and your partner should collaborate on each part.

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 Canvas, following the submission checklist below.

Introduction

This project will introduce you to common network protocols, to network packet trace analysis, and to the basics of network penetration testing.

Objectives

- Gain exposure to core network protocols and concepts.
- Learn to apply manual and automated traffic analysis to detect security problems.
- Understand offensive techniques used to attack local network traffic.
- Practice network penetration testing.

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!** Per course 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 network packet trace (commonly called a "pcap") that we recorded on 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, which is available at https://www.wireshark.org.

Download the pcap from http://courses.cs.vt.edu/cs4264/static/project4/proj4.pcap, and examine it using Wireshark. Familiarize yourself with Wireshark's features. and try exploring the various options for filtering and for reconstructing data streams.

Concisely answer the questions below. Each response should require at most 2–3 sentences.

1. Multiple devices are connected to 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 containing your answers. Make sure each answer is formatted as a single line. Format your file using this template:

```
# Question 1

1. [Answer ...]

# Question 2

2. [Answer ...]

# Question 3

3a. [Answer ...]

3b. [Answer ...]

3c. [Answer ...]

3d. [Answer ...]

# Question 4

4a. [Answer ...]

4b. [Answer ...]

4c. [Answer ...]

4d. [Answer ...]

4e. [Answer ...]

# Question 5

5a. [Answer ...]

5b. [Answer ...]

5c. [Answer ...]

5d. [Answer ...]
```

Part 2. Anomaly Detection

In Part 1, you manually explored a network trace. Now, you will programmatically analyze a pcap file 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. In one kind of 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. To do this, you will use `dpkt`, a library for packet manipulation and dissection. It is available in most package repositories. You can find more information about `dpkt` at https://github.com/kbandla/dpkt and view documentation by running `pydoc dpkt`, `pydoc dpkt.ip`, etc.; there's also a helpful tutorial here: https://jon.oberheide.org/blog/2008/10/15/dpkt-tutorial-2-parsing-a-pcap-file/.

Your program will take the path of the pcap file to be analyzed as a command-line parameter, e.g.:

```
python3 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 large 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 3 program that accomplishes the task specified above, as a file named `detector.py`. You should assume that `dpkt` and `scapy` 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.

Submission Checklist

Upload to Canvas a gzipped tar file (`.tar.gz`) named `project4.`*`pid1`*`.`*`pid2`*`.tar.gz` that contains only the files listed below. Make sure you have the proper filenames and behaviors. You can generate the tarball at the shell using this command:

```
tar -zcf project4.pid1.pid2.tar.gz report.txt detector.py
```

Part 1: Exploring Network Traces

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

Part 2: Anomaly Detection

`detector.py`	Your plain text Python 3 program for `SYN` scan detection.