Project 3: Network Security

This project is due on **Thursday, March 16** at **6 p.m.** and counts for 8% of your course grade. 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 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 Moodle, following the submission checklist below.

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 (https://www.wireshark.org).

Download the network trace at https://ecen5032.org/static/proj3.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 a telnet server during the trace.
   (a) What is the DNS hostname of the server it connects to?
   (b) Based on the packet capture, what's one major vulnerability of the telnet protocol?
   (c) What is a command run on the telnet server?
   (d) Name a network protocol that can be used in place of telnet to provide secure remote login.
4. The trace shows that at least one of the clients makes HTTPS connections. 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 is trying to use a search engine.
   (a) Name the domain of the first search engine used.
   (b) What is insecure about the search engine?
   (c) Name something the user searched for
   (d) How can users protect themselves against this type of attack?
6. What is 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?

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 man-in-the-middling an HTTP connection to a website we control, and replacing some of its content.

We have set up the website http://freeaeskey.xyz/, which is a website that provides "random" AES-256 keys for free to anyone who visits. Professor Vuln has decided to use this website for encrypting his research. To do this, he has created a program that first fetches a fresh key from freeaeskey.xyz, and uses it to encrypt the private data. Your goal is to get Professor Vuln to encrypt the secret research under a key known to you.
To do this, you are able to get a program to run on Professor Vuln's network. For the purposes of this assignment, you can assume your program runs as root on the same machine that Professor Vuln uses to download his key. Your task is to write a Python program that watches for requests to freeaeskey.xyz, and replaces the key provided with one known to you:

`49276d20737475636b20696e20616e20414553206b657920666163746f727921`

The rest of the web page should remain un-modified to avoid suspicion.
Your script will run as root, and any other users on the same machine that visit freeaeskey.xyz while it is running should receive this injected key. We will grade this project using an Ubuntu 16.04 machine (i.e. your attack should assume a Linux networking stack).

You are welcome to use any of the following libraries. If you believe you need additional ones, please ask on Piazza.
https://pypi.python.org/pypi/dpkt
https://pypi.python.org/pypi/dnet
https://pypi.python.org/pypi/scapy

Note: we strongly recommend you inject IP-level packets using the following code:

```
def inject_pkt(pkt):
    import dnet
    dnet.ip().send(pkt)
```

Bonus: Attack HTTPS [Extra credit]
Professor Vuln has realized it is unwise to download keys over HTTP, and has switched to using HTTPS to download his keys, from https://freeaeskey.xyz. Make a new script (`attack_https.py` that caries out the same attack as before against HTTPS. (Hint: what is unique about the certificate served from freeaeskey.xyz?)

What to submit Submit a Python script named `attack.py` that performs the attack when run as root on the local machine. For the (optional) bonus, submit `attack_https.py` as well.

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. 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://dpkt.readthedocs.io/en/latest/ 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/. 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 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 Moodle a gzipped tarball (`.tar.gz`) named `project3.`*`identikey1`*`.`*`identikey2`*`.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.py`	A Python script that caries out the attack specified in Part 2.
`attack_https.py*`	A Python script that does the HTTPS attack (extra credit)

Part 3: Anomaly Detection

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