Total points: 80

Assignment 3

Due: 11:59 PM, 4th Aug (Fri)

Written assignment

1. [14 points] Visit `metrics.torproject.org` to answer the following questions about Tor:
   (a) [4 points] Give the total amount of advertised bandwidth of relays with the “Guard” flag (but not the “Exit” flag) and relays with the “Exit” flag (but not the “Guard” flag) on 2020-02-01. Which is more? Give one reason to explain this phenomenon.

   (b) [5 points] Give the median download rate of a file (in bytes per second) for a 50 KiB file and a 5 MiB file to the `op-hk` **onion** server on 2020-02-01. Both files are loaded over 1 round trip. Assuming those two file loads have the same latency and transfer rate, calculate that latency and transfer rate. Then, calculate what percentage of the total load time is due to latency, and what percentage is due to transfer rate. (Those two numbers should add up to 1.)

   (c) [3 points] How long would it take to load a 50 KiB file using 2 nodes instead of 3 nodes? For simplicity, assume that all nodes have the same transfer rate and all connections (between nodes, or between nodes and users/servers) have the same latency.

   (d) [2 points] How does the use of 4 nodes instead of 3 nodes affect Tor’s performance and privacy?

2. We have four data privacy techniques:

    1. k-anonymity
    2. Differential privacy
    3. Secure multiparty computation (SMPC)
    4. Private information retrieval (PIR)

    For each scenario below, two of the four data privacy techniques will be proposed to resolve the challenge. Choose the correct one, explain why it is suitable, and why the other choice is not suitable.

    (a) [4 points] A failing streaming entertainment company wants to revitalize its business by improving its recommendation algorithm. The recommendation algorithm takes in private data such as user location, viewing hours, and demographic information, and outputs recommendations of what shows to watch. The company will run a public contest and select the best algorithm. To run the contest, they need to give contestants access to millions of entries of private data, which is a violation of privacy if not done with a privacy-preserving method. Proposals: k-anonymity, SMPC.

    (b) [4 points] You want to buy a new smart device that encourages a healthy lifestyle by monitoring your daily exercise. The device needs to track your movement on a map to know how much calories you are actually expending (e.g. hiking and swimming is different from walking). However, you consider this to be a privacy risk: you do not want the app to know where you are at all times. A new company making these smart devices is willing to use a data privacy technique to protect your privacy. Proposals: differential privacy, PIR.

    (c) [4 points] You would like to purchase a new web domain. However, you are aware of the practice of cybersquatting; people may purchase the domain first if they know it is in demand, and sell it to you at an elevated price. You want to know if the domain is still available, but you are worried that attempting a DNS query for the domain will lead to some DNS servers purchasing it immediately for cybersquatting. To assure potential customers that it is not malicious, a DNS server is willing to cooperate with you and implement a privacy-preserving algorithm. Proposals: k-anonymity, PIR.

    (d) [4 points] People are anxious to know if they are living in the same apartment as someone infected with the infectious CROW disease. The hospital that holds all this information is willing to run a privacy-preserving algorithm with worried potential infectees to check if someone living in their apartment has been infected. Proposals: differential privacy, SMPC.

Programming assignment

Firewall [50 points]

In this assignment, you are asked to write an IPv4 packet-filtering firewall for the subnet 142.58.22.0/24. It examines each packet to decide whether or not to filter it out (drop it).

The input to your program is a tcpdump-like file containing IP packet data dumps. An example, packets.txt, has been provided. It is simplified to remove packet header information, which is contained in the packet data. Each packet starts with a packet number on its own line, then the packet data in the following tabbed lines. A packet data line has up to 16 bytes of data. It starts with a hexadecimal byte count, then the actual data, also written in hexadecimals.

For example, a typical packet in the assignment may start with:

```
4500 00a3
```

This is four bytes:

- The first hexadecimal (first half-byte) is 4, indicating IPv4.
- The second hexadecimal is the header length, which is a minimum of 5 and always 5 in this assignment, as optional headers are not used.
- The third and fourth hexadecimals (second byte) are services that are not used in this assignment, thus set to 0.
- The fifth to eight hexadecimals are the packet length in bytes, including the header. 00a3 is equal to 163, so this packet has 163 bytes.

To do this assignment, you will need to understand how to read IP, ICMP and TCP headers. These headers will be referenced in the assignment, and you can read more about their positions and functionality in a variety of sources including Wikipedia and RFCs.

Your program should be called filter and will be called with (python3 example):

```
python3 filter.py <option> <filename>
```

where:

- option is either -i, -j, -k, corresponding to three subparts of the assignment
- filename is the packet dump file in the same format as packets.txt.

In standard output, write the result of whether or not each packet is dropped. Each line should correspond to one packet, and should start with the number, followed by a space, followed by either "yes" (it is a malicious packet and should be dropped) or "no" (it is not a malicious packet and should be allowed to pass). For example, `python3 filter.py -i packets.txt` should output:

```
1 yes
2 yes
3 no
4 yes
5 no
6 yes
7 yes
```

Do not write anything else to standard output.

Your packet filter should not be overzealous. That is, if the question does not ask you to filter out a certain packet, then you should not do so. To simplify the assignment, you do not need to deal with re-assembly or re-ordering of packets within this assignment. The header checksums should also be ignored. The sequence numbers are always correct. The problems are not cumulative: for examples, the answer to (b) should not filter out violating packets of (a).

(a) [10 points] Egress filtering (-i)

Write an egress packet filter that filters out all packets with source and/or destination IPs that do not make sense based on your firewall's location. For this question, your firewall expects to only see **outgoing packets** from the subnet to outside the subnet. Other packets are attack packets. For other questions, your firewall should expect to see both incoming and outgoing packets.

(b) [20 points] Two ping-based attacks (-j)

Write a packet filter that filters out both pings of death and smurf attacks targeting hosts in your subnet. Both of these are pings, more formally known as ICMP echo packets. To prevent pings of death, study the slides and implement a filter that would cause a buffer overflow described in the slides. To prevent smurf attacks, drop the ping packets sent to the network broadcast address that would cause a host in your subnet to be flooded.

Hints:

- An ICMP echo is defined on RFC 792 page 13. It is **not** the same as an echo reply.
- There is a protocol field in the IP header that indicates whether or not the packet is an ICMP message.
- You may assume that the IP length field is always correct.
- Only pings should be dropped in this question.
- Only attacks targeting hosts in your subnet should be dropped. Ignore other packets.

(c) [20 points] SYN floods (-k)

Write a packet filter that filters out SYN floods. Specifically, if a single outside IP attempts to half-open more than 10 connections with a host in your subnet, then filter out all further SYN packets until the total number of half-open connections is reduced below 10. A count should be maintained, such that an outside IP can reach 10 half-open connections, go below 10 half-open connections, but never go above 10. TCP SYNs as well as other TCP packets relevant to this part are described in RFC 793.

Hints:

- Once the SYN-ACK has been ACK'd, the connection is fully open, not half-open. Fully open connections should not count towards the limit of 10 half-open connections.
- A connection, whether half-open or fully open, can be killed by a RST or a FIN from either side. You should take both into account.
- Only one TCP connection, whether half-open or fully open, can exist at a time between two TCP ports. Any attempt to open a new connection between two TCP ports while a connection already exists will be completely ignored by the application without affecting the current connection.
- You need to allow multiple IPs to maintain up to 10 half-open connections each — they do not interfere with each other.

Packet examples

packets.txt contains examples of the following packets.

- Packet 1 is a ping from 1.2.3.4 to 142.58.22.107.
- Packet 2 is a ping of death from 1.2.3.4 to 142.58.22.107. Note the large offset.
- Packets 3 to 5 are a regular TCP handshake from 142.58.22.107 to the HTTPS port of 1.2.3.4: SYN, SYN-ACK, and ACK.
- Packet 6 is a TCP FIN from 1.2.3.4 to 142.58.22.107. This would kill the connection, whether it is fully open or half open.
- Packet 7 is a TCP RST from 1.2.3.4 to 142.58.22.107. This would also kill the connection.

Submission instructions

All submissions should be done through CourSys. Submit the following programs:

- `a3.pdf`, containing all your written answers.
- Code for the programming assignment, detailed below:

For the programming assignment, submit your code; do not submit any compiled files.

C++: Submit filter.cpp. I will compile them and call `./filter <option> <testfile>`.

Python: Submit filter.py. I will call `python3 filter.py <option> <testfile>`.

Java: Submit filter.java. I will compile with `javac filter.java` and then call `java filter <option> <testfile>`.

If there is a Makefile in your folder, the Makefile will override all of the above. This implies if you are not writing in C++, Python, or Java, you must include a Makefile.

Keep in mind that plagiarism is a serious academic offense; you may discuss the assignment, but write your assignment alone and do not show or send anyone your answers and code.