Total points: 80 (+0 bonus)

Assignment 3

Due: 11:59 PM, 12th Aug (Fri)

Written assignment

1. [17 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) [4 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. Can you explain the difference?
   (c) [5 points] The download times measured in the above were done on circuits with three nodes, randomly chosen across the globe. Based on these figures, estimate the median download rate for a 50 KiB file if Tor used only one node. You may need to make certain assumptions; write them down.
   (d) [4 points] Determine the growth (as a ratio) in the bandwidth spent on answering directory requests between 2017-01-01 to 2018-01-01. A directory request is a list of all Tor nodes sent by directories when a user connects to Tor. What is the main cause of this growth? Use figures to support your conclusion.

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 entires 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

Error Correcting Codes [47 points]

Error Correcting Codes (ECCs) can be used in unsafe transmission or storage devices to improve reliability by providing redundancy. Unlike checksums, ECCs can automatically correct errors up to a certain number of bits. In this question, you will be asked to implement the general Hamming code, which can correct any 1-bit error in a general string. (Do not use a Hamming code library — it should be your own code.) The specifications of the Hamming code are as follows; it has been slightly modified to discourage code sharing from other Hamming code implementations.

1. Write the input data string as a bitstring according to ASCII.
2. Choose enough parity bits and intersperse the data bits with parity bits, so that the i-th parity bit p_i is followed by exactly $2^{i-1} - 1$ data bits. Denote the i-th bit of H as H_i; the first bit is H_1. Denote the length of H as $|H|$.
3. Let $B(x)$ be the bitstring representation of the integer x, for example, $B(5) = 00101$.
4. Define each M_i to be a set of specific bit positions in H, such that M_i is the set of integers $1 \leq x \leq |H|$ where the i-th least significant bit of $B(x)$ is 1.
5. Set the parity bit p_i to be the XOR of all the bits of H in the positions indicated by M_i, except p_i itself, which is always located at $H_{2^{i-1}}$:

$$p_i = \bigoplus_{x \in M_i \setminus \{2^{i-1}\}} H_x$$

Here is a more detailed explanation of M_i. Consider the integer $x = 10$, so $B(10)$ is 1010. Then the 1st least significant bit is 0, the 2nd is 1, the 3rd is 0, and the 4th is 1 (reverse the bitstring). So x belongs to M_2 and M_4, not to M_1 and M_3.

To work out an example:

1. The input data string is "ab", which becomes 0110000101100010 (16 data bits).
2. We need five parity bits, p_1, p_2, p_3, p_4, p_5, as follows:

$$H = p_1 p_2 0 p_3 110 p_4 0001011 p_5 00010$$

   p_5 can be followed by up to 15 data bits, but we ran out of data bits, so we stopped there. $H_3 = 0$ and $H_8 = p_4$. $|H| = 21$.
3. For example, $B(20) = 10100$.
4.

$$\begin{aligned} M_1 &= \{1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21\} \\ M_2 &= \{2, 3, 6, 7, 10, 11, 14, 15, 18, 19\} \\ M_3 &= \{4, 5, 6, 7, 12, 13, 14, 15, 20, 21\} \\ M_4 &= \{8, 9, 10, 11, 12, 13, 14, 15\} \\ M_5 &= \{16, 17, 18, 19, 20, 21\} \end{aligned}$$

5.

$$
\begin{aligned}
p_1 &= H_3 \oplus H_5 \oplus H_7 \oplus H_9 \oplus H_{11} \oplus H_{13} \oplus H_{15} \oplus H_{17} \oplus H_{19} \oplus H_{21} &= 0 \\
p_2 &= H_3 \oplus H_6 \oplus H_7 \oplus \ldots &= 1 \\
p_3 &= H_5 \oplus H_6 \oplus H_7 \oplus \ldots &= 0 \\
p_4 &= \ldots &= 1 \\
p_5 &= \ldots &= 1
\end{aligned}
$$

(a) [20 points] Write a program that will automatically generate the correct parity bits for any input string. The input string will be the first argument of the program. Your program, called **a3a**, should write the parity bits in the correct order to `stdout` (e.g. cout or print) as **characters**, "0" or "1". Continuing the above example, if the program is called with:

```
./a3a ab
```

The program should display:

```
01011
```

(b) [20 points] Write a program that will automatically correct any one-bit error in either the parity or the data string. The parity will be the first argument of the program, and the data string will be the second argument of the program. Your program, called **a3b**, should output the *corrected* parity bits to the first line of `stdout`, and the *corrected* data string to the second line of `stdout`. If there is no correction to be made, you should still output the original parity bits and the data string. For example, if I run your program with:

```
./a3b 01011 ac
```

Your output should be:

```
01011

ab
```

The program will always be tested with the correct number of parity bits, and there will either be a one-bit error in the parity or the data string, or no error. Either type of error should be fixed. Also, note the meaning of an ASCII one-bit error: "a" (01100001) can become "i" (01101001), but "a" cannot become "b" (01100010) because that would be two bit flips.

(c) [7 points] Submit two sentences in files `sentence1` and `sentence2`, such that they have the same Hamming code (as output by part a). They should all be logical and grammatically correct sentences with no typos, and they should all include your @sfu.ca SID. Any ASCII characters that are not letters, numbers, or basic punctuation will be removed.

Hint: It may be easier to construct these sentences if you use numbers somewhere in your sentence.

Submission instructions

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

- `a3.pdf`, containing all your written answers.
- Files for the programming assignment: `sentence1`, `sentence2`.
- Code for the programming assignment, detailed below:

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

C++: Submit a3a.cpp, a3b.cpp. I will compile them and call `./a3a <message>`.

Python: Submit a3a.py, a3b.py. I will call `python a3a.py <message>`.

Java: Submit a3a.java, a3b.java. I will compile with `javac a3a.java` and then call `java a3a <message>`.

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.

The submission system will be closed exactly 48 hours after the due date of the assignment. Submissions after then will not be accepted unless you have requested an extension before the due date of the assignment. You will receive no marks if there is no submission within 48 hours after the due date.