18734 Homework 5

Due: 12 Noon Eastern, 9 AM Pacific, Nov 30

Problem 1: Implementing Laplace mechanism (30 marks)

In this problem you will implement Laplace mechanism to provide differential privacy for the provided database. The first part that you need to think about is how to sample from a Laplace distribution. In this problem you are free to use C++, Java or Python. You will need to figure out how to generate random real numbers between $[0, 1]$. (e.g., for Java you can see `http://java.about.com/od/javautil/a/randomnumbers2.htm`)

1. Assume you have access to a sampler that samples from the uniform distribution over $[0, 1]$. Read about inverse transform sampling `http://en.wikipedia.org/wiki/Inverse_transform_sampling`.
   (a) Write the CDF function F for Laplace distribution with mean 0 and variance $2\lambda^2$.
   (b) Use inverse transform sampling and the uniform distribution sampler to write code to sample from a Laplace distribution with mean zero and variance $2\lambda^2$. (Write a function where the variance is given as an argument).
2. Download the scores.txt file[1]. It has the scores of 10,000 students with scores between 1 and 100. You are asked to provide differential privacy for the query `average`.
   (a) State the global sensitivity of the query.
   (b) Using your implementation of sampling from Laplace distribution, write code to provide 0.001-differential privacy for the above query.
   (c) Repeat the following 100 times (not manually, but, in the code)
   In each iteration i repeat the query n_i times till the average of the n_i answers is 10^{-4} close to the true average.
   Report the number $\frac{\sum_i n_i}{100}$.

Note that the number $\frac{\sum_i n_i}{100}$ is an estimate of the number of times you can query before you reveal the true average with error of less than 10^{-4}.

[1] `https://www.ece.cmu.edu/~ece734/homeworks/scores.txt`

Problem 2: Anonymous Communication (35 marks)

General Protocol (15 marks)

In the lecture, we discussed the Dining Cryptographers protocol. In this problem, we will explore how to use that protocol as a building block to construct a general protocol for anonymous communication. Consider a group of n agents. (You may want to read this `http://users.ece.cmu.edu/~adrian/731-sp04/readings/dcnets.html`)

1. Describe a protocol using which one of the n agents can send an m-bit message. Explain informally why the protocol is *correct* (i.e., all agents receive exactly the message that was sent) and *anonymous* (i.e., none of the other agents have any clue who the real sender is).
2. State and prove rigorously that anonymity is preserved by the protocol for the case where $n = 4$ and $m = 1$. (You need to show that from the point of view of any non-sender, the probability of any of the other agents being the sender is 1/3).
3. How many bits of randomness and how many message transmissions are needed to complete this protocol with n agents and an m-bit message?
4. How robust is this protocol to collusion, i.e., if k out of the n non-sender agents collude, what is the probability that they can figure out who the real sender is?

Hidden services (5 marks)

Tor can also provide anonymity for servers, apart from providing anonymity for clients. Read the relevant part of the paper `https://svn.torproject.org/svn/projects/design-paper/tor-design.pdf` and explain how hidden services work in Tor. The explanation **must be a bulleted list of the main points**. Marks will be deducted for writing paragraphs.

Nymble (15 marks)

Tor can sometimes lead to some undesirable consequences. This problem asks you to look at the paper `http://www.cs.dartmouth.edu/~sws/pubs/jkts07.pdf` and answer the following questions:

1. What potential problem with Tor are identified in the paper?
2. Provide an overview of how the Nymble system works. Section 3 in the paper has such an overview. You can read that overview, however, your answer must be in your own words.
3. List the (informally) cryptographic properties that the Nymble system relies on.

Problem 3: Zero knowledge proofs (20 marks)

Amplification (10 marks)

This problem gives you the essence of amplification: going from a small difference in the completeness and soundness probabilities for interactive proofs to almost 1 difference in these probabilities. As mentioned in class, this involves repetition of the interactive protocol. Below, we ask a question on probability that demonstrates amplification. Suppose there are two types of coins: one made with bronze and one with gold. Your task is to figure out if a given coin in made of bronze (B) or gold (G). You also know that bronze coins produce heads with probability $\frac{1}{2} + \delta$ and gold coins produce heads with probability $\frac{1}{2} - \delta$.

You come with the idea that if you flip the coin k times and take the majority vote, then an answer of head indicates a bronze coin and an answer of tails indicates gold coin (with high probability). To be absolutely sure that the idea is right lets do the following computation.

1. You want to bound the probability of committing a mistake. Suppose the true coin type is X (which is either B or G). You make a mistake when your final answer is not X. And your final answer if not X when the majority of coin flips indicate tails when $X = B$ or indicate heads when $X = G$. But, note that in either case (B or G) the side of the coin that showed up in majority has a probability $\frac{1}{2} - \delta$ of occurring. Thus, we get the following

$$P(mistake) = P\left(\geq k/2 \text{ events with probability of each event } \frac{1}{2} - \delta\right)$$

2. Next, note that the right hand side of the above equation can be written as

$$P\left(\bigcup_{i=0}^{k/2} k/2 + i \text{ events with probability of each event } \frac{1}{2} - \delta\right)$$

3. Using union bounds (`http://en.wikipedia.org/wiki/Boole's_inequality`) show that the above value is bounded by

$$\sum_{i=0}^{k/2} P\left(k/2 + i \text{ events with probability of each event } \frac{1}{2} - \delta\right)$$

4. Show that

$$P\left(k/2 + i \text{ events with probability of each event } \frac{1}{2} - \delta\right) = \binom{k}{k/2+i}(\frac{1}{2} - \delta)^{k/2+i}(\frac{1}{2} + \delta)^{k/2-i}$$

5. Argue that the above value in 4 is less than $\binom{k}{k/2+i}(\frac{1}{2} - \delta)^{k/2}(\frac{1}{2} + \delta)^{k/2}$

6. Using the above bound in 5, argue that sum in 3 is less than $2^k(\frac{1}{2} - \delta)^{k/2}(\frac{1}{2} + \delta)^{k/2}$, which can easily be reduced to $(1 - 4\delta^2)^{k/2}$. Thus, the probability of mistake is bounded by $(1 - 4\delta^2)^{k/2}$.

Now, clearly by choosing many repetitions k, the probability of mistake becomes very small for any constant $\delta > 0$.

Simulation (10 marks)

This problem asks you to argue informally why the distribution generated by the prover and verifier (denoted as $\langle P, V \rangle$) is computationally indistinguishable from the distribution generated by the simulator (denoted as $\langle M \rangle$).

The simulation is shown in Figure 1. Given $\langle N, Y, g\rangle$, P claims to know s, such that $Y = g^s \mod N$. s is known as the discrete logarithm of Y given $\langle N, g\rangle$ and finding it is known to a be difficult problem given Y, N, g [2].

1. P picks a random r. Then she sends $Y = (g^s \mod N)$ and $A = (g^r \mod N)$ to V.
2. V picks a random challenge c and sends it to P.
3. P then sends over $z = r + cs \mod (N-1)$.

V accepts the proof iff $AY^c = g^z \mod N$.

Note that the distribution $\langle P, V\rangle$ has four (single dimensional) random variables corresponding to the four values that are exchanged in the interaction. You have to argue separately for each random variable, why the distribution of that random variable in $\langle P, V\rangle$ is roughly same as the distribution of the corresponding random variable in $\langle M\rangle$.

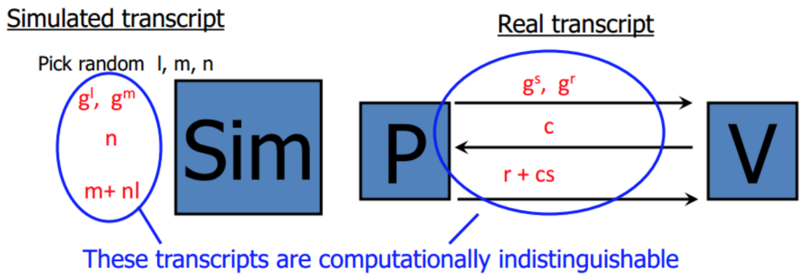

Figure 1: Simulation.

Problem 4: Oblivious Transfer (15 marks)

Consider a simplified version of the oblivious transfer protocol proposed by Even, Goldreich and Lempel. Let R be the receiver with input $b \in \{0, 1\}$ and let S be the sender with input $x_0, x_1 \in \mathcal{X}$. Assume both S and R have access to an oracle $\mathcal{O}_{map} : \mathcal{X} \to \mathcal{X}$, which returns an element in $\mathcal{X}$ as well as an inversion oracle such that

$$\forall x \in \mathcal{X} : \mathcal{O}_{inv}(\mathcal{O}_{map}(x)) = x$$

To transfer x_b from the sender to the receiver, they carry out the following steps:

1. R sends c_0, c_1 to S, where $c_b \leftarrow \mathcal{O}_{map}(r)$ for $r \overset{\$}{\leftarrow} \mathcal{X}$ and $c_{1-b} \overset{\$}{\leftarrow} \mathcal{X}$, where $a \overset{\$}{\leftarrow} \mathcal{X}$ indicates that a is randomly selected from $\mathcal{X}$.
2. S replies with d_0, d_1, where $d_0 \leftarrow \mathcal{O}_{inv}(c_0) \oplus x_0$ and $d_1 \leftarrow \mathcal{O}_{inv}(c_1) \oplus x_1$.
3. R reconstructs $x_b \leftarrow r \oplus d_b$.

[2] `http://en.wikipedia.org/wiki/Discrete_logarithm`

Correctness (5 marks)

For the above scheme, show that the receiver indeed recovers x_b.

Obliviousness (5 marks)

Give an informal argument for why the sender cannot determine the b at the end of the protocol.

Extending OT (5 marks)

Suppose you have access to a device that can perform 1-out-of-2 oblivious transfer. How would you use this device to create a device that can perform 1-out-of-3 oblivious transfer? You may have to use the device more than once.

Submission

You have to submit two files:

1. Merge all the written parts into a single pdf file ⟨your_andrew_id⟩_HW5.pdf.
2. Rename the program file (.c/.cpp/.java/.py) you used for Problem1 as ⟨your_andrew_id⟩_laplace.⟨extension⟩.

Zip these files into ⟨your_andrew_id⟩_HW5.zip and submit the zip file on BlackBoard.