Massachusetts Institute of Technology
6.857: Network and Computer Security
Professors Ronald L. Rivest and Yael Tauman Kalai

Handout 6
April 22, 2019
Due: May 6, 2019

Problem Set 5

This problem set is due on *Monday, May 6, 2019* at **11:59 PM**. Please note our late submission penalty policy in the course information handout. Please submit your problem set, in PDF format, on Gradescope. *Each problem should be in a separate PDF.* When submitting the problem in Gradescope, ensure that **all your group members are listed on Gradescope**, and not in the PDF alone.

You are to work on this problem set in groups of your choosing of size three or four. If you need help finding a group, try posting on Piazza or email `6.857-tas@mit.edu`. You don't have to tell us your group members, just make sure you indicate them on Gradescope. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration.

Homework must be submitted electronically! Each problem answer must be provided as a separate pdf. Mark the top of each page with your group member names, the course number (6.857), the problem set number and question, and the date. We have provided templates for LaTeX and Microsoft Word on the course website (see the *Resources* page).

Grading: All problems are worth 10 points.

With the authors' permission, we may distribute our favorite solution to each problem as the "official" solution—this is your chance to become famous! If you do not wish for your homework to be used as an official solution, or if you wish that it only be used anonymously, please note this in your profile on your homework submission.

Our department is collecting statistics on how much time students are spending on psets, etc. For each problem, please give your estimate of the number of person-hours your team spent on that problem.

Problem 5-1. Voting

This question asks you to vote using the "end-to-end verifiable" online election system "Helios" (`https://vote.heliosvoting.org/`).

Visit the Helios site. Browse the documentation. Then:

1. Create an election using Helios.
2. Vote in this election with your friends. (Challenge at least once of your cast votes.)
3. Close and tally the election.
4. Verify that your vote was properly tallied.

Then:

- Describe what you did in each of the above steps.
- Discuss the usability of the Helios system. What worked well, and what was confusing?
- Choose one aspect of the security architecture in either steps 2, 3, or 4 and describe it. What could be problematic about this feature (from a security point of view)?

Problem 5-2. Differential Privacy

Suppose we have a database $X = (x_1, \ldots, x_n)$, where each $x_i \in \{0, 1\}$, and we want to release the sum $\sum_{i=1}^{n} x_i$ in a differentially private manner.

(a) Suppose, rather than using the Laplace mechanism that was discussed in class we use the following mechanism (for a fixed $\epsilon \in (0, 1)$: For each $i \in [n]$, independently at random, let $x'_i = 1 - x_i$ with probability ϵ, and otherwise let $x'_i = x_i$ with probability $1 - \epsilon$. Release $\sum_{i=1}^{n} x'_i$. Is this mechanism ϵ-differentially private?

(b) For any $\epsilon \in (0,1)$, consider the following alternative mechanism: Choose a random subset $I \subseteq [n]$ of size $\lfloor \epsilon n \rfloor$, and for every $i \in [n]$, if $i \in I$ then let $x'_i = 1 - x_i$ and if $i \in [n] \setminus I$ then let $x'_i = x_i$. Release $\sum_{i=1}^{n} x'_i$. Is this mechanism δ-differentially private for some $\delta > 0$?

Problem 5-3. N-out-of-N BLS Signatures

Suppose Alice and Bob wish to jointly sign a message without doubling the size of their signature. Let $(sk_a, vk_a) = (x_a, g^{x_a})$ by Alice's BLS sign-verify key pair and $(sk_b, vk_b) = (x_b, g^{x_b})$ be Bob's BLS sign-verify key pair, for the BLS signature scheme we saw in class.

(a) Explain how to create a BLS verification key from Alice and Bob's verification keys such that the corresponding BLS signing key is $sk_a + sk_b = x_a + x_b$.

(b) Explain how to take two BLS signatures of a message m, one created by Alice using sk_a and one created by Bob using sk_b, and create a new signature of m that verifies under the joint verification key you created in part (a).

We can extend this scheme to n parties to prove that all n of them signed a message without growing the size of the signature (or the work required to verify). We will call this scheme an n-out-of-n BLS signature scheme.

Now, let's prove the security of this scheme.

We will first specify the security game that we are considering. We will say that a signature scheme is secure if the probability that the adversary wins the following game is negligible:

1. The adversary begins by committing to a message m^*.
2. The challenger sends the adversary a verification key vk.
3. Repeat the following steps a polynomial number of times:
   (a) The adversary sends a message $m \neq m^*$ to the challenger.
   (b) The challenger responds with a signature σ such that $\mathsf{Verify}(vk, m, \sigma) = \mathsf{True}$.
4. The adversary sends σ^* to the challenger. The adversary wins if $\mathsf{Verify}(vk, m^*, \sigma^*) = \mathsf{True}$

We will say that our n-out-of-n signature scheme is secure if the probability that the adversary wins the following game is negligible:

1. The adversary begins by committing to a message m^*.
2. The challenger sends the adversary a verification keys $vk_1, \ldots, vk_n$ as well as the signing keys $sk_1, \ldots, sk_{n-1}$. Call vk_S the joint verification key as created in part (a).
3. Repeat the following steps a polynomial number of times:
   (a) The adversary sends a message $m \neq m^*$ to the challenger.
   (b) The challenger responds with a signature σ such that $\mathsf{Verify}(vk_S, m, \sigma) = \mathsf{True}$.
4. The adversary sends σ^* to the challenger. The adversary wins if $\mathsf{Verify}(vk_S, m^*, \sigma^*) = \mathsf{True}$

Suppose there exists an adversary that can win the second game when the signature scheme is our n-out-of-n BLS signature scheme, which would show that our n-out-of-n scheme is not secure. We will use this adversary to construct a new adversary that can win the first game when the signature scheme is the regular BLS scheme we saw in class.

We're going to think of this adversary as being in the middle of the challenger for the first game and the adversary that can win the second game. To the challenger for the first game, this adversary should appear to be a normal adversary, but to the adversary for the second game, this adversary should look like the challenger for the second game. In other words, this adversary should play both games at once, acting as the adversary in the first game and the challenger in the second game.

(c) Suppose you are given a BLS n-out-of-n signature of a message m that verifies under a verification key that is a joint of $vk_1, \ldots, vk_n$. Suppose, also, that you know the corresponding signing keys $sk_1, \ldots, sk_{n-1}$. Create a BLS signature of m that verifies under verification key vk_n (the verification key corresponding to the one signing key you do not know).

(d) Let $\mathcal{A}$ be an adversary that can win the second game for the n-out-of-n BLS signature scheme. Use the technique in the previous part to construct an adversary $\mathcal{B}$ that wins the first game with the BLS signature scheme we saw in class by acting as the challenger in the second game with $\mathcal{A}$.

Conclude that the new n-out-of-n scheme is as secure as the original BLS scheme.