6.893: Cryptographic Techniques for Protecting Privacy **Fall 2020**

Problem Set 2

Due: October 2, 2020 at 5pm, Boston time via Gradescope.

Instructions: You **must** typeset your solution in LaTeX using the provided template:

`https://6893.csail.mit.edu/homework.tex`

Submission Instructions: You must submit your problem set via Gradescope. Please use course code **MGWNYV** to sign up. For ease of grading, please begin each *part* of each question on a new page.

Bugs: I make mistakes! If it looks like there might be a mistake in the statement of a problem, please ask a clarifying question on Piazza.

Problem 1: True/False [5 points].

(a) Computational assumptions are necessary for non-trivial single-server PIR.

(b) Computational assumptions are necessary for non-trivial two-server PIR.

(c) Under suitable computation assumptions, there exist PIR schemes with 100 servers that can maintain client privacy even against an adversary that controls all 100 servers.

(d) The servers in PIR maintain state that changes with each PIR query that the client makes.

(e) The server in ORAM maintains state that changes with each ORAM operation that the client makes.

Problem 2: Reducing Computation in PIR [10 points].

(a) Construct a two-server PIR protocol in which the client sends $10\log_2 n$ bits to each server and receives $n/(10\log_2 n)$ bits from each server in return.

(b) Construct a two-server PIR protocol in which each server precomputes a data structure of size $\text{poly}(n)$ (i.e., the data structure depends on the database) such that, given this data structure, the server can respond to client queries in time $o(n)$.

Hint: Use your construction from Part (a).

(c) In class, I mentioned that there is an information-theoretic two-server PIR scheme of Dvir and Gopi that has communication $n^{O(\sqrt{\log\log n/\log n})} = n^{o(1)}$. Show that this result implies that for any constant $c > 0$, there exists a two-server PIR scheme in which the client uploads $O(\log n)$ bits to each server and receives $O(n/\log^c n)$ bits in response.

(d) Use your construction from Part (c) to show that there is a two-server PIR scheme in which, for any $c > 0$, each server stores a preprocessed data structure of size $\text{poly}(n)$ and can respond to client queries in time $O(n/\log^c n)$.

Problem 3: Reducing Communication in PIR [15 points]. Throughout this question, we consider one-round information-theoretic PIR over an n-bit database.

In class, we saw a simple two-server PIR with $O(n^{1/2})$ communication complexity. In this problem, you will first construct a *four*-server PIR scheme with communication complexity $O(n^{1/3})$. Then you will construct a *two*-server PIR with much improved $O(n^{1/3})$ communication complexity. This $O(n^{1/3})$ scheme was essentially the best-known two-server PIR scheme for many many years (until 2015), so in this problem you will reprove a very nice and very non-trivial result.

(a) In the following box, we describe a four-server PIR scheme with $O(\sqrt{n})$ communication. Prove that the scheme is correct. Explain *informally* in 2-3 sentences why the scheme is secure as long as the adversary controls at most *one* server.
(**Hint**: Using matrix notation will make your life easy. The correctness argument should not require more than a few lines of math.)

Four-Server $O(\sqrt{n})$-Communication PIR Scheme

Write the n-bit database as a matrix $X \in \mathbb{Z}_2^{\sqrt{n}\times\sqrt{n}}$. The client wants to read the bit X_{ij} from this database, where $i, j \in [\sqrt{n}]$. Recall that $e_i \in \mathbb{Z}_2^{\sqrt{n}}$ is the dimension-$\sqrt{n}$ vector that is zero everywhere except with a "1" at position i.

- $\mathsf{Query}(i, j) \to (q_{00}, q_{01}, q_{10}, q_{11})$.
  Sample random vectors $r_0, r_1, s_0, s_1 \in \mathbb{Z}_2^{\sqrt{n}}$ subject to $r_0 + r_1 = e_i \in \mathbb{Z}_2^{\sqrt{n}}$ and $s_0 + s_1 = e_j \in \mathbb{Z}_2^{\sqrt{n}}$.
  For $b_0, b_1 \in \{0,1\}$, let $q_{b_0 b_1} \leftarrow (r_{b_0}, s_{b_1})$.
  Output $(q_{00}, q_{01}, q_{10}, q_{11})$.
- $\mathsf{Answer}(X, q) \to a$.
  Parse the query q as a pair (r, s) with $r, s \in \mathbb{Z}_2^{\sqrt{n}\times 1}$.
  Return as the answer the single bit $a \leftarrow r^T X s \in \mathbb{Z}_2$.
- $\mathsf{Reconstruct}(a_{00}, a_{01}, a_{10}, a_{11}) \to X_{ij}$.
  Output $X_{ij} \leftarrow a_{00} + a_{01} + a_{10} + a_{11} \in \mathbb{Z}_2$.

(b) Say that you have a k-server PIR scheme that requires the client to upload $U(n)$ bits to each server and download one bit from each server. Explain how to use this scheme to construct a k-server PIR scheme in which, for any $\ell \in \mathbb{N}$, each client uploads $U(n/\ell)$ bits to each server and downloads ℓ bits from each server. (You may assume that n is a multiple of ℓ.)

Sketch—without a formal proof—why your construction does not break the correctness or security of the initial PIR scheme.

(c) Show how to combine parts (a) and (b) get a four-server PIR scheme with total communication $O(n^{1/3})$. In particular, you should calculate the optimal value of the parameter ℓ used in part (b).

(d) Sketch how to generalize the PIR scheme in part (a) to give an eight-server PIR scheme in which the client sends $O(n^{1/3})$ bits to each server and receives a single bit from each server in return. This should only take a few sentences to describe.

(e) Now comes the grand finale! Use the *eight*-server scheme from part (d) to construct a *two*-server scheme with communication $O(n^{1/3})$.

Hint:

- Label the queries of the eight-server scheme from part-(d) as $q_{000}, q_{001}, q_{010}, \ldots, q_{111}$. The two queries in your new two-server scheme should be q_{000} and q_{111} from the eight-server scheme.
- The two servers can clearly send back the 1-bit answers for q_{000} and q_{111} respectively. NOW, here is the beautiful idea: show that by sending back to the client $O(n^{1/3})$ additional bits, each of the two servers can enable the client to recover the answers for three additional queries.

Problem 4: Dangers of Counter Mode [15 points]. One of the most popular modes of operation for a block cipher, such as AES, is Galois counter mode (GCM). At its core, AES-GCM encrypts a message by first generating a stream of pseudorandom bytes and then XORing these bytes with the message. (There is also a message authentication tag appended to the message, but that is not relevant for this problem.)

In this problem, we will see that encrypting long messages with AES-GCM is unsafe. This relates to the limits of the PRP-PRF Switching Lemma, which we discussed in class.

Let $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher over keyspace $\mathcal{K}$. Say that the attacker gets the counter-mode encryption of the message

$$(\underbrace{0^n \| 0^n \| \cdots \| 0^n}_{T \text{ times}} \| \underbrace{S \| S \| \cdots \| S}_{T \text{ times}}),$$

where $S \in \{0,1\}^n$ is a secret value. That is, the attacker gets the ciphertext:

$$\big(E(k,1) \| E(k,2) \| \cdots \| E(k,T) \quad \| \quad E(k,T+1) \oplus S \| E(k,T+2) \oplus S \| \cdots \| E(k,2T) \oplus S\big).$$

Here, we identify integers in $\{1,\ldots,2T\}$ with n-bit strings in the natural way. When we use counter mode in practice, we start the counter at a random value—the IV—which is critical for maintaining semantic security when encrypting many messages with the same key. To simplify the discussion here, assume that the counter always starts at 1.

Throughout this problem, take $T = 2^{n/2} \cdot \text{poly}(n)$, where the polynomial can be any one of your choosing.

(a) Explain (in ≤ 3 sentences) one practical context in which an attacker might be able to obtain the encryption of a bunch of zeros followed by many repetitions of a secret message.

(b) Show that given only the ciphertext ct, an attacker can recover the secret $S \in \{0,1\}^n$ in time $2^n \cdot \text{poly}(n)$. You need not give a precise formal proof that your attack runs in this amount of time, but you should sketch an argument for why it does. (**Hint:** Your attack should make use of the fact that $E(k,\cdot)$ is a permutation on $\{0,1\}^n$. **Another hint:** This is the hardest part of the problem. You might want to solve the other parts first and then come back to this.)

Notice that the attack cost here is independent of the size of the keyspace. So if you run AES-GCM with 256-bit keys, and you encrypt a message of size $\approx 2^{64}$, there is a message-recovery attack of this form that runs in time $\approx 2^{128} \ll 2^{256}$.

(c) For some string $\sigma \in \{0,1\}^{n-1}$, let $s_0 = (\sigma \| 0) \in \{0,1\}^n$ and $s_1 = (\sigma \| 1) \in \{0,1\}^n$. Show that if the attacker knows that the secret S is either s_0 or s_1, then the attacker can recover the secret in time $2^{n/2} \cdot \text{poly}(n)$.

(d) Say now that the attacker can obtain the encryption of many messages, each using the same secret S, but with distinct AES keys. In particular, for any string σ of the attacker's choosing (provided that $|\sigma| \leq \text{poly}(n)$), the attacker can obtain the counter-mode encryption of the string:

$$M_\sigma = (\underbrace{0^n \| \cdots \| 0^n}_{T \text{ times}}) \quad \| \quad \sigma \quad \| \quad (\underbrace{S \| 0^n \| S \| 0^n \| S \| \cdots \| S \| 0^n}_{T \text{ times}}).$$

Show that if the attacker can obtain encryptions of this form, it can recover the secret S in time $2^{n/2} \cdot \text{poly}(n)$. (**Hint:** Use your attack from Part (c) as a subroutine. **Another hint:** Think about what happens when σ is less than n bits long.)

Notice that since $2^{n/2} \ll 2^n$, this attack runs *much* faster than the attack of Part (b). The trick that you will use in this part is one of the very useful tools in real-world cryptanalysis.

Problem 5: Feedback [1 points].

(a) Roughly how many hours did you spend on this problem set?

(b) What was your favorite problem on this problem set? In one sentence: why?

(c) What was your least favorite problem on this problem set? In one sentence: why?

(d) **[Optional]** If you have any other feedback on this problem set or on the course, please write it here or submit it using the anonymous feedback form.