CS 355: Topics in Cryptography **Spring 2019**

Problem Set 5

Due: June 7, 2019 at 5pm (submit via Gradescope)

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

https://crypto.stanford.edu/cs355/19sp/homework.tex

Submission Instructions: You must submit your problem set via Gradescope. Please use course code **9KY4BB** to sign up. Note that Gradescope requires that the solution to each problem starts on a **new page**.

Problem 1: Conceptual Questions [10 points]. For each of the following statements, say whether it is TRUE or FALSE. Write *at most one sentence* to justify your answer.

(a) Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a pairing for an elliptic curve group $\mathbb{G}$. If the discrete log problem in $\mathbb{G}$ is easy, then the BDDH problem for e is easy as well.

(b) Suppose that LWE is hard for some parameter values n, $m \gg n$, and q and noise vector $\mathbf{e}$ sampled from some distribution χ_m over $\mathbb{Z}_q^m$. Then LWE is also hard for parameter values n, $2m$, and q and noise vector $\mathbf{e}$ sampled from the distribution $\hat{\chi}_{2m} = \left\{ \begin{bmatrix} \mathbf{e} \\ \mathbf{e} \end{bmatrix} : \mathbf{e} \xleftarrow{R} \chi_m \right\}$ over $\mathbb{Z}_q^{2m}$.

(c) For every large enough $n \in \mathbb{N}$, $m = n^2$, $q = n$ and $B = n/4$, there exists a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ such that every vector $\mathbf{u} \in \mathbb{Z}_q^m$ can be written in the form $\mathbf{u} = \mathbf{As} + \mathbf{e}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ and $\mathbf{e} \in [-B, B]^m$.

(d) Given an encryption of a message $m \in \{0,1\}^n$ in a private-key FHE scheme, it is possible to create a valid encryption of an arbitrary message $m' \in \{0,1\}^n$, without access to the private key.

(e) Every *designated-verifier* SNARG can be converted into a *public-verifier* SNARG by attaching the verification state to the public CRS (common reference string).

Problem 2: On The Importance of Elliptic-Curve Point Validation [10 points]. Let $E\colon y^2 = x^3 + Ax + B$ be an elliptic curve over $\mathbb{F}_p$, where $q := |E(\mathbb{F}_p)|$ is a prime number and $P \in E(\mathbb{F}_p)$ is a generator. Observe that the elliptic-curve group addition formulae are *independent of the parameter B* of the curve equation. In particular, for every curve $\hat{E}\colon y^2 = x^3 + Ax + \hat{B}$ for some $\hat{B} \in \mathbb{F}_p$, applying the formulae for addition in $E(\mathbb{F}_p)$ to any two points $\hat{V}, \hat{W} \in \hat{E}(\mathbb{F}_p)$ gives the point $\hat{V} \boxplus \hat{W} \in \hat{E}(\mathbb{F}_p)$.

As a result, all parties in a cryptographic protocol must verify that adversarially chosen points are on the right curve, and failing to do so may break security. We exemplify this by considering a variant of elliptic-curve Diffie-Hellman key exchange in which the server uses the same key pair across multiple sessions. More specifically, the server holds a *fixed* secret key $\alpha \xleftarrow{R} \mathbb{Z}_q$ and advertises (e.g., in its TLS certificate) the corresponding fixed public key $\alpha P \in E(\mathbb{F}_p)$. A client connects to the server by choosing $\beta \xleftarrow{R} \mathbb{Z}_q$, computing $V = \beta P$, and sending V to the server. Both sides then compute the shared secret $W = \alpha\beta P$. For simplicity, we assume that the server then sends the message $E_s(W, \text{“Hello!”})$ to the client, where (E_s, D_s) is some symmetric cipher.

(a) Explain how the server can check that the point V it receives from the client is indeed in $E(\mathbb{F}_p)$.

(b) Suppose there exists a curve $\hat{E}\colon y^2 = x^3 + Ax + \hat{B}$ such that $|\hat{E}(\mathbb{F}_p)|$ is divisible by a small prime t (i.e., $t = O(\text{polylog}(q))$). Show that if the server *does not check* that $V \in E(\mathbb{F}_p)$, a malicious client can efficiently learn $\alpha \bmod t$. You may assume one can efficiently find a point of order t in $\hat{E}(\mathbb{F}_p)$.

(c) Use Part (b) to show how a malicious client can efficiently learn the secret key α, if the server *does not check* that $V \in E(\mathbb{F}_p)$. You may assume that if $\hat{B} \xleftarrow{R} \mathbb{F}_p$, then $|\hat{E}(\mathbb{F}_p)|$ is uniform in $\left[p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p}\right]$ and is efficiently computable. (As in Part (b), you may assume that whenever the order of a curve has a small prime factor t, one can efficiently find a point of order t on that curve.)

Problem 3: Private Information Retrieval from Fully Homomorphic Encryption [10 points].

(a) Show how to use an FHE scheme to construct a single-server one-round PIR scheme in which the client uploads $O(\log n)$ ciphertexts to the server, for an n-bit database, and the server responds with a single ciphertext to the client.

(b) Say that the database contains n pairs $(\text{name}_i, \text{salary}_i)$. The client holds a string $\sigma \in \{0,1\}^\ell$ and wants to learn the sum of the salaries of the people whose names begin with the string σ. Use FHE to construct a single-server one-round PIR scheme in which the client uploads ℓ FHE ciphertexts and the server responds with $O(\log(n \cdot \texttt{SALARY\_MAX}))$ ciphertexts.

(c) Suppose that you have a *somewhat* homomorphic encryption scheme that supports computing boolean circuits consisting of only OR and AND gates of AND-depth $d+1$ (i.e., there are at most $d+1$ AND gates between any input and the output wire) for some constant $d \geq 2$. Construct a single-server one-round PIR scheme in which the client sends $O(n^{1/d})$ ciphertexts to the server and the server replies with a single ciphertext. Note that the constant in the big-O can depend on d. (It is actually sufficient to have an FHE scheme that supports circuits of AND-depth $O(\log d)$ but it is not required in this problem.)

Problem 4: Key-Exchange from LWE [18 points]. In this problem, we will formalize the concept of a *non-interactive key exchange* (NIKE) protocol, and then construct it from LWE. NIKE protocols are a core component of Internet protocols like TLS, and the lattice-based NIKE that we develop in this problem is a simplified variant of some of the leading candidates in the NIST competition for standardizing post-quantum key-exchange.

A *non-interactive key exchange* (NIKE) protocol for a key space $\mathcal{K}$ consists of the following PPT algorithms:

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$: On input the security parameter λ, the setup algorithm outputs the public parameters pp.
- $\mathsf{ClientPublish}(\mathsf{pp}) \to (\mathsf{priv}, \mathsf{pub})$: On input the public parameters pp, the client-publish algorithm outputs a secret value priv, and a public message pub.
- $\mathsf{ServerPublish}(\mathsf{pp}) \to (\mathsf{priv}, \mathsf{pub})$: On input the public parameters pp, the server-publish algorithm outputs a secret value priv, and a public message pub.
- $\mathsf{KeyGen}(\mathsf{priv}, \mathsf{pub}) \to \mathsf{key}$: On input a secret value priv, and a public message pub, the key generation algorithm outputs a key $\mathsf{key} \in \mathcal{K}$.

Correctness. We require that for all $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{pub}_0, \mathsf{priv}_0) \leftarrow \mathsf{ClientPublish}(\mathsf{pp})$, $(\mathsf{pub}_1, \mathsf{priv}_1) \leftarrow \mathsf{ServerPublish}(\mathsf{pp})$, we have

$$\Pr\left[\mathsf{KeyGen}(\mathsf{priv}_0, \mathsf{pub}_1) = \mathsf{KeyGen}(\mathsf{priv}_1, \mathsf{pub}_0)\right] = 1 - \mathrm{negl}(\lambda).$$

Security. For a NIKE protocol $(\mathsf{Setup}, \mathsf{ClientPublish}, \mathsf{ServerPublish}, \mathsf{KeyGen})$, we define the following two experiments:

Experiment b $\quad (b = 0, 1)$:

- The challenger computes the following:
  $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$,
  $(\mathsf{priv}_0, \mathsf{pub}_0) \leftarrow \mathsf{ClientPublish}(\mathsf{pp})$,
  $(\mathsf{priv}_1, \mathsf{pub}_1) \leftarrow \mathsf{ServerPublish}(\mathsf{pp})$,
  $\mathsf{key}_0 \leftarrow \mathsf{KeyGen}(\mathsf{priv}_0, \mathsf{pub}_1)$,
  $\mathsf{key}_1 \xleftarrow{R} \mathcal{K}$.
  It provides $(\mathsf{pp}, \mathsf{pub}_0, \mathsf{pub}_1, \mathsf{key}_b)$ to the adversary.
- The adversary outputs a bit $\hat{b} \in \{0, 1\}$.

Let W_b be the event that $\mathcal{A}$ outputs 1 in Experiment b. Then, we say that a NIKE protocol is secure if

$$\left|\Pr[W_0] - \Pr[W_1]\right| = \mathrm{negl}(\lambda).$$

(a) Explain in words why the security definition above captures our intuitive notion of security for key-exchange.

(b) Construct a NIKE protocol from the decisional Diffie-Hellman assumption (DDH).[1] Use the following setup algorithm:

$\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$: Let $\mathbb{G}$ be a cyclic group of prime order p for which the DDH problem is hard, and let $g \in \mathbb{G}$ be a generator. Set $\mathsf{pp} = (\mathbb{G}, g)$.

You should specify the key space $\mathcal{K}$, define the algorithms $(\mathsf{ClientPublish}, \mathsf{ServerPublish}, \mathsf{KeyGen})$, and prove correctness/security of your scheme under the DDH assumption.

(c) Consider the following NIKE protocol:[2]

> Let $n = \text{poly}(\lambda), q, \chi_B$ be parameters for which $\mathsf{LWE}_{\mathsf{HNF}}(n, n, q, \chi_B)$ and $\mathsf{LWE}_{\mathsf{HNF}}(n, n+1, q, \chi_B)$ is hard. Recall from lecture that in practice, for $\lambda = 128$, we use $n \approx 800$.
>
> Define the key space $\mathcal{K} = \{0,1\}$ and consider the following algorithms.
>
> - $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$: Sample a matrix $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ and set $\mathsf{pp} = \mathbf{A}$.
> - $\mathsf{ClientPublish}(\mathsf{pp}) \to (\mathsf{priv}, \mathsf{pub})$: Sample vectors $\mathbf{s} \leftarrow \chi_B^n$, $\mathbf{e} \leftarrow \chi_B^n$. Then, set $\mathsf{priv} = \mathbf{s}$, and $\mathsf{pub} = \mathbf{A}^T\mathbf{s} + \mathbf{e}$.
> - $\mathsf{ServerPublish}(\mathsf{pp}) \to (\mathsf{priv}, \mathsf{pub})$: Sample vectors $\mathbf{s} \leftarrow \chi_B^n$, $\mathbf{e} \leftarrow \chi_B^n$. Then, set $\mathsf{priv} = \mathbf{s}$, and $\mathsf{pub} = \mathbf{A}\mathbf{s} + \mathbf{e}$.
> - $\mathsf{KeyGen}(\mathsf{priv}, \mathsf{pub}) \to \mathsf{key}$: Let $\mathsf{priv} = \mathbf{s} \in \mathbb{Z}_q^n$ and $\mathsf{pub} = \mathbf{b} \in \mathbb{Z}_q^n$. The key generation algorithm first samples a small noise term $e \leftarrow \chi_B$. Then, if $\|\langle \mathbf{s}, \mathbf{b} \rangle + e\|_\infty \le \lfloor q/4 \rceil$, set $\mathsf{key} = 0$. Otherwise, set $\mathsf{key} = 1$.

Suppose that q is prime and chosen to satisfy $4nB^2/q = \text{negl}(\lambda)$. Prove that the protocol satisfies correctness. For the proof, feel free to use the following fact (you do not need to prove this fact):

> For any prime q, for $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ any two non-zero vectors $\mathbf{s}_0, \mathbf{s}_1 \in \mathbb{Z}_q^n$, and $c \in \mathbb{Z}_q$,
>
> $$\Pr_{\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}} \left[\mathbf{s}_0^T \mathbf{A} \mathbf{s}_1 = c\right] = 1/q - \text{negl}(\lambda),$$
>
> where the probability is over the random choice of $\mathbf{A}$.

(d) Prove that the protocol above is secure assuming $\mathsf{LWE}_{\mathsf{HNF}}(n, n, q, \chi_B)$ and $\mathsf{LWE}_{\mathsf{HNF}}(n, n+1, q, \chi_B)$. The definition of $\mathsf{LWE}_{\mathsf{HNF}}$ is on the last page of this problem set. [**Hint:** Use a hybrid argument.]

[1] Definition 10.8 in Boneh-Shoup (pg. 405).

[2] We restrict the key space to $\mathcal{K} = \{0,1\}$ for simplicity. To get a NIKE protocol for $\mathcal{K} = \{0,1\}^{128}$, we can simply run 128 parallel instances of the protocol using the same public matrix $\mathbf{A}$.

Problem 5: SNARGs in the Random Oracle Model [12 points]. In this problem, we will show how to leverage probabilistically-checkable proofs (PCPs) to construct a succinct non-interactive argument (SNARG) in the random oracle model. We will rely on the following adaptation of the famous PCP theorem:

Theorem (PCP)**.** Let $\mathcal{L}$ be an NP language. There exists two efficient algorithms $(\mathcal{P}, \mathcal{V})$ defined as follows:

- The prover algorithm $\mathcal{P}$ is a deterministic algorithm that takes as input a statement $x \in \{0,1\}^n$, a witness $w \in \{0,1\}^h$ and outputs a bitstring $\pi \in \{0,1\}^m$, where $h, m = \text{poly}(n)$. We refer to π as the proof string.
- The verifier algorithm $\mathcal{V}^\pi$ is a *randomized* algorithm that takes as input a statement $x \in \{0,1\}^n$ and has oracle access to a proof string $\pi \in \{0,1\}^m$. The verifier reads $O(1)$ bits of π. The verifier chooses the bits it reads *nonadaptively* (i.e., they can depend on the statement x, but *not* on the values of any bit in π).

Moreover, $(\mathcal{P}, \mathcal{V})$ satisfy the following properties:

- **Completeness:** For all $x \in \mathcal{L}$, if w is a valid witness for x, then

$$\Pr[\mathcal{V}^\pi(x) = 1 : \pi \leftarrow \mathcal{P}(x, w)] = 1.$$

- **Soundness:** If $x \notin \mathcal{L}$, then for all $\pi \in \{0,1\}^m$,

$$\Pr[\mathcal{V}^\pi(x) = 1] \leq 1/2.$$

(a) Let λ be a security parameter and let $H : \{0,1\}^* \to \{0,1\}^\lambda$ be a collision-resistant hash function. Use H to construct a commitment scheme $(\mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ with the following properties:

- $\mathsf{Commit}(x) \to c$: The commitment algorithm should take a message $x \in \{0,1\}^m$ and output a commitment $c \in \{0,1\}^\lambda$.
- $\mathsf{Open}(x, c, i) \to \sigma$: The open algorithm takes a message $x \in \{0,1\}^m$, a commitment $c \in \{0,1\}^\lambda$, and an index $i \in [m]$, and outputs an opening σ.
- $\mathsf{Verify}(c, i, b, \sigma) \to \{0,1\}$: The verification algorithm takes a commitment $c \in \{0,1\}^\lambda$, an index $i \in [m]$, a value $b \in \{0,1\}$, and an opening σ, and outputs a bit.

Show that your commitment scheme satisfies the following properties:

- **Completeness:** For all $x \in \{0,1\}^m$ and $i \in [m]$,

$$\Pr[\mathsf{Verify}(c, i, x_i, \sigma) = 1 : c \leftarrow \mathsf{Commit}(x); \sigma \leftarrow \mathsf{Open}(x, c, i)] = 1.$$

- **Binding:** For all efficient adversaries $\mathcal{A}$, if we set $(c, i, (b, \sigma), (b', \sigma')) \leftarrow \mathcal{A}(1^\lambda)$, then

$$\Pr[b \neq b' \text{ and } \mathsf{Verify}(c, i, b, \sigma) = 1 = \mathsf{Verify}(c, i, b', \sigma')] = \text{negl}(\lambda).$$

- **Succinctness:** The commitment c output by Commit and opening σ output by Open satisfy $|c| = O(\lambda)$ and $|\sigma| = O(\lambda \log m)$.

In other words, the commitment scheme $(\mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ allows a user to succinctly commit to a long bitstring and then selectively open up a single bit of the committed string. (In this question, we do not require any hiding properties from the commitment scheme.)

(b) Let $\mathcal{L}$ be an NP language (with statements of length n). Show how to construct a 3-round succinct argument system for $\mathcal{L}$ using your commitment scheme from Part (a). Specifically, your argument system should satisfy perfect completeness, have soundness error $\mathrm{negl}(\lambda)$ against computationally-bounded provers, and the total communication complexity between the prover and the verifier should be $\mathrm{poly}(\lambda, \log n)$. In particular, the communication complexity scales *polylogarithmically* with the length of the NP statement. [**Hint:** Use the PCP theorem.]

(c) Explain how to convert your succinct argument from Part (b) into a SNARG in the random oracle model.

Problem 6: Time Spent [3 points for answering]. How long did you spend on this problem set? This is for calibration purposes, and the response you provide will not affect your score.

Optional Feedback [0 points]. Please answer the following questions to help us design future problem sets. You do not need to answer these questions, and if you would prefer to answer anonymously, please use this form. However, we do encourage you to provide us feedback on how to improve the course experience.

(a) What was your favorite problem on this problem set? Why?

(b) What was your least favorite problem on this problem set? Why?

(c) Do you have any other feedback for this problem set?

(d) Do you have any other feedback on the course so far?

Appendix: Definition of LWE in Hermite Normal Form.

We review the formal definitions of the Learning with Errors problem in *Hermite Normal Form*. Note that in this variant of the LWE problem, the vector $\mathbf{s}$ is sampled from the B-bounded error distribution χ_B instead of the uniform distribution. This version of the LWE problem is known to be as hard as the standard LWE problem.

$\mathsf{LWE}_{\mathsf{HNF}}(n, m, q, \chi_B)$: Let $n, m, q, B \in \mathbb{N}$ be positive integers, and let χ_B be a B-bounded distribution over $\mathbb{Z}_q$. For a given adversary $\mathcal{A}$, we define the following two experiments:

Experiment b $\quad (b = 0, 1)$:

- The challenger computes

$$\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{m \times n}, \quad \mathbf{s} \leftarrow \chi_B^n, \quad \mathbf{e} \leftarrow \chi_B^m, \quad \mathbf{b}_0 \leftarrow \mathbf{A} \cdot \mathbf{s} + \mathbf{e}, \quad \mathbf{b}_1 \xleftarrow{R} \mathbb{Z}_q^m,$$

  and gives the tuple $(\mathbf{A}, \mathbf{b}_b)$ to the adversary.

- The adversary outputs a bit $\hat{b} \in \{0, 1\}$.

Let W_b be the event that $\mathcal{A}$ outputs 1 in Experiment b. Then, we define $\mathcal{A}$'s advantage in solving the $\mathsf{LWE}_{\mathsf{HNF}}$ problem for the set of parameters n, m, q, χ_B to be

$$\mathsf{HNF\text{-}LWEAdv}_{n,m,q,\chi_B}[\mathcal{A}] := \Big|\Pr[W_0] - \Pr[W_1]\Big|.$$