CS 355: Topics in Cryptography **Spring 2018**

Problem Set 4

Due: May 25, 2018 at 5pm (submit via Gradescope)

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

https://crypto.stanford.edu/cs355/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 [12 points]. For each of the following statements, say whether it is TRUE or FALSE. Write *at most one sentence* to justify your answer.

(a) For all positive integers n, m, q, B, the $\mathsf{SIS}(n, m, q, B)$ problem is at least as hard as the $\mathsf{SIS}(n, m+1, q, B)$ problem.

(b) For all positive integers n, m, q, the $\mathsf{SIS}(n, m+1, q, 1)$ problem is at least as hard as the $\mathsf{ISIS}(n, m, q, 1)$ problem. (Note: $B = 1$).

(c) Let p, q, r, and r' be distinct large primes. Let $N = pqr$ and $N' = pqr'$. Assume that there does *not* exist an efficient (probabilistic polynomial time) factoring algorithm. Say whether each of the following statements are TRUE or FALSE.

- There is an efficient algorithm that takes N as input and outputs r.
- There is an efficient algorithm that takes N and N' as input and outputs r.
- There is an efficient algorithm that takes N and N' as input and outputs q.

(d) We saw in lecture that using the Schnorr signature scheme to sign two distinct messages with the same signing randomness (nonce) causes a dramatic security failure (i.e., anyone with the two signatures and messages can recover the signer's private key). The same nonce-reuse attack applies to the EC-DSA signing algorithm.

Problem 2: 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 $(b = 0, 1)$:

- 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:

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

$\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$: 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, any two non-zero vectors $\mathbf{s}_0, \mathbf{s}_1 \in \mathbb{Z}_q^n$, and $c \in \mathbb{Z}_q$,
>
> $$\Pr_{\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}} \left[\mathbf{s}_0^T \mathbf{A} \mathbf{s}_1 = c\right] = 1/q - \text{negl}(\lambda).$$

(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.]

Problem 3: Homomorphic Signatures from Lattices [12 points]. Recall from lecture that a homomorphic encryption scheme enables computation on encrypted data. In this problem, we will develop an analogous scheme for computing on *signed* data (i.e., a *homomorphic signature* scheme). We first describe the high-level semantics of a homomorphic signature scheme:

- A homomorphic signature scheme enables a party who knows a message $x \in \{0,1\}^\ell$ and a signature σ_x on x to compute a signature $\sigma_{f(x)}$ on any function evaluation $f(x)$.

[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 $\mathbf{A}$.

- The signature verification procedure takes a signature σ, a function f, and a value z and decides whether σ is a valid signature on z with respect to the function f.
- The high-level security requirement is that no efficient adversary who has message-signature pairs $(x_1, \sigma_{x_1}), \ldots, (x_n, \sigma_{x_n})$ can produce a signature σ_z on some value z with respect to a function f where $z \notin \{f(x_1), \ldots, f(x_n)\}$. Namely, the only signatures an adversary can compute are signatures on functions of signatures it already possesses.

In this problem, we describe how to build a *one-time* homomorphic signature scheme (where security holds against an adversary that sees a single message-signature pair).[3]

- Let $\{0,1\}^\ell$ be the message space for the signature scheme. Let n, m, q, B be lattice parameters.
- The public key for the signature scheme consists of $\ell + 1$ matrices $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell \in \mathbb{Z}_q^{n \times m}$ and the secret key is a trapdoor $\mathsf{td}_{\mathbf{A}}$ for $\mathbf{A}$. Namely, during key-generation, we sample $(\mathbf{A}, \mathsf{td}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(n, m, q)$. The matrices $\mathbf{B}_1, \ldots, \mathbf{B}_\ell \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ are sampled uniformly at random.
- A signature for a message $x \in \{0,1\}^\ell$ consists of ℓ matrices $\mathbf{R}_1, \ldots, \mathbf{R}_\ell \in \mathbb{Z}_q^{m \times m}$ where $\mathbf{A}\mathbf{R}_i + x_i \cdot \mathbf{G} = \mathbf{B}_i$, $\|\mathbf{R}_i\|_\infty \leq B$ for all $i \in [\ell]$, and $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ is the gadget matrix. For a matrix $\mathbf{R}_i$, we write $\|\mathbf{R}_i\|_\infty$ to denote the *maximum* entry in $\mathbf{R}_i$. Intuitively, we can view $\mathbf{R}_i$ as a signature on the i^{th} bit x_i of the message. Throughout this problem, the bound B is the norm bound on vectors output by the trapdoor preimage sampling algorithm.
- Let $f\colon \{0,1\}^\ell \to \{0,1\}$ be a function. To verify a signature $\mathbf{R}_{f,z} \in \mathbb{Z}_q^{m \times m}$ on some message $z \in \{0,1\}$ with respect to the function f, the verifier checks that $\|\mathbf{R}_{f,z}\|_\infty \leq B_f$, where B_f is a bound that depends on B and f, and moreover, that $\mathbf{A}\mathbf{R}_{f,z} + z \cdot \mathbf{G} = \mathbf{B}_f$, where $\mathbf{B}_f \in \mathbb{Z}_q^{n \times m}$ is a matrix that *only* depends on the public key $(\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell)$ and the function f. In particular, this means that the verifier can verify a signature σ with respect to a function f *without* needing access to the original signatures used to generate σ. In fact, homomorphic signatures are compact: the length of a signature $\sigma_{f(x)}$ on $f(x)$ depends only on the depth of the circuit computing f and is *independent* of the size of the input x.

In this problem, we will complete the specification of the homomorphic signature scheme.

(a) **Signing.** Describe how the signing algorithm works. Namely, given the public key $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell$, the secret key $\mathsf{td}_{\mathbf{A}}$, and a message $x \in \{0,1\}^\ell$, show how to construct the signature $\mathbf{R}_1, \ldots, \mathbf{R}_\ell \in \mathbb{Z}_q^{m \times m}$ where $\|\mathbf{R}_i\|_\infty \leq B$ and $\mathbf{A}\mathbf{R}_i + x_i \cdot \mathbf{G} = \mathbf{B}_i$ for all $i \in [\ell]$.

(b) **Homomorphic addition.** Let $\mathbf{R}_i, \mathbf{R}_j$ be signatures on bits x_i, x_j where $\|\mathbf{R}_i\|_\infty, \|\mathbf{R}_j\|_\infty \leq B$, $\mathbf{A}\mathbf{R}_i + x_i \cdot \mathbf{G} = \mathbf{B}_i$ and $\mathbf{A}\mathbf{R}_j + x_j \cdot \mathbf{G} = \mathbf{B}_j$. Show how to construct a signature $\mathbf{R}_{x_i + x_j}$ on the sum $x_i + x_j$, where $\|\mathbf{R}_{x_i+x_j}\|_\infty \leq 2B$. Specifically, your solution should specify the following:

- How to compute the signature $\mathbf{R}_{x_i+x_j}$ from the public parameters $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell$, the signatures $\mathbf{R}_i, \mathbf{R}_j$, and the values x_i, x_j. Moreover, verify that $\|\mathbf{R}_{x_i+x_j}\|_\infty \leq 2B$.
- How to compute the verification component $\mathbf{B}_{x_i+x_j}$ from the public parameters $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell$.
- Finally, show that the verification relation is satisfied: $\mathbf{A}\mathbf{R}_{x_i+x_j} + (x_i + x_j) \cdot \mathbf{G} = \mathbf{B}_{x_i+x_j}$.

[3] It is not too difficult to obtain a many-time homomorphic signature scheme, but for simplicity, we will just consider a one-time signature scheme.

(c) **Homomorphic multiplication.** Let $\mathbf{R}_i, \mathbf{R}_j$ be signatures on bits x_i, x_j where $\|\mathbf{R}_i\|_\infty, \|\mathbf{R}_j\|_\infty \le B$, $\mathbf{AR}_i + x_i \cdot \mathbf{G} = \mathbf{B}_i$, and $\mathbf{AR}_j + x_j \cdot \mathbf{G} = \mathbf{B}_j$. Show how to construct a signature $\mathbf{R}_{x_i x_j}$ on the product $x_i x_j$, where $\left\|\mathbf{R}_{x_i x_j}\right\|_\infty \le (m+1) \cdot B$. Specifically, you should specify the following:

- How to compute the signature $\mathbf{R}_{x_i x_j}$ from the public parameters $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell$, the signatures $\mathbf{R}_i, \mathbf{R}_j$, and the values x_i, x_j Moreover, verify that $\left\|\mathbf{R}_{x_i x_j}\right\|_\infty \le (m+1) \cdot B$.
- How to compute the verification component $\mathbf{B}_{x_i x_j}$ from the public parameters $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_\ell$.
- Finally, show that the verification relation is satisfied: $\mathbf{AR}_{x_i x_j} + (x_i x_j) \cdot \mathbf{G} = \mathbf{B}_{x_i x_j}$.

(d) **[Extra Credit: 3 points.]** Show that an adversary who obtains signatures on *any* pair of distinct messages $x, y \in \{0,1\}^\ell$ can completely break unforgeability. That is, show how such an attacker can forge a signature $\mathbf{R}$ on any message z with respect to any function f, where $\|\mathbf{R}\|_\infty \le 2mB$. Namely, as long as there is some slack in the norm bounds (which there often is), then the adversary is able to forge signatures on arbitrary messages (with respect to any function).

Problem 4: Factoring [10 points]. For this problem, let p and q be distinct large primes. Remember that finding roots of polynomials over the reals is easy.

(a) The best algorithm for factoring an RSA modulus $N = pq$ is the general number field sieve (GNFS). This algorithm runs in time

$$e^{\Omega((\log N)^{1/3} \cdot (\log\log N)^{2/3})}.$$

Compute the smallest integral constant $c \in \mathbb{Z}^+$ such that an RSA modulus of λ^c bits takes time $\Omega(2^\lambda)$ to factor.

Note: The keys for standard elliptic-curve cryptosystems only need to be 2λ bits long to guarantee $\Omega(2^\lambda)$ security against the best known attacks. The answer to this question explains why elliptic-curve cryptosystems are much more popular than RSA today.

(b) Say that you are given an algorithm $\mathcal{A}$ that takes as input $N = pq$ and outputs $p + q$. Show that you can use $\mathcal{A}$ to factor N.

(c) Let f and g be fixed/public polynomials of constant degree with integral coefficients. In an attempt to shorten the size of your RSA secret key, you compute an RSA modulus using the following algorithm:

- Choose a random poly(λ)-bit integer x.
- If $f(x)$ and $g(x)$ are both primes, return $N = f(x) \cdot g(x)$.
- Else, restart.

Give an efficient algorithm that factors such a modulus N in polynomial time.

Problem 5: 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

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