CS 6222: Introduction to Cryptography **Spring 2021**

Homework 5: Cryptographic Protocols

Due: May 7, 2021 at 5pm (Submit on Gradescope) **Instructor:** David Wu

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

`https://www.cs.virginia.edu/dwu4/courses/sp21/static/homework.tex`

You must submit your problem set via Gradescope. Please use course code **D55GP5** to sign up.

Collaboration Policy. You may discuss your general *high-level* strategy with other students, but you may not share any written documents or code. You should not search online for solutions to these problems. If you do consult external sources, you must cite them in your submission. You must include the computing IDs of all of your collaborators with your submission. Refer to the official course policies for the full details.

Problem 1: Schnorr's Protocol and Signatures [15 points].

(a) Suppose in Schnorr's protocol, the challenger sampled the challenge from a set S where $|S| = k$. Show that this protocol has soundness error $1/k$.

Remark: This shows that to achieve negligible soundness error in Schnorr's protocol, the challenge must be samples from a set of super-polynomial size.

(b) Let $\mathbb{G}$ be a cyclic group of order $n = kq$, where $k > 1$ is a small constant that is relatively prime with q (e.g., $k = 8$) and $q \gg k$ is prime. Let g be a generator of $\mathbb{G}$, and let $\hat{g} = g^k \in \mathbb{G}$. In this case, $\hat{g}$ generates a subgroup of $\mathbb{G}$ of prime order q. Consider Schnorr's protocol defined in the prime order group generated by $\hat{g}$ (this coincides with the typical setting in cryptography where we work over a group of prime order). Namely, on input an instance $(\hat{g}, \hat{h} = \hat{g}^x)$, the prover's first message is $\hat{g}^r$ where $r \xleftarrow{R} \mathbb{Z}_q$, the verifier's challenge is $t \xleftarrow{R} \mathbb{Z}_q$, and the prover's response is $r + xt \in \mathbb{Z}_q$. Show that even if $\hat{h} \notin \langle \hat{g} \rangle$, then a (malicious) prover is still able to convince the verifier to accept with probability that is negligibly close to $1/k$ (you can assume that the ratio k/q is negligible). Namely, even if $\hat{h}$ is not in the group generated by $\hat{g}$, the prover can nonetheless convince the verifier that it knows the discrete log of $\hat{h}$ base $\hat{g}$ (even though the discrete log does *not* exist in this case).

Remark: This show that Schnorr's protocol is no longer sound if the base $\hat{g}$ is not a generator of the full group $\mathbb{G}$ and $\mathbb{G}$ has small prime factors. To prevent this attack and recover soundness, the verifier must *additionally* check that $\hat{h}$ is indeed in the group generated by $\hat{g}$. This setting is very common in practice (and should not be glossed over when designing cryptographic systems). For instance, if we consider groups over the integers, we typically work over a prime-order subgroup of $\mathbb{Z}_p^*$ (e.g., the subgroup of prime order q when $p = kq + 1$ for some constant k) or over an elliptic-curve group with order kq and small k (in this case, k is referred to as the "cofactor" of the curve).

(c) We saw in lecture that in the Schnorr signature scheme, signing different messages using the same randomness leaks the secret key. Show that this is also the case for ECDSA.

Remark: This attack was used to break security on the Sony Playstation 3. It turns out that Sony had used ECDSA with a *fixed* string as the randomness to sign software for the PS3. This attack allowed hackers to extract Sony's ECDSA signing key and in turn, issue arbitrary firmware updates for the PS3. Thus, when using ECDSA, it is also critical to derandomize the signing algorithm.

Problem 2: Oblivious Transfer [35 points].

(a) Suppose that in a separate offline phase, a trusted party samples $r_0, r_1 \xleftarrow{\text{R}} \{0,1\}^\ell$ and $b \xleftarrow{\text{R}} \{0,1\}$. It gives (r_0, r_1) to the sender and (b, r_b) to the receiver. The pair of values (r_0, r_1) and (b, r_b) is referred to as an "OT correlation." Show that the sender and the receiver can use their OT correlation to implement a two-round oblivious transfer protocol on ℓ-bit messages. Prove that your scheme is correct and satisfies *perfect sender security* as well as *perfect receiver security*.

(b) Instead of relying on a trusted party to generate the OT correlations in the offline phase, show how the sender and receiver can generate an OT correlation using an OT protocol.

Remark: Observe that this OT protocol is *input-independent*, and thus can be done *before* any protocol execution. Moreover, using a technique called *OT extension*, the sender and the receiver can precompute a large number of OTs with very modest cost.

(c) A "1-out-of-k" OT protocol generalizes OT to the setting where the sender has k messages $m_0, \dots, m_{k-1}$ and the receiver holds an index $i \in \{0, \dots, k-1\}$. At the end of the protocol, the receiver should learn m_i while the sender learns nothing. We can generalize the OT correlation construction from Part (a) to obtain a 1-out-of-k correlation as follows. The trusted party samples $r_0, \dots, r_{k-1} \xleftarrow{\text{R}} \{0,1\}^\ell$ and an index $j \xleftarrow{\text{R}} \{0, \dots, k-1\}$. It gives $(r_0, \dots, r_{k-1})$ to the sender and (j, r_j) to the receiver. Show that the sender and the receiver can use their OT correlation to implement a two-round 1-out-of-k OT protocol on ℓ-bit messages. Prove that your scheme is correct and satisfies perfect sender security and perfect receiver security.

(d) In class, we described a protocol for 1-out-of-2 OT. Show how to construct a 1-out-of-2^t OT protocol (on λ-bit messages) using t invocations of *any* 1-out-of-2 OT protocol (on λ-bit messages). You can use a secure PRF $F \colon \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ in your construction. Justify the correctness, sender security, and receiver security of your construction. You do not need to provide formal proofs of these properties, but your description should appeal to concrete properties of the underlying OT protocol and security of the PRF. **Hint:** Your construction should use the 1-out-of-2 OT protocol as a *black box* (you do need to rely on any implementation details of the underlying protocol). Start by having the sender sample $2t$ independent PRF keys. The sender will use these keys to blind the messages $m_0, \dots, m_{2^t-1}$.

Problem 3: Regev Encryption [25 points]. In lecture, we described Regev encryption in the setting where the message is encoded in the *most significant bits* of the ciphertext. Here, we will consider a variant where the message is encoded in the *least significant bits* of the ciphertext. For simplicity, we will just consider the symmetric setting (but everything generalizes to the public-key setting in the manner described in lecture). Let the message space be $\mathbb{Z}_p$, n be the lattice dimension, q be the modulus, and χ be the error distribution. Suppose that $\gcd(p, q) = 1$. Note that p and q need *not* be prime here.

- Setup: Sample and output $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$.

- $\mathsf{Encrypt}(\mathbf{s}, m)$: On input a secret key $\mathbf{s} \in \mathbb{Z}_q^n$ and a message $m \in \mathbb{Z}_p$, sample $\mathbf{a} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ and $e \leftarrow \chi$, and output the ciphertext $\mathsf{ct} = (\mathbf{a}, \mathbf{s}^{\mathsf{T}}\mathbf{a} + pe + m)$.

(a) Show how to define $\mathsf{Decrypt}(\mathbf{s}, \mathsf{ct})$. Prove the correctness of the resulting scheme. You may assume that $\Pr[e \leftarrow \chi : |e| < q/(2p) - 1] = 1$.

(b) Show that under the $\mathsf{LWE}_{n,q,\chi}$ assumption, the above encryption scheme is semantically secure. (Technically, it is CPA-secure under the LWE assumption, but you do not have to show this.)

Problem 4: Time Spent [3 extra credit points]. How long did you spend on this problem set? This is for calibration purposes, and the response you provide does not affect your score.

Optional Feedback. Please answer the following *optional* questions to help us design future problem sets. You do not need to answer these questions. 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?