CS 388H: Cryptography **Fall 2021**

Take-Home Final Exam

Due: December 12, 2021 at 5:00pm (Submit on Gradescope) **Instructor:** David Wu

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

https://www.cs.utexas.edu/~dwu4/courses/fa21/static/homework.tex

You must submit your completed exam via Gradescope (accessible through Canvas).

Collaboration Policy. This is an *individual* assignment. You are not allowed to collaborate with anyone on this problems and you are not permitted to search online for solutions to these problems. If you do consult external sources (these cannot include solutions), you must cite them in your submission.

1 Part I: Conceptual Questions

Problem 1: Conceptual Questions [35 points]. You do **not** need to provide any justification for any part of this question. For the multiple choice questions, there could be **multiple** answers or **zero** correct answers. For full credit, you should select **all** correct responses, or indicate that there are **none**.

1. **Pseudorandom functions.** Let $F\colon \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ be any secure PRF. Which (if any) of the following functions is always a secure PRF (so long as F is secure)?

   (a) $F'((k_1,k_2),x) := \begin{cases} F(k_1, x \oplus k_2) & x \neq k_2 \\ k_1 & x = k_2. \end{cases}$

   (b) $F'(k,x) := F(k,x) \oplus F(k,F(k,x))$.

   (c) $F'(k,x) := F(k,x) \oplus F(x,k)$.

   (d) $F'(k,x) := F(F(k,x),0^\lambda)$

2. **Symmetric encryption.** Let p be a prime. Consider the following symmetric encryption scheme with message space $\mathbb{Z}_p^*$. The secret key is a random integer $k \xleftarrow{R} \mathbb{Z}_p^*$. To encrypt a message $x \in \mathbb{Z}_p^*$, sample a random $r \xleftarrow{R} \mathbb{Z}_p^*$ and output $(r, rx+k)$. To decrypt a ciphertext (y,z), the decrypter computes $y^{-1}(z-k) \bmod p$. Which (if any) of the following statements are true about this encryption scheme?

   (a) The scheme is semantically secure.

   (b) The scheme is CPA-secure.

   (c) The scheme is CCA-secure.

   (d) The scheme is an authenticated encryption scheme.

3. **Diffie-Hellman assumptions.** Let $\mathbb{G}$ be a group of prime-order p and generator g. Suppose the CDH problem is hard in $\mathbb{G}$. Which (if any) of the following problems are hard in $\mathbb{G}$?

   (a) Given g^x where $x \xleftarrow{R} \mathbb{Z}_p$, compute $x^{-1} \bmod p$.

(b) Given $h \xleftarrow{R} \mathbb{G}$, find $z \in \mathbb{G}$ such that $z^3 = h$.

(c) Given g^x where $x \xleftarrow{R} \mathbb{Z}_p$, compute g^{3x+17}.

(d) Given g^x where $x \xleftarrow{R} \mathbb{Z}_p$, compute g^{x^2+5}.

4. **Assumptions needed for cryptography.** Which (if any) of the following cryptographic primitives exist in a world where P = NP?

   (a) A secure key-agreement protocol.

   (b) A zero-knowledge proof system for NP.

   (c) A k-time MAC (i.e., where security holds against an adversary that gets to see k message-tag pairs), where $k > 0$ is a fixed constant.

   (d) A CPA-secure symmetric encryption scheme with key-space $\{0,1\}^{10n}$, message-space $\{0,1\}^n$, and ciphertext space $\{0,1\}^{2n}$.

5. **Black-box separations.** Which (if any) of the following would resolve the P vs. NP question (i.e., would imply either P = NP *or* P ≠ NP).

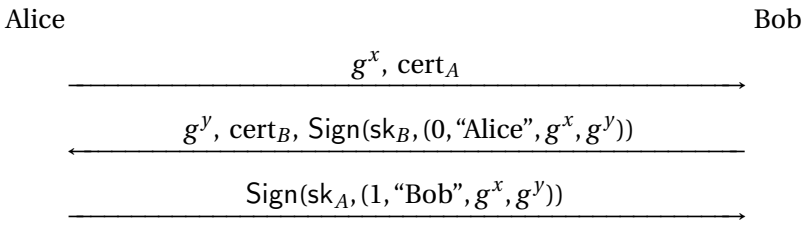
   (a) A construction of secure key-agreement from the discrete log assumption.

   (b) A black-box construction of secure key-agreement from a public-key encryption scheme.

   (c) A black-box construction of secure key-agreement from a digital signature scheme.

   (d) A black-box construction of secure key-agreement from a pseudorandom function.

6. **Authenticated key exchange.** Consider the following Diffie-Hellman-based protocol for mutual authentication. We assume that Alice and Bob each possess a certificate cert_A and cert_B. Alice's certificate will identify her name "Alice" as well as her verification key pk_A. Likewise, Bob's certificate cert_B will identify his name "Bob" and his verification key pk_B. We work in a group $\mathbb{G}$ of prime order p and generator g (where DDH is believed to hold). As usual, let $H\colon \mathbb{G} \to \{0,1\}^n$ be a key-derivation function. During the protocol, Alice samples $x \xleftarrow{R} \mathbb{Z}_p$ and Bob samples $y \xleftarrow{R} \mathbb{Z}_p$. The protocol proceeds as follows:



In the above protocol, Alice will check that Bob provides a valid signature on her identity "Alice" as well as the ephemeral Diffie-Hellman elements (g^x, g^y). If so, she completes the protocol with output $k \leftarrow H(g, g^x, g^y, g^{xy})$ and identity "Bob." Similarly, Bob will check that he receives a signature containing his identity "Bob" as well as the ephemeral Diffie-Hellman elements. If so, he completes the protocol with output $k \leftarrow H(g, g^x, g^y, g^{xy})$ and identity "Alice." It can be shown that the above protocol is a secure mutual authentication protocol. For each of the protocol variants described below, identify whether it (a) is vulnerable to a key-recovery attack (i.e., *either* Alice or Bob completes the protocol with a key k and some identity $\mathsf{id} \neq \mathcal{A}$, and the active network adversary $\mathcal{A}$ knows k); (b) is vulnerable to an identity misbinding attack; (c) is vulnerable to a replay attack; or (d) remains secure. If multiple attacks apply, select the first applicable option (i.e., (a) > (b) > (c) > (d)).

(6.1) *Each party signs their own name instead:* namely, Alice signs $(1, \text{“Alice”}, g^x, g^y)$ and Bob signs $(0, \text{“Bob”}, g^x, g^y)$.

(6.2) *The certificates are signed instead of the identities:* namely, Alice signs $(1, \text{cert}_B, g^x, g^y)$ and Bob signs $(0, \text{cert}_A, g^x, g^y)$.

(6.3) *The 0/1 values are not signed:* namely, Alice signs $(\text{“Bob”}, g^x, g^y)$ and Bob signs $(\text{“Alice”}, g^x, g^y)$.

(6.4) *Bob does not sign the ephemeral Diffie-Hellman values:* namely Bob signs $(0, \text{“Alice”})$.

7. **Schnorr's protocol.** In the following, let $\mathbb{G}$ be a prime-order group. Which (if any) of the following statements are true about Schnorr's protocol for proving knowledge of discrete log in $\mathbb{G}$?

   (a) If discrete log is easy in $\mathbb{G}$, then Schnorr's protocol is no longer complete.

   (b) If discrete log is easy in $\mathbb{G}$, then Schnorr's protocol is no longer a proof of knowledge.

   (c) If discrete log is easy in $\mathbb{G}$, then Schnorr's protocol is no longer honest-verifier zero-knowledge.

2 Part II: Cryptographic Primitives and Constructions

Instructions. Answer any **two** of the three problems in this section. If you answer more than two problems, only the first two you answer will be graded.

Problem 2: Proxy Re-Encryption [25 points]. Let $\mathbb{G}$ be a group of prime order p and generator g. Recall the vanilla ElGamal encryption scheme from class: the public key is a pair of group elements $\mathsf{pk} = (g, h)$ and the secret key is the exponent $\mathsf{sk} = x$ where $h = g^x$; an encryption of $m \in \mathbb{G}$ is the pair $(g^r, h^r \cdot m)$.

Let $(\mathsf{pk}_{\mathsf{Alice}}, \mathsf{sk}_{\mathsf{Alice}})$ be Alice's ElGamal key-pair and $(\mathsf{pk}_{\mathsf{Bob}}, \mathsf{sk}_{\mathsf{Bob}})$ be Bob's ElGamal key-pair. Both Alice and Bob give their public keys to an email server. When the email server receives mail for Alice encrypted under $\mathsf{pk}_{\mathsf{Alice}}$, it forwards it to Alice, and correspondingly with Bob. The mail server does not know $\mathsf{sk}_{\mathsf{Alice}}$ or $\mathsf{sk}_{\mathsf{Bob}}$, so it cannot decrypt Alice's or Bob's emails.

(a) Suppose Alice goes on vacation, and she wants to delegate her email responsibilities to Bob. Show that Alice and Bob can compute a "proxy key" $\mathsf{sk}_{\mathsf{proxy}}$ that allows the mail server to translate a ciphertext ct encrypted under $\mathsf{pk}_{\mathsf{Alice}}$ to a new ciphertext ct' that encrypts the *same* message under Bob's public key $\mathsf{pk}_{\mathsf{Bob}}$. Moreover, assuming semantic security of the ElGamal encryption scheme, messages encrypted under $\mathsf{pk}_{\mathsf{Alice}}$ should remain semantically secure against the proxy even given knowledge of $\mathsf{sk}_{\mathsf{proxy}}$ together with Alice and Bob's public keys. The proxy key can depend on $\mathsf{sk}_{\mathsf{Alice}}$ and $\mathsf{sk}_{\mathsf{Bob}}$.

Prove that your construction satisfies correctness and semantic security (assuming correctness and semantic security of ElGamal encryption). For correctness, you should show that if ct decrypts to m under $\mathsf{sk}_{\mathsf{Alice}}$, then the translated ciphertext ct' decrypts to m under $\mathsf{sk}_{\mathsf{Bob}}$. In the semantic security game, you may assume that the adversary is given $\mathsf{pk}_{\mathsf{Alice}}$, $\mathsf{pk}_{\mathsf{Bob}}$, and $\mathsf{sk}_{\mathsf{proxy}}$ and is trying to distinguish encryptions of $m_0 \in \mathbb{G}$ from encryptions of $m_1 \in \mathbb{G}$ under $\mathsf{pk}_{\mathsf{Alice}}$.

(b) To ensure that the mail server is behaving honestly, we require the mail server include a NIZK proof that it is translating the ciphertexts properly. The NIZK proof would be logged and independently audited afterwards. Using Fiat-Shamir, it suffices to construct a Σ-protocol for the "correct" translation procedure from Part (a). Show how to construct this Σ-protocol, and prove completeness, special

soundness, and HVZK of your protocol. Recall also that the prover must be efficient in a Σ-protocol. You cannot use a general-purpose zero-knowledge proof for NP languages here.

In this setting, the statement contains the public keys $\mathsf{pk}_{\mathsf{Alice}}$, $\mathsf{pk}_{\mathsf{Bob}}$ and the ciphertexts $\mathsf{ct}_{\mathsf{Alice}}$, $\mathsf{ct}_{\mathsf{Bob}}$. A statement $(\mathsf{pk}_{\mathsf{Alice}}, \mathsf{pk}_{\mathsf{Bob}}, \mathsf{ct}_{\mathsf{Alice}}, \mathsf{ct}_{\mathsf{Bob}})$ is true if $\mathsf{ct}_{\mathsf{Alice}}$ and $\mathsf{ct}_{\mathsf{Bob}}$ encrypt identical messages under $\mathsf{pk}_{\mathsf{Alice}}$ and $\mathsf{pk}_{\mathsf{Bob}}$, respectively. Note that the proxy does *not* know $\mathsf{sk}_{\mathsf{Alice}}$, $\mathsf{sk}_{\mathsf{Bob}}$, or the message. **Hint:** One approach is to reduce this problem to one we encountered in class or on a previous homework assignment. If you do this, you do *not* have to re-prove all of the required properties.

Problem 3: DSKS Attacks on RSA-FDH [25 points]. Recall the RSA-FDH signature scheme from class with message space $\{0,1\}^t$ and a hash function $H\colon \{0,1\}^t \to \mathbb{Z}$. Suppose (m,σ) is a valid message-signature pair under a verification key $\mathsf{vk} = (N,e)$: namely, $\sigma^e = H(m) \bmod N$. In this problem, we will show that an adversary who intercepts (m,σ) can craft a new verification key $\mathsf{vk}' = (N',e')$ such that σ is a valid signature on m with respect to vk': namely, that $\sigma^{(e')} = H(m) \bmod N'$, where $N' = p'q'$ is a valid RSA moduli and $\gcd\big(e', \varphi(N')\big) = 1$. These types of attacks are called *duplicate signature key selection* (DSKS) attacks. In this problem, we consider a restricted setting that illustrates this attack.

(a) Let $\mathbb{G}$ be a group of order 2^ℓ for some known integer ℓ. Show that given any $g, h \in \mathbb{G}$ where $h = g^x$ for some $x \in \mathbb{Z}_{2^\ell}$, there is an algorithm that computes x in time $\mathrm{poly}(\ell)$.

(b) Suppose $p' = 2^{\ell_p} + 1$ and $q' = 2^{\ell_q} + 1$ are primes and $N' = p'q' > N$. Given a message $m \in \{0,1\}^n$ and a signature $\sigma \in [0, N-1]$, your goal is to find e' such that $\sigma^{e'} = H(m) \bmod N'$ and $\gcd(e', \varphi(N')) = 1$. In this case, σ is a valid signature on message m under the key $\mathsf{vk}' = (N', e')$. State a sufficient condition on N', σ, m under which you can efficiently compute e' satisfying $\gcd\big(e', \varphi(N')\big) = 1$ using the algorithm from Part (a). Assuming your condition holds, show how to use your algorithm from Part (a) to compute the verification key $\mathsf{vk}' = (N', e')$.

Remark: While the specific condition you identified here may not hold with high probability (over a random choice of the primes p' and q'), these ideas readily generalize to yield polynomial-time DSKS attacks on RSA-FDH signatures that succeed with high probability.

Remark: DSKS attacks on signature schemes can be problematic for a number of reasons. For example, imagine that Alice submits a signed request (m,σ) to her bank to deposit a check into her account (i.e., the account associated with her signature verification key vk). If the underlying signature scheme is vulnerable to a DSKS attack, then an adversary can intercept Alice's request and register a new verification key vk' under their name for which (m,σ) is also a valid message-signature pair. The adversary can then present their verification key vk' together with (m,σ) to the bank. Because (m,σ) is valid under both vk and vk', the bank cannot tell who originated the request! A common (and recommended) approach to defend against such DSKS attacks is to sign the message *together* with the verification key.

Problem 4: Cryptographic Hash Functions [25 points]. Let $H\colon \{0,1\}^{2\lambda} \to \{0,1\}^\lambda$ be a collision-resistant hash function.

(a) Show that H is a one-way function. Remember to compute the advantage of the adversary you construct in your security reduction.

(b) Use H to construct a function f that is one-way but *not* collision resistant.

Remark: This shows that collision-resistance and one-wayness are *not* equivalent notions. In fact, there is a black-box separation of collision-resistant hash functions from one-way functions.

(c) Use H to construct a new hash function H' with domain $\{0,1\}^{2\lambda}$ and range $\{0,1\}^m$ (for some $m < 2\lambda$) such that the following two properties hold: (1) if H is collision-resistant, then so is H'; and (2) the function $H''\colon \{0,1\}^{2\lambda} \to \{0,1\}^{m-1}$, where $H''(x)$ outputs the first $m-1$ bits of $H'(x)$, is *not* collision-resistant. Prove that your construction H' satisfies both properties.

Remark: This shows that dropping even a single bit of the output of a collision-resistant hash function can break collision resistance.

(d) In many practical applications, we model a cryptographic hash function as a random oracle. It is easy to show that the function $F(k,x) := H(k\|x)$ is a secure PRF when H is modeled as a random oracle. Suppose that $H'\colon \{0,1\}^{\leq(n+1)\lambda} \to \{0,1\}^{\lambda}$ is a Merkle-Damgård hash function built from a secure compression function $h\colon \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$. Define the variable-domain PRF $F\colon \{0,1\}^{\lambda} \times \{0,1\}^{\leq n\lambda} \to \{0,1\}^{\lambda}$ to be $F(k,x) := H'(k\|x)$. Sketch an attack on this PRF and analyze its advantage. You do *not* need to give a formal specification of your algorithm, but your description should include all of the key details to reconstruct the actual algorithm if needed. Here, we are *not* modeling H' as a random oracle, and indeed, your attack demonstrates why modeling H' as a random oracle is inappropriate in this setting.

Remark: This shows that it is unreasonable to model Merkel-Damgård hash functions (e.g., SHA-256) as a random oracle on *variable-size* domains. The recommended practice for instantiating random oracles in practice is to use HMAC with a fixed-key (e.g., the all-zeroes key).

Optional Feedback. If you have any suggestions for improving future iterations of the course, please feel free to share your thoughts here!