CS255: Cryptography and Computer Security **Winter 2022**

Take-home Final Exam

Instructions:

- **Please answer all five questions. You have three hours.**
- You may take the exam at any time during the exam window. You have three hours from the moment you begin until the moment you submit your answers on Gradescope.
- The exam is open book, open notes, open laptops, and open Internet (e.g., to consult a static online resource). However, you are expected to do the exam on your own. You may not interact, collaborate, or discuss the exam with another person during the exam window.
- To submit your answers please either (i) use the provided LaTeX template, or (ii) print out the exam and write your answers in the provided spaces, or (iii) write your answers on blank sheets of paper, but please make sure to start each question on a new page. When done, please upload your solutions to Gradescope (course code `4PEBZ3`).
- The LaTeX template for the final is available here. Please do not share the link with others.
- Students are bound by the Stanford honor code. In particular, you are expected to do the exam on your own.

Problem 1. (24 points) Questions from all over.

a. You are given a secure MAC scheme (S, V) that can be used to sign a sequence of bytes whose length is a multiple of eight. That is, its length can be 8 bytes, 16 bytes, 24 bytes, etc. Explain how to use this MAC scheme to sign a sequence of bytes of arbitrary length (i.e., including messages whose length may not be a multiple of eight).

Your answer:

b. A recent paper shows that the implementation of AES-GCM on a widely used phone can end up using the same (key,nonce) pair many times. Recall that AES-GCM is built from randomized counter mode encryption (rCTR). Suppose an attacker has a message $m \in \{0,1\}^{\ell}$ and has its rCTR encryption $ct := (IV, c)$. The attacker also has another rCTR ciphertext $ct' := (IV, c')$ of some unknown message m', where ct$'$ is constructed with the same IV and key as ct. Moreover, c and c' are the same length. Explain how the attacker can decrypt ct' using the data at its disposal.

Your answer: $m' =$

c. Suppose Alice has a password pwd and this password is also known to server Bob. When Alice connects to the server, the server uses a MAC-based challenge-response identification protocol to authenticate Alice. Show that an attacker who eavesdrops on network traffic between Alice and the server can mount a dictionary attack to recover Alice's password. You may assume Alice's password is chosen from a dictionary PWD so that $pwd \in PWD$.

Your answer:

d. For a prime p, consider the following PRG defined over $(\mathbb{Z}_p, \mathbb{Z}_p^2)$: given an input $x \in \mathbb{Z}_p$, the PRG outputs $G(x) := (x^2, x^3) \in \mathbb{Z}_p^2$. Is this PRG secure? If so, explain why. If not, describe an attack.

Your answer:

e. Give an example of a function f that is believed to be a one-way function, but is trivially not collision resistant. You can assume that SHA256 is a one-way function. Make sure to explain why your function is one way and why it is easy to find a collision for it.

Your answer:

f. Let p be a prime with $p \equiv 2 \bmod 3$. Show an efficient algorithm that takes $\alpha \in \mathbb{Z}_p^*$ as input and outputs the cube root of α in $\mathbb{Z}_p$. That is, show how to efficiently solve the equation $x^3 - \alpha = 0$ in $\mathbb{Z}_p$. **Hint:** Show that $\sigma := \alpha^{(2p-1)/3}$ in $\mathbb{Z}_p$ is the cube root of α.

Your answer:

Problem 2. (20 points) A two-query PRF.

a. For a prime $p \geq 5$ consider the following PRF defined over $(\mathbb{Z}_p^2, \mathbb{Z}_p, \mathbb{Z}_p)$:

$$F\big((k_1, k_2), x\big) := k_1 x + k_2.$$

Show that this PRF is not secure: construct an adversary $\mathcal{A}$ that distinguishes this PRF from a random function from $\mathbb{Z}_p$ to $\mathbb{Z}_p$.

Your answer:

b. One can show that the PRF from part (a) is secure against a PRF adversary that can issue at most two evaluation queries. We say that F is a 2-time secure PRF.

More generally, let F be a PRF defined over $(\mathcal{K}, \mathcal{M}, \mathcal{T})$. Recall that in class we defined a MAC scheme (S, V) derived from the PRF F as follows: $S(k, m) := F(k, m)$ and $V(k, m, t)$ accepts if $t = F(k, m)$. We showed that if the range $\mathcal{T}$ of the PRF is sufficiently large and F is a secure PRF, then (S, V) is a secure MAC scheme.

Suppose that F is not a secure PRF, but is 2-time secure, as is the F from part (a). Show that the derived MAC scheme (S, V) is a secure one-time MAC, meaning that it is secure if the adversary can issue at most a *single* chosen message query.

Hint: As usual, prove the contra-positive: Let $\mathcal{A}$ be a MAC adversary that succeeds in forging a message-tag pair after issuing only one chosen message query to the MAC challenger. Show how to use $\mathcal{A}$ to build a PRF adversary $\mathcal{B}$ that breaks the PRF using at most two queries. Your argument shows that the PRF from part (a) gives a very efficient one-time MAC.

Your answer:

Problem 3. (20 points) Attacks on Schnorr signatures.

Let $\mathbb{G}$ be a finite cyclic group of order q with generator $g \in \mathbb{G}$. Let $H : \mathcal{M} \times \mathbb{G} \to \mathbb{Z}_q$ be a hash function. Recall that a Schnorr secret key is a random $\text{sk} := \alpha \xleftarrow{R} \mathbb{Z}_q$, and the public key $\text{pk} := h \leftarrow g^\alpha \in \mathbb{G}$. A (non-optimized) signature is generated by choosing a random $\rho \xleftarrow{R} \mathbb{Z}_q$, computing

$$R \leftarrow g^\rho \in \mathbb{G}, \qquad c \leftarrow H(m, R) \in \mathbb{Z}_q, \qquad z \leftarrow \rho + c \cdot \alpha \in \mathbb{Z}_q,$$

and outputting $\sigma := (R, z)$. The verification algorithm $V\big(\text{pk}, m, (R, z)\big)$ computes $c \leftarrow H(m, R)$ and accepts if $g^z = R \cdot h^c$, where $h = \text{pk} = g^\alpha$.

a. Faulty hashing. Suppose that during signing and verification the challenge c is computed as $c \leftarrow H(m)$ instead of $c \leftarrow H(m, R)$. Show that the resulting signature scheme is insecure: an adversary who has pk can forge a signature on any message $m \in \mathcal{M}$.

Your answer:

b. Faulty randomness. Let (sk, pk) be a key pair for the Schnorr signature scheme. Suppose the signing algorithm is faulty and chooses *dependent* values for R in consecutively issued signatures. In particular, when signing a message m_0 the signing algorithm chooses a uniformly random ρ_0 in $\mathbb{Z}_q$, as required. However, when signing the next message m_1 it chooses ρ_1 as $\rho_1 \leftarrow a \cdot \rho_0 + b \in \mathbb{Z}_q$ for some publicly known $a, b \in \mathbb{Z}_q$. Show that if the adversary obtains the corresponding Schnorr message-signature pairs (m_0, σ_0) and (m_1, σ_1) and knows a, b and pk, it can learn the secret signing key sk, with high probability. Recall that $\sigma_0 = (R_0, z_0)$ and $\sigma_1 = (R_1, z_1)$.

Hint: build a system of two linear equations in two variables, α and ρ_0.

Your answer:

c. Related keys. Let $h := g^{\alpha} \in \mathbb{G}$ be a random Schnorr public key. Define n related Schnorr public keys $\mathrm{pk}_1, \ldots, \mathrm{pk}_n$ by setting $\mathrm{pk}_i := h \cdot g^i \in \mathbb{G}$ for $i = 1, \ldots, n$. Show that this scheme is insecure with respect to the public keys $\mathrm{pk}_1, \ldots, \mathrm{pk}_n$. In particular, show that a signature (R, z) on a message m with respect to pk_j, lets the adversary construct a signature on the same message m with respect to pk_i for some $i \neq j$. This is an existential forgery on pk_i.

Your answer:

Discussion: Some Bitcoin wallets (called HD wallets) use a technique related to the one in part (c) to generate signature key pairs. They defend against your attack by including the public key pk_i as part of the input to the hash function H. That is, they compute $c \in \mathbb{Z}_q$ as $c \leftarrow H(\mathrm{pk}_i, m, R)$.

Problem 4. (18 points) Birthday paradox

a. Let $x_1, \ldots, x_n$ be randomly sampled integers in the range $[1, B]$. The birthday paradox says that when $n \geq 1.2\sqrt{B}$, the probability that there is a collision (i.e. exists $i \neq j$ such that $x_i = x_j$) is at least $1/2$. How large must n be, as a function of k and B, so that the expected number of collisions is at least k, for some given $k > 1$?
Hint: For $1 \leq i, j \leq n$ define the indicator random variable $I_{i,j}$ to be 1 if $x_i = x_j$ and zero otherwise. Then the expected number of collisions is $\sum_{i \neq j} E[I_{i,j}]$. Moreover, $E[I_{i,j}] = 1/B$ for $i \neq j$. You may use the approximation $n(n-1) \approx n^2$.

Your answer:

b. A three way collision is a triple of distinct i, j, ℓ such that $x_i = x_j = x_\ell$. How large must n be, as a function of B, so that there is at least one 3-way collision in expectation?
Hint: Use a similar approach as in part (a). For $1 \leq i, j, \ell \leq n$ define the indicator random variable $I_{i,j,\ell}$ to be 1 if $x_i = x_j = x_\ell$ and zero otherwise. Then the expected number of 3-way collisions is $\sum_{i \neq j \neq \ell \neq i} E[I_{i,j,\ell}]$. You may use the approximation $n(n-1)(n-2) \approx n^3$.

Your answer:

Problem 5. (18 points) Meet in the middle attacks

a. Let $\mathcal{E} := (E, D)$ be a cipher defined over $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ where $\mathcal{C} \subseteq \mathcal{M}$. One can define a cipher with double the key length, called $2\mathcal{E}$, defined over $(\mathcal{K}^2, \mathcal{M}, \mathcal{C})$ as follows:

$$2E\big((k_1, k_2), m\big) := E\big(k_1, E(k_2, m)\big).$$

That is, we apply the encryption algorithm E twice with independent keys k_1 and k_2. Write out the decryption algorithm:

Your answer: $2D\big((k_1, k_2), c\big) :=$

b. Now, recall that the AES block cipher can take either 128, 192, or 256 bit keys, denoted AES128, AES192, and AES256, respectively. You are probably wondering why is there a need for AES256. We can simply define AES256 to be 2AES128, namely define

$$\text{BadAES256}\big((k_1, k_2), m\big) := \text{2AES128}\big((k_1, k_2), m\big) = \text{AES128}\big(k_1, \text{AES128}(k_2, m)\big).$$

The key for this BadAES256 is (k_1, k_2) which is 256 bits, as required. So why is AES256 a separate algorithm? Why can't we simply use 2AES128?

Let us first compare the running time of 2AES128 with the running time of the real AES256. How many rounds of the AES round function are used in AES256? How many are used in 2AES128?

Your answer:
Num. rounds in AES256 = **; Num. rounds in 2AES128 =**

c. Ok, so AES256 is faster than 2AES128. What about the security of 2AES128? Let us show that the $2\mathcal{E}$ cipher is no more secure than the underlying $\mathcal{E}$ cipher. This means that $2AES128$ is no more secure than AES128, which is not what we want for AES256.

For a multi-block message $M = (m_1, m_2, \ldots, m_n)$ write $E(k, M) := \big(E(k, m_1), \ldots, E(k, m_n)\big)$. You are given a pair (M, C) where $C = 2E\big((k_1, k_2), M\big)$. You may assume that there is a *unique* (k_1, k_2) that satisfies $C = 2E\big((k_1, k_2), M\big)$. Your goal is to find this (k_1, k_2).

An exhaustive search algorithm over all possible (k_1, k_2) will take time $|\mathcal{K}|^2$. Your goal is to design an algorithm that finds (k_1, k_2) in time $O(|\mathcal{K}|)$. This is the time to break $\mathcal{E}$ which means that $2\mathcal{E}$ is no more secure than $\mathcal{E}$.

Hint: observe that if $C = 2E\big((k_1, k_2), M\big)$ then

$$E(k_2, M) = D(k_1, C) \tag{1}$$

Try building a table T of pairs $\big(k,\ E(k, M)\big)$ for all $k \in \mathcal{K}$. This takes $|\mathcal{K}|$ evaluations of $E(k, M)$, and the table will contain all possible values of the left-hand side of (1). Now that T is built, show that you can find (k_1, k_2) in time $O(|\mathcal{K}|)$. To do so, use the right-hand side of (1). You can assume that testing if T contains a pair $(*, c)$ can be done in constant time.

Your answer: **Given M and $C = 2E\big((k_1, k_2), M\big)$ the algorithm does:**

d. The algorithm from part (c) is called a *meet in the middle attack* because you are attacking the mid-point of algorithm $2\mathcal{E}$. Meet in the middle attacks come up often. Let us show a meet in the middle attack on the discrete logarithm problem. Let $\mathbb{G}$ be a cyclic group of prime order q with generator $g \in \mathbb{G}$. Let $h = g^\alpha$ for some $\alpha \in \mathbb{Z}_q$. Your goal is to design an algorithm that finds α in time $O(\sqrt{q})$ using a meet in the middle attack.
Hint: Let $Q := \lceil \sqrt{q} \rceil$. We can write α in base Q so that $\alpha = \alpha_1 Q + \alpha_2$ where $0 \le \alpha_1, \alpha_2 < Q$. Then $g^\alpha = h$ can be re-written as

$$(g^Q)^{\alpha_1} = h / g^{\alpha_2}.$$

Use the same strategy as in part (c) to find (α_1, α_2) using a table T and a total of about $2Q$ multiplications in $\mathbb{G}$. The table T will contain pairs $\big(\gamma,\ (g^Q)^\gamma\big)$ for all $0 \le \gamma < Q$.

Your answer: **Given g and $h = g^{\alpha_1 Q + \alpha_2}$ the algorithm does:**