Massachusetts Institute of Technology
6.857: Computer and Network Security
Professors Ronald L. Rivest and Yael Tauman Kalai

Handout 5
April 05, 2021
Due: April 21, 2020

Take–Home Quiz

- This quiz is due on *Wednesday, April 21, 2021* at **11:59 PM**.
- The number of points allocated for each problem is a rough estimate in minutes of how long the problem will take to solve in an in-class exam for someone who was well prepared. (They will probably take longer on a take–home exam if you "study" as you work through them.)

Problem 1	40 points
Problem 2	40 points
Problem 3	30 points
Problem 4	45 points
Problem 5	35 points
Problem 6	30 points
Problem 7	30 points
Problem 8	30 points
Problem 9	20 points
Total	300 points

- Please submit your problem set, in PDF format, on Gradescope. A LaTeX template of the problem set is provided. Include all problems in the same PDF file. You do not need to show code you may have used.
- You are to work on this problem set **individually**.
- This problem set is **open notes**. You may use the lecture notes posted to the course website, and any notes that you took yourself during class. You may also use any other resources online or otherwise, except for other students. Consulting office hours and private posts on Piazza is also allowed.
- Corrections, if any, will be announced on Piazza.

Problem 1. True or False.

Determine whether the following statements are true or false, and **briefly justify your answer, supplying an example or counterexample if appropriate.**

True **False** U.S. citizenship of all team members was required for team to submit a hash function design to NIST to be considered for any of the U.S. Digital Signature standards SHA-1, SHA-2, or SHA-3.

True **False** When using SHA-3 ("sponge construction") as a pseudo-random number generator, it is important that at any point in time the number of truly random bits "absorbed" into the state not be less than the number of pseudo-random bits "squeezed out" of the state.

True **False** The smallest non-prime number p satisfying the Fermat test for primality $a^{p-1} = 1 \pmod{p}$ for $a = 2$ has three decimal digits.

True **False** A symmetric encryption method can not be CCA-secure if it is "self-inverse" (the encryption operation and the decryption operation are the same function).

True **False** With Shamir's secret-sharing method, the secret to be shared is used as the constant term of the secret-sharing polynomial $c_0 + c_1 x + c_2 x^2 \ldots$. That is, c_0 is set to s and the other coefficients are randomly chosen. If instead the coefficient c_1 is set to the secret s and the other coefficients are chosen randomly, the modified scheme is still secure.

True **False** A public-key encryption method is *re-randomizable* if the encryption operation $c = \texttt{Enc}(\texttt{PK}, m, r)$ takes as input not only the public key `PK` and the message m, but also a random value r, and furthermore anyone knowing c can "re-randomize" it to a new, distinct ciphertext for m, e.g., to $c' = \texttt{Enc}(\texttt{PK}, m, r')$, *without knowing* the secret key `SK` or the randomness value r.

True or False: A CPA-secure public-key encryption method can not be re-randomizable. (It is OK to make reasonable computational assumptions for this question.)

True **False** If you have a choice between a security mechanism that provides *prevention* of security violations, and one that provides merely *detection* of security policy violations, you should **always** choose the one providing prevention, assuming that both methods are fool-proof at what they do.

True **False** When signing a message in RSA with "Hash and Sign," suppose you replace the hash function with a `simpleSum` algorithm, which instead

splits a message into 64-bit sections $M = m_0...m_i$, and adds them all together modulo 2^{64}. Claim: the new scheme is secure against an ACMA attack ("adaptive chosen message attack") if the previous one was.

Problem 2. One-Time Pad (OTP).

Non-committing Encryption

While working on this quiz, your TAs, being security conscious, decide to encrypt all their messages using the one time pad (OTP). Specifically, they use the Vernam cipher, a version on the one time pad that uses the English alphabet as its set of symbols. (That is, working modulo 26, with $A = 0$, $B = 1$, ..., $Z = 25$.[1])

First, Billy, Deep, and Andrés meet to agree on a set of secret keys chosen independently and uniformly at random to be used for encrypting their messages. Once they have a shared set of keys, they go home and communicate about the quiz by sending their ciphertexts over email.

Billy sends the first message about the quiz, taking the first key $k_1 =$ `GHTKAIS` and producing the ciphertext $c_1 =$ `ZVHOAAQ`, which he emails to Deep and Andrés.

After receiving and decrypting Billy's email, Deep takes the second key $k_2 =$ `SHRBGLA` and uses it to produce the ciphertext $c_2 =$ `LVFIGCD`, which he sends as a reply to Billy and Andrés.

Finally, Andrés, after seeing both Billy and Deep's opinions on the quiz, crafts his own ciphertext $c_3 =$ `BSNMXTU` using $k_3 =$ `IEZBJGO`. However, Andrés accidentally forwards the email chain with c_1, c_2, and c_3 (but not the keys) to the whole class instead of just replying to Deep and Billy!

Being understandably curious after receiving an email from their TAs with the subject line "quiz difficulty," students in the class asked to know what the plaintexts of these emails are. The quiz is still being written though, and the TAs don't want to make students nervous with their early opinions. They could refuse to decrypt the messages, but that also seems a bit incriminating.

(a) Can Billy, Deep, and Andrés produce *different* keys k_1', k_2', and k_3' such that each of c_1, c_2, and c_3 appears to decrypt to the message `PERFECT`?

(b) Based off the original keys, what did each of Billy, Deep, and Andrés actually think of the quiz?

'Unlucky' Keys?

Alice and Bob are using the one time pad (Vernam edition) to encrypt their messages. They have been vary careful to implement the scheme correctly, and are encrypting their messages with single-use shared keys that are the same length as the messages and have been selected uniformly at random. That is, if Alice is encrypting a n-letter message m to Bob, she produces her ciphertext c as $c = m +_{26} k$, where k is an n-letter string chosen earlier uniformly at random by Alice and shared securely with Bob and $+_{26}$ denotes addition modulo 26 (using the symbol set to represent residues modulo 26). After encrypting her message,

[1] An online tool for producing and decrypting Vernam ciphertexts can be found here: https://www.dcode.fr/vernam-cipher-vigenere

Alice sends c over the wire to Bob. However, an eavesdropper, Eve, has been listening on the wire and is recording all of Alice's ciphertexts!

Assume that Alice and Bob really did implement their OTP scheme correctly and that Eve knows that Alice and Bob are using a correct OTP implementation.

(c) Assume Alice and Bob randomly chose, and now share, the secret key $k = \texttt{AAAAAAAA}$. Alice wants to send the message $m = \texttt{OTPISFUN}$. When she goes to encrypt it with $k = \texttt{AAAAAAAA}$, her ciphertext c is *also* `OTPISFUN`! What, if anything, does Eve learn about Alice's plaintext m or key k from seeing her ciphertext?

(d) On a distinct OTP setup from the previous part (where k is no longer fixed), suppose that Alice *always* begins her messages with the string `HIBOB` and that Eve is aware of Alice's habit. Alice and Bob generate a secret key k, and Eve observes the ciphertext

$$c = \texttt{HIBOBLETSMEETSATURDAY}.$$

What does Eve learn about Alice and Bob's key k from observing this ciphertext? Can Eve infer that Alice and Bob are probably meeting on Saturday?
(Again, Alice and Bob *are* using the OTP scheme, and Eve knows this. But Eve is a passive adversary and only observes c.)

Problem 3. RSA Shenanigans.

Alice has taken 6.857, and is concerned that basic RSA can not even be CPA-secure, because it is not randomized. She considers the following randomized variant of RSA, where the sender chooses the encryption exponent e at random:

1. The primes p and q are safe primes, so that primes $p = 2r+1$ and $q = 2s+1$ are chosen such that r and s are $(\lambda - 1)$-bit primes.
2. The public key $PK = (n, \lambda)$ is the number $n = pq$ and the security parameter λ. The secret key is the pair (r, s).
3. To encrypt a message m in Z_n, the *sender* randomly chooses an odd value of $e > 1$, whose bit-length is at most $\lambda - 2$, and sends $c = (e, m^e \bmod n)$ as the ciphertext.

(a) Show that Alice can successfully decrypt a message sent to her.

(b) Although Alice's RSA variant is now randomized, argue that it is nonetheless not CPA–secure.

Problem 4. MACs.

Let $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ be a CPA secure secret-key encryption scheme, and let $\texttt{MAC}$ be a message authentication code that is secure against adaptive chosen message attacks. Consider the following new encryption algorithm $(\texttt{Gen}', \texttt{Enc}', \texttt{Dec}')$:

1. The key generation algorithm $\texttt{Gen}'(1^n)$ runs $\texttt{Gen}(1^n)$ to generate a secret key sk, and then in addition $\texttt{Gen}'$ randomly generates a key $k \leftarrow \{0,1\}^n$ for the $\texttt{MAC}$.
2. The Encryption algorithm $\texttt{Enc}'$ is defined by $\texttt{Enc}'((sk,k),m) = Enc(sk, m||\texttt{MAC}(k,m))$. You may assume that $\texttt{MAC}$ produces an output of length that is fixed and known.
3. The decryption algorithm $\texttt{Dec}'$ takes as input a secret key (sk,k) and a ciphertext c, it first runs $\texttt{Dec}$ to compute $m' = \texttt{Dec}(sk,c)$. Then it parses m' as $m||t$ and outputs m if $t = \texttt{MAC}(k,m)$; otherwise it outputs $\perp$.

(Note: here, as throughout the quiz, $a||b$ means a concatenated with b.)

(a) Given $\texttt{Enc}'((sk,k),m_1)$ for an arbitrary message m_1, is it possible to generate a valid encryption to any other message $m_2 \neq m_1$? Explain your answer. (By a valid ciphertext we mean a ciphertext that does not decrypt to $\perp$.)

(b) Argue that the encryption scheme $(\texttt{Gen}', \texttt{Enc}', \texttt{Dec}')$ is not CCA secure. Give an example of two messages whose encryptions can be distinguished.

Problem 5. Symmetric cryptography in the random oracle model.

Suppose you are in a world in which there is access to a random oracle $\mathcal{H}$. With no other assumptions, which of the following can you construct? For each, either give your construction or argue why it cannot be constructed from $\mathcal{H}$. (Tip: pay careful attention to the use of any keys.)

(a) A pseudo–random function $F(k, \cdot)$.

(b) A CPA–secure symmetric encryption scheme.

(c) A secure message authentication code.

(d) A CCA–secure symmetric encryption scheme.

Problem 6. Breaking and maintaining collision resistance.

Let $H = \{h_k(x)\}$ be a collision–resistant (CR) family of hash functions where for each key $k \in \{0,1\}^n$ the hash function h_k maps $\{0,1\}^*$ to $\{0,1\}^{d(n)}$.

(a) Is $H' = \{h_k(h_k(x))\}$ necessarily CR? Explain.

(b) Consider $H' = \{f_{k'}(x)\}$, where

$$f_{k'}(x) = h_{k_1}(x)||h_{k_2}(x),$$

with $k' = k_1||k_2 \in \{0,1\}^{2n}$. Is H' necessarily CR? Explain.

(c) Consider $H' = \{g_k(x)\}$ where

$$g_k(x) = h_k(x_1) \oplus h_k(x_2),$$

with $x = x_1||x_2$ such that $|x_1| = |x_2| = |x|/2$. (You may assume $|x|$ is even.) Is H' necessarily CR? Explain.

Problem 7. Block cipher.

Let $\texttt{Enc}(k, m)$ denote a given block cipher that takes as input an n–bit key k and an n–bit message block m, and returns an n–bit ciphertext block $c = \texttt{Enc}(k, m)$. In this problem you may assume that Enc is an ideal block cipher.

Define a new block cipher $\texttt{Enc}'((k_1, k_2), m)$ in terms of Enc as follows. The block cipher $\texttt{Enc}'$ takes as input a key k consisting of *two* n-bit key-parts k_1 and k_2, and an n-bit message block m, and returns the $2n$-bit ciphertext block

$$c = (c_1, c_2) = \texttt{Enc}'((k_1, k_2), m) = \texttt{Enc}(k_1, r) \,||\, \texttt{Enc}(k_2, s)$$

where r and s are random values that add to m modulo 2^n. That is, the result is the concatenation of the encryption of a random n–bit value r under Enc using key k_1 and the encryption of $s = m - r$ under Enc using key k_2. Arithmetic is modulo 2^n, so that $r + s = m \pmod{2^n}$.

(a) Is $\texttt{Enc}'$ a CPA–secure block cipher? Explain.

(b) Is $\texttt{Enc}'$ a CCA–secure block cipher? Explain.

Problem 8. Secret Sharing over Rings.
In class, we have studied the use of Shamir's secret sharing scheme over finite fields. However, there exists an algebraic structure similar to fields called *rings* (for more info, see `https://en.wikipedia.org/wiki/Ring_(mathematics)`). For this problem, we will consider $\mathbb{Z}_{1000}$, the ring of elements $\{0, 1, \ldots, n-1\}$, where addition and multiplication are modulo $n = 1000$. Note that $\mathbb{Z}_{1000}$ is a group under addition but is not a group under multiplication.

Suppose the 6.857 staff wants to safeguard a secret $s \in \mathbb{Z}_{1000}$. The staff runs Shamir's secret sharing scheme's `share`(s) algorithm. s is the constant term of the secret-hiding polynomial $p(x)$, and we choose the remaining polynomial coefficients at random from $\mathbb{Z}_{1000}$.

(a) Suppose we give you the share $(2, p(2))$. What information does this share expose about the secret s?

(b) Argue that any share of the form $(k, p(k))$ where $k \in \mathbb{Z}_{1000}$ and $\gcd(k, 1000) > 1$ leaks information about secret s.

(c) Argue that no share of the form $(k, p(k))$ where $k \in \mathbb{Z}_{1000}$ and $\gcd(k, 1000) = 1$ leaks information about secret s.

Problem 9. CDH El-Gamal.

In class and recitation we saw the Computational Diffie-Hellman assumption (CDH), which informally says that, given g^x and g^y, it is hard for an adversary to compute g^{xy}. This is weaker than the similar Decisional Diffie-Hellman assumption (DDH), where the adversary only needs to distinguish g^{xy} from g^z, where z is randomly chosen.

(a) As a warmup, name a group where CDH holds but DDH doesn't.

(b) In class we showed that the CPA-security of the El-Gamal encryption scheme relies on DDH being hard on the cyclic group generated by the `Gen` algorithm. Show that we can not weaken the assumption to CDH. That is, argue that El-Gamal need *not* be CPA-secure if the group satisfies CDH but not DDH (that is, we *do* need the stronger assumption).

(c) Fix the problem(s) you found in the previous part, and modify the El-Gamal scheme so that CDH *is* enough for it to be CPA-secure. Argue that your construction is secure under CDH.