CS255: Cryptography and Computer Security

Practice Worksheet

This is an optional practice worksheet; no need to submit it.

Problem 1. Let (E, D) be a (one-time) semantically secure cipher with key space $K = \{0,1\}^\ell$. A bank wishes to split a decryption key $k \in \{0,1\}^\ell$ into two shares p_1 and p_2 so that both are needed for decryption. The share p_1 can be given to one executive and p_2 to another, so that both must contribute their shares for decryption to proceed.

The bank generates random k_1 in $\{0,1\}^\ell$ and sets $k_1' \leftarrow k \oplus k_1$. Note that $k_1 \oplus k_1' = k$. The bank can give k_1 to one executive and k_1' to another. Both must be present for decryption to proceed since, by itself, each share contains no information about the secret key k: each share is a one-time pad encryption of k.

Now, suppose the bank wants to split k into three shares p_1, p_2, p_3 so that any two of the shares enable decryption using k. This ensures that even if one executive is out sick, decryption can still succeed. To do so the bank generates two random pairs (k_1, k_1') and (k_2, k_2') as in the previous paragraph so that $k_1 \oplus k_1' = k_2 \oplus k_2' = k$. How should the bank assign shares so that any two shares enable decryption using k, but no single share can decrypt?

A) $p_1 = (k_1, k_2), \quad p_2 = (k_1'), \quad p_3 = (k_2')$

B) $p_1 = (k_1, k_2), \quad p_2 = (k_2, k_2'), \quad p_3 = (k_2')$

C) $p_1 = (k_1, k_2), \quad p_2 = (k_1, k_2), \quad p_3 = (k_2')$

D) $p_1 = (k_1, k_2), \quad p_2 = (k_1', k_2'), \quad p_3 = (k_2')$

E) $p_1 = (k_1, k_2), \quad p_2 = (k_2'), \quad p_3 = (k_1', k_2)$

Problem 2. Let $\mathcal{M} = \mathcal{C} = \mathcal{K} = \{0, 1, 2, \ldots, 255\}$, and consider the following cipher defined over $(\mathcal{K}, \mathcal{M}, \mathcal{C})$:

$$E(k, m) = m + k \pmod{256} \qquad ; \qquad D(k, c) = c - k \pmod{256} \ .$$

Does this cipher have perfect secrecy?

A) No, there is a simple attack on this cipher.

B) No, only the One Time Pad has perfect secrecy.

C) Yes.

D) It would, if 255 were a prime number.

Problem 3. Let (E, D) be a (one-time) semantically secure cipher where the message and ciphertext space is $\{0,1\}^n$. Which of the following encryption schemes is a (one-time) semantically secure cipher? (here $\|$ is the concatenation of two strings)

A) $E'(k,m) = E(k,m) \,\|\, \mathrm{LSB}(m)$ ($\mathrm{LSB}(m)$ is the least significant bit of m)

B) $E'(k,m) = 000 \,\|\, E(k,m)$

C) $E'(k,m) = E(k,m) \,\|\, k$

D) $E'(k,m) = E(0^n, m)$

Problem 4. Suppose that using commodity hardware it is possible to build a computer for about \$200 that can brute force about 10 billion AES keys per second. Suppose an organization wants to run an exhaustive search for a single 128-bit AES key and is willing to spend four trillion (4×10^{12}) dollars to buy these machines (this is about the annual US federal budget). How long would it take the organization to run an exhaustive search for this single 128-bit AES key with these machines, in the worst case? Ignore additional costs such as power and maintenance.

A) More than an hour but less than a day.

B) More than a day but less than a month.

C) More than a year but less than 100 years

D) More than a 100 years but less than 10,000 years

E) More than a billion (10^9) years.

Problem 5. Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a secure PRF (i.e., the key space, input space, and output space are all $\{0,1\}^n$), where $n = 128$. Consider the following derived PRFs:

$$F_1(k,x) = F(k,\ x \oplus 1^n); \quad F_2(k,x) = F(k,x) \,\|\, 0; \quad F_3\big((k_1,k_2),\ x\big) = \begin{cases} F(k_1,x) & \text{when } x \neq 0^n \\ k_2 & \text{otherwise} \end{cases}$$

Which of these is a secure PRF?

A) F_1, F_2, but not F_3.

B) F_1, F_3, but not F_2.

C) F_2, F_3, but not F_1.

D) F_1, but not F_2 or F_3.

E) F_3, but not F_1 or F_2.

Problem 6. Let m be a message consisting of ℓ AES blocks (say $\ell = 100$). Alice encrypts m using randomized counter mode and transmits the resulting ciphertext to Bob. Due to a network error, ciphertext block number $\ell/2$ is corrupted during transmission. All other ciphertext blocks are transmitted and received correctly. Once Bob decrypts the received ciphertext, how many plaintext blocks will be corrupted?

A) 1

B) 2

C) $\ell - 1$

D) ℓ

Problem 7. In nonce-based CBC mode encryption (where the nonce is unique but not random), how does one generate the IV?

A) By choosing the IV randomly.

B) By setting the IV to zero.

C) By computing AES(k, nonce) where k is the message encryption key.

D) By computing AES(k', nonce) where k' is a key used only for IV generation.

Problem 8. Suppose a MAC system (S, V) is used to protect files in a file system by appending a MAC tag to each file. The MAC signing algorithm S is applied to the file contents and nothing else. Verification works the same. What tampering attacks are not prevented by this system?

A) Changing the first byte of the file contents.

B) Replacing the contents and MAC tag of one file with the contents and MAC tag of a file from another computer that is protected by the same MAC system, but a different key.

C) Swapping the contents of two files in the file system, but keeping the original file names.

D) Erasing the last byte of the file contents.

Problem 9. Consider the encrypted CBC MAC (ECBC) built from AES. Suppose we compute the tag for a long message m comprising of n AES blocks. Let m' be the n-block message obtained from m by flipping the last bit of m (i.e. if the last bit of m is b then the last bit of m' is $b \oplus 1$). What is the minimum number of calls to AES needed to compute the MAC tag for m' given the MAC tag for m and the MAC key? (in this question please ignore message padding and simply assume that the message length is always a multiple of the AES block size)

A) 3

B) 4

C) $n - 1$

D) n

E) $n + 1$

Problem 10. Suppose H_1 and H_2 are collision resistant hash functions mapping inputs in a set $\mathcal{M}$ to $\{0,1\}^{256}$. Our goal is to show that the function $H_2(H_1(m))$ is also collision resistant. We prove the contra-positive: suppose $H_2(H_1(\cdot))$ is not collision resistant, that is, we are given $x \neq y$ such that $H_2(H_1(x)) = H_2(H_1(y))$. We build a collision for either H_1 or for H_2. This will prove that if H_1 and H_2 are collision resistant, then so is $H_2(H_1(\cdot))$. Which of the following must be true:

A) Either the pair x, y is a collision for H_1 or the pair x, y is a collision for H_2.

B) Either the pair x, y is a collision for H_1 or the pair $H_1(x), H_1(y)$ is a collision for H_2.

C) Either the pair x, y is a collision for H_2 or the pair $H_1(x), H_1(y)$ is a collision for H_1.

D) Either the pair $x, H_1(y)$ is a collision for H_2 or the pair $H_2(x), y$ is a collision for H_1.

Problem 11. If you are building an application that needs to encrypt multiple messages using a single symmetric key, what encryption method should you use?

A) Invent your own encryption mode using AES and implement it yourself.

B) Use a standard implementation of CBC encryption with a random IV.

C) Use a standard implementation of an authenticated encryption mode such as AES-GCM.

D) Implement Encrypt-then-MAC yourself based on Intel's AES-NI.

Problem 12. Let (E, D) be a symmetric encryption scheme with message space $\mathcal{M}$ (think of $\mathcal{M}$ as only consisting for short messages, say 32 bytes). Define the following MAC (S, V) for messages in $\mathcal{M}$:

$$S(k, m) := E(k, m) \quad ; \quad V(k, m, t) := \begin{cases} 1 & \text{if } D(k, t) = m \\ 0 & \text{otherwise} \end{cases}$$

What is the property that the encryption scheme (E, D) needs to satisfy for this MAC system to be secure?

A) Ciphertext integrity

B) Perfect secrecy

C) Semantic security

D) Semantic security under a chosen plaintext attack

Problem 13. What is $7^{1,000,000} \pmod{1255}$? Use Euler's theorem.
You do not need a calculator ... please do not use one. Hint: 251 is a prime number.

A) 7

B) 1

C) 7^{-1}

D) -1

E) 3

Problem 14. Consider the RSA public key (n, e) where $n = 1255$ and $e = 3$. What is the private decryption exponent d?

A) $d = 3$

B) $d = 541$

C) $d = 667$

D) $d = 87$

E) $d = 17$

Problem 15. Suppose Alice and Bob run the Diffie-Hellman protocol in the cyclic group $\mathbb{G} = \mathbb{Z}_{101}^*$ with generator $g = 7$. What is the Diffie-Hellman secret $s = g^{ab} \in \mathbb{G}$ if Alice uses $a = 3$ and Bob uses $b = 67$? (again, you do not need a calculator)

A) $s = 1$

B) $s = -1$

C) $s = 7$

D) $s = 15$

E) $s = 49$

Problem 16. Let $\mathbb{G}$ be a finite cyclic group (e.g. $\mathbb{G} = \mathbb{Z}_p^*$) with generator g. Suppose the Diffie-Hellman function $\mathrm{DH}_g(g^x, g^y) = g^{xy}$ is difficult to compute in $\mathbb{G}$. Consider the following related functions:

$$f_1(g^x, g^y) = g^{xy+3x} \; ; \qquad f_2(g^x, g^y) = g^{(x+2)(y-3)} \; ; \qquad f_3(g^x, g^y) = g^{x+2y}$$

Which of these functions must also be difficult to compute?
Hint: as usual, identify the functions for which the contra-positive holds: if $f(\cdot, \cdot)$ were easy to compute then so would $\mathrm{DH}_g(\cdot, \cdot)$.

A) f_1 and f_2, but not f_3.

B) f_2 and f_3, but not f_1.

C) f_3 and f_1, but not f_2.

D) all three are hard to compute.

E) f_2, but not f_1 or f_3.

Problem 17. Let (Gen, E, D) be a semantically secure public key encryption system. Can algorithm E be deterministic?

A) Yes, for example, the RSA trapdoor function is deterministic.

B) Some semantically secure public key encryption schemes are deterministic, while others are not.

C) No, but chosen-ciphertext secure public key encryption can be deterministic.

D) No, semantically secure public key encryption must be randomized.

Problem 18. Suppose Alice and Bob live in a country with 50 states. Alice is currently in state $a \in \{1, \ldots, 50\}$ and Bob is currently in state $b \in \{1, \ldots, 50\}$. They can communicate with one another and Alice wants to test if she is currently in the same state as Bob. If they are in the same state, Alice should learn that fact, but if not then she should learn nothing else about Bob's location. Bob should learn nothing about Alice's location.
They agree on the following scheme:

- They fix a group $\mathbb{G}$ of prime order q and generator g of $\mathbb{G}$
- Alice chooses random x and y in $\mathbb{Z}_q$ and sends to Bob $(A_0, A_1, A_2) = \left(g^x,\ g^y,\ g^{xy+a}\right)$
- Bob choose random r and s in $\mathbb{Z}_q$ and sends back to Alice $(B_1, B_2) = \left(A_1^r g^s,\ \ (A_2/g^b)^r A_0^s\right)$

What should Alice do now to test if they are in the same state (i.e. to test if $a = b$)?
Note that Bob learns nothing from this protocol because he simply received a plain ElGamal encryption of g^a under the public key g^x. One can show that if $a \neq b$ then Alice learns nothing else from this protocol because she recieves the encryption of a random value.

A) Alice tests if $a = b$ by checking if $B_2^x / B_1 = 1$.

B) Alice tests if $a = b$ by checking if $B_2^x B_1 = 1$.

C) Alice tests if $a = b$ by checking if $B_2 B_1^x = 1$.

D) Alice tests if $a = b$ by checking if $B_2 / B_1^x = 1$.

E) Alice tests if $a = b$ by checking if $B_2 / B_1 = 1$.

Problem 19. Recall that password systems make it harder to mount an offline dictionary attack by using a slow hash function. This forces the attacker to spend more effort to evaluate the hash function at many inputs. One way to construct a slow hash function is to start from a standard hash function, such as SHA256, and iterate it many times. That is,

$$H_n(x) = \text{SHA256}(\ \text{SHA256}(\cdots \text{SHA256}(x) \cdots)) \qquad \text{[iterated } n \text{ times]}$$

The number of iterations n is set so that the running time of $H_n(\cdot)$ is about 0.1 seconds for the real identification server who is verifying the password. In class we saw a slow a hash function called PBKDF2 that builds upon this basic iteration method.
In this question we show that iteration does not always slow down the time to evaluate a function. Consider the function $H : \mathbb{Z} \to \mathbb{Z}$ defined by

$$H_{p,a,b}(x) = ax + b \bmod p$$

where p is a prime and a, b are some fixed integers in $\mathbb{Z}_p$ that are chosen at random when the function is first defined. The attacker knows p, a, b. This function is not one-way and should not be used to hash passwords, but is useful for making the point of this exercise.
Let $H^{(n)}$ be the result of iterating $H_{p,a,b}$ a total of n times (say $n = 1000$). The attacker is given p, a, b and its goal is to write down the fastest program for evaluating $H^{(n)}(x)$ for $x \in \mathbb{Z}_p$. How fast can this program be?

A) Evaluating $H^{(n)}(x)$ can be done as fast as evaluating $H_{p,a,b}(x)$.

B) Evaluating $H^{(n)}(x)$ takes twice as long as evaluating $H_{p,a,b}(x)$.

C) Evaluating $H^{(n)}(x)$ takes time $O(n)$.

D) Evaluating $H^{(n)}(x)$ takes time $O(\log n)$.

Problem 20. In class we saw a one-sided AKE (authenticated key exchange protocol) with forward-secrecy and a two-sided AKE without forward-secrecy. Let's try to construct the best of both worlds: a two-sided AKE with forward-secrecy.

Consider the following two-sided AKE with forward-secrecy between Alice and Bank: They each have a certificate for a signing key and we denote by $S_{\text{alice}}(\text{data})$ and $S_{\text{bank}}(\text{data})$ their respective signatures on 'data'. They fix a group $\mathbb{G}$ of order q and generator $g \in \mathbb{G}$. Alice chooses a random a and Bank chooses a random b, both in $\mathbb{Z}_q$. They exchange the following messages:

$$\begin{array}{lcl} \textit{Alice} & \xrightarrow{\quad g^a\,,\quad \text{cert}_{\text{alice}}\,,\quad S_{\text{alice}}\,(g^a) \quad} & \textit{Bank} \\ & \xleftarrow{\quad g^b\,,\quad \text{cert}_{\text{bank}}\,,\quad S_{\text{bank}}\,(g^a,\ g^b) \quad} & \\ k \leftarrow H(g^{ab}, \text{“alice”}) & & k \leftarrow H(g^{ab}, \text{id from cert}_{\text{alice}}) \end{array}$$

Both sides compute the same key k using a hash function $H : \mathbb{G} \times \mathcal{ID} \to \mathcal{K}$, and each side deletes its secret a or b. If all the certificates and signatures verify correctly then Alice thinks she is speaking with Bank and Bank thinks it is speaking with Alice. The protocol provides forward-secrecy because a compromise of the server or the client does not compromise past sessions.

Since the Diffie-Hellman messages in this protocol are signed by the participants, one might expect that the protocol is secure against a person-in-the-middle attack. Unfortunately that is incorrect: the protocol is vulnerable to an identity misbinding attack. Which of the following actions by a person-in-the-middle leads to identity misbinding?

A) The attacker blocks Alice's message and replaces it with the following message to Bank:

$$\text{Evil} \xrightarrow{\quad g^a\,,\quad \text{cert}_{\text{evil}}\,,\quad S_{\text{evil}}\,(g^a) \quad} \text{Bank}$$

B) The attacker sets $a' \xleftarrow{R} \mathbb{Z}_q$, blocks Alice's message, and replaces it with the following message to Bank:

$$\text{Evil} \xrightarrow{\quad g^{(a')}\,,\quad \text{cert}_{\text{evil}}\,,\quad S_{\text{evil}}\big(g^{(a')}\big) \quad} \text{Bank}$$

C) The attacker blocks Bank's message and replaces it with the following message to Alice:

$$\text{Alice} \xleftarrow{\quad g^b\,,\quad \text{cert}_{\text{evil}}\,,\quad S_{\text{evil}}\,(g^a,\ g^b) \quad} \text{Evil}$$

D) The attacker chooses $b' \xleftarrow{R} \mathbb{Z}_q$, blocks Bank's message, and replaces it with the following message to Alice:

$$\text{Alice} \xleftarrow{\quad g^{(b')}\,,\quad \text{cert}_{\text{evil}}\,,\quad S_{\text{evil}}\big(g^a,\ g^{(b')}\big) \quad} \text{Evil}$$

Problem 21. Alice and Bob caught the flu. Alice has a list of people $S_a = \{u_1, \ldots, u_n\} \subseteq \mathcal{ID}$ that she recently came in contact with. Similarly, Bob has a list of people $S_b \subseteq \mathcal{ID}$ that he came in contact with. They want to identify the subset of people that they both came in contact with, namely the people in $S_a \cap S_b$, that could have been the source of the flu. The problem is that Bob does not want to reveal his contacts S_b to Alice, and similarly Alice does not want to reveal her contacts S_a to Bob. How can they compute the intersection?

This problem is called *private set intersection*: the items in the intersection of S_a and S_b should be revealed, but nothing else should be revealed about the sets. The simplest solution uses a mechanism called an *oblivious PRF*. Let F be a secure PRF defined over $(\mathcal{K}, \mathcal{ID}, \mathcal{Y})$, where $\mathcal{Y} = \{0,1\}^{256}$. Suppose Alice has some $u \in \mathcal{ID}$, and Bob has a random $k \in \mathcal{K}$. An oblivious PRF is a protocol between Alice and Bob, so that at the end of the protocol Alice obtains $y := F(k, u)$. However, Bob learns nothing about u, and Alice learns nothing else about k. Section 11.7.3 in the book gives a simple oblivious PRF from the Computational Diffie-Hellman assumption (CDH).

Suppose Alice has $S_a = \{u_1, \ldots, u_n\} \subseteq \mathcal{ID}$, and Bob has $S_b = \{v_1, \ldots, v_m\} \subseteq \mathcal{ID}$. To compute the intersection $S_a \cap S_b$, they decide to use the following protocol:

- *step 1:* Bob chooses a $k \in \mathcal{K}$.
- *step 2:* Alice and Bob run the oblivious PRF protocol n times, once for each element in S_a. Alice learns $\hat{u}_i := F(k, u_i)$ for $i = 1, \ldots, n$, but Bob learns nothing about S_a other than its size.
- *step 3:* Bob computes $\hat{v}_j := F(k, v_j)$ for all $j = 1, \ldots, m$ and sends $\hat{v}_1, \ldots, \hat{v}_m \in \mathcal{Y}$ to Alice.
- *step 4:* Alice finds all $u \in S_a$ such that $\hat{u} := F(k, u)$ is in $\{\hat{v}_1, \ldots, \hat{v}_m\}$. She outputs the set of all such u as the intersection $S_a \cap S_b$.

Bob clearly learns nothing about S_a in this protocol, other than its size. What should Bob do to ensure that Alice learns the intersection, but learns nothing else about his set S_b other than its size?

A) Choose $k, k' \xleftarrow{R} \mathcal{K}$, then use k in step 2 and k' in step 3.

B) Always set k to $0 \in \mathcal{K}$ in step 1.

C) Choose $k \xleftarrow{R} \mathcal{K}$ in step 1, and delete k once the protocol is finished.

D) Send k to Alice at the end of the protocol.