CS 6222: Introduction to Cryptography **Spring 2020**

Homework 3: Authenticated Encryption and Public-Key Cryptography

Due: March 18, 2020 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/sp20/static/homework.tex

You must submit your problem set via Gradescope. Please use course code **MB84NW** 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. Understanding Definitions [20 points]. Throughout this problem, you should assume that secure PRFs and secure collision-resistant hash functions exist. You may cite examples and constructions from class (or lecture notes) without proof.

(a) Give an example of a CPA-secure encryption scheme where the ciphertexts are *not* pseudorandom (i.e., the ciphertexts are not indistinguishable from uniformly random strings). *Remark: This shows that CPA-security only says that the ciphertext hides the message; it does not mean that ciphertexts look like random strings, and in many schemes, the ciphertext will* not *look like a random string.*

(b) Give an example of an encryption scheme that is CCA-secure but not an authenticated encryption scheme. *Remark: This shows that the converse of the statement "authenticated encryption implies CCA-security" is false.*

(c) Give an example of a collision-resistant hash function that is no longer collision resistant if you drop the last bit of the output. *Remark: This shows that even dropping a single bit of the output of a CRHF can break collision resistance.*

Problem 2. Hash-then-Encrypt [20 points]. The Android KeyStore uses "hash-then-CBC-encrypt" to construct an authenticated encryption scheme to generate and manage cryptographic keys for Android applications. Abstractly, the scheme operates as follows: Let $(\mathsf{Enc}_{\mathsf{CBC}}, \mathsf{Dec}_{\mathsf{CBC}})$ be a randomized CBC-mode encryption scheme built from a block cipher $F\colon \mathcal{K} \times \mathcal{X} \to \mathcal{X}$. Let $H\colon \mathcal{X}^{\leq L} \to \mathcal{X}$ be a collision-resistant hash function. Define the following candidate authenticated encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{Enc}(k, m)$: Output $c \leftarrow \mathsf{Enc}_{\mathsf{CBC}}(k, H(m) \| m)$.
- $\mathsf{Dec}(k, c)$: Compute $(t, m) \leftarrow \mathsf{Dec}_{\mathsf{CBC}}(k, c)$ and output m if $t = H(m)$ and $\perp$ otherwise.

In the following, assume that $\mathcal{X} = \{0,1\}^n$ and $L \geq 2$.

(a) Show that $(\mathsf{Enc}, \mathsf{Dec})$ does not provide ciphertext integrity.

(b) Show that $(\mathsf{Enc}, \mathsf{Dec})$ is not CCA-secure. Recall that for encryption schemes over a variable-length message space, the adversary can only query the encryption oracle on pairs (m_0, m_1) where m_0 and m_1 have the *same* length.

(c) Would the above problems go away if the Android KeyStore had used randomized counter mode encryption instead of CBC-mode encryption? Give a brief explanation.

Both attacks show that the Android KeyStore does not provide authenticated encryption. These attacks were discovered in January 2016 and Google has confirmed that the encryption scheme will be removed from the system.

Problem 3. Commitment Schemes from Discrete Log [16 points]. A commitment scheme is a digital analog of a "sealed envelope." Specifically, a sender can *commit* to a message m and send the resulting commitment c to a receiver (i.e., seal the message in an envelope). The commitment c should not reveal anything about the committed value m. Later on, the sender can *open* up the commitment and convince the receiver that c is indeed a commitment to the message m (i.e., open up the envelope and recover the original message). The commitment scheme is *hiding* if c hides the message m and is *binding* if the sender cannot open the commitment c to any message $m' \neq m$. In this problem, we will construct a commitment scheme from the discrete log assumption:

- **Public parameters:** Let $\mathbb{G}$ be a group of prime order p and let $g, h \in \mathbb{G}$ be arbitrary elements of $\mathbb{G}$ (that are not the identity element).
- **Commitment:** To commit to a message $m \in \mathbb{Z}_p$, sample $r \xleftarrow{R} \mathbb{Z}_p$ and output the commitment $c \leftarrow g^m h^r$.
- **Open:** To open the commitment c to the message m, the sender gives (m, r) to the receiver and the receiver checks that $c = g^m h^r$.

(a) Show that the above commitment scheme is *perfectly hiding* (i.e., the commitment c does not leak *any* information about the committed message m). Namely, show that given the commitment $c \in \mathbb{G}$, every candidate message $m' \in \mathbb{Z}_p$ is *equally likely* (over the randomness of r). One way to show this is that for every $m' \in \mathbb{Z}_p$, there is a *unique* $r' \in \mathbb{Z}_p$ such that $c = g^{m'} h^{r'}$.

(b) Show that the above commitment scheme is *computationally binding* assuming hardness of discrete log in $\mathbb{G}$. Namely, show that if an efficient adversary can output a commitment c together with openings (m, r) and (m', r') such that $g^m h^r = c = g^{m'} h^{r'}$ and $m \neq m'$, then the adversary can also compute the discrete log of h base g. In other words, if the sender can open the commitment in two different ways, then it can also compute the discrete log of h in $\mathbb{G}$.

Remember to give a brief explanation why any inverses you take actually exist.

Problem 4. Encrypted Group Chat [20 points]. Suppose a group of n people (denoted $P_1, \ldots, P_n$) want to set up a shared key for an encrypted group chat. At the end of the key-exchange, everyone within the group should know the key, but an eavesdropper on the network should not. We will use the following variant of Diffie-Hellman over a group $\mathbb{G}$ of prime order p and generator g:

- At the beginning of the protocol, P_1 chooses $s \xleftarrow{R} \mathbb{Z}_p$. We will view P_1 as the group administrator that all of the other parties know.
- Each of the other parties P_i ($2 \le i \le n$) samples $r_i \xleftarrow{R} \mathbb{Z}_p$ and sends $x_i \leftarrow g^{r_i}$ to the group administrator P_1. The administrator P_1 replies to P_i with x_i^s.
- The group key is then defined to be $k \leftarrow H(g^s)$, where $H\colon \mathbb{G} \to \{0,1\}^\lambda$ is a hash function.

Both the group description $(\mathbb{G}, p, g)$ and the hash function H are public and known to everyone (both the protocol participants and the eavesdropper).

(a) Show that both the group administrator P_1 and each of the parties P_i ($2 \le i \le n$) are able to efficiently compute the group key.

(b) We say that the group key-exchange protocol is secure against eavesdroppers if no efficient adversary who sees the transcript of messages sent by the honest parties $P_1, \ldots, P_n$ is able to distinguish the group key k from a uniform random string over $\{0,1\}^\lambda$, except perhaps with negligible probability. If we model H as an "ideal hash function" (i.e., random oracle), it suffices to argue that the shared Diffie-Hellman secret g^s is *unguessable* to any efficient eavesdropper $\mathcal{A}$:

$$\Pr[\mathcal{A}(x_2, x_2^s, \ldots, x_n, x_n^s) = g^s] = \mathrm{negl}(\lambda),$$

where $x_i = g^{r_i}$ and $r_2, \ldots, r_n, s \xleftarrow{R} \mathbb{Z}_p$. This means that an eavesdropper who only observes the messages sent by the honest parties cannot guess g^s, and correspondingly, the shared key $H(g^s)$ is uniformly random and unknown to the adversary. Show that under the CDH assumption in $\mathbb{G}$, the shared Diffie-Hellman secret g^s in the group key-exchange protocol above is unguessable. Namely, show that if there exists an efficient adversary $\mathcal{A}$ that can predict the Diffie-Hellman secret in the above key-exchange protocol, then there exists an efficient algorithm $\mathcal{B}$ that breaks CDH in $\mathbb{G}$. **Hint:** Your algorithm $\mathcal{B}$ may need to invoke $\mathcal{A}$ *more than once*. Remember to compute the advantage of the adversary you construct.

Problem 5. Collision-Resistant Hashing from RSA [14 points]. In this problem, we will show how to construct a collision-resistant hash function from the RSA assumption. Let $N = pq$ be an RSA modulus and take $e \in \mathbb{N}$ to be a prime that is also relatively prime to $\varphi(N)$. Let $u \xleftarrow{R} \mathbb{Z}_N^*$, and define the hash function

$$H_{N,e,u}\colon \mathbb{Z}_N \times \{0, \ldots, e-1\} \to \mathbb{Z}_N \quad \text{where} \quad H_{N,e,u}(x,y) = x^e u^y \in \mathbb{Z}_N.$$

In this problem, we will show that under the RSA assumption, $H_{N,e,u}$ defined above is collision-resistant. Namely, suppose there is an efficient adversary $\mathcal{A}$ that takes as input (N, e, u) and outputs $(x_1, y_1) \neq (x_2, y_2)$ such that $H_{N,e,u}(x_1, y_1) = H_{N,e,u}(x_2, y_2)$. We will use $\mathcal{A}$ to construct an efficient adversary $\mathcal{B}$ that takes as input (N, e, u) where $u \xleftarrow{R} \mathbb{Z}_N^*$ and outputs x such that $x^e = u \in \mathbb{Z}_N$.

(a) Show that using algorithm $\mathcal{A}$ defined above, algorithm $\mathcal{B}$ can efficiently compute $a \in \mathbb{Z}_N$ and $b \in \mathbb{Z}$ such that $a^e = u^b \pmod N$ and $0 \neq |b| < e$. Remember to argue why any inverses you compute will exist (or alternatively, if they do not exist, then $\mathcal{B}$ can directly break RSA).

(b) Use the above relation to show how $\mathcal{B}$ can *efficiently* compute $x \in \mathbb{Z}_N$ such that $x^e = u$. **Hint:** Since $|b| < e$ and e is prime, $\gcd(b, e) = 1$. Now, apply Bezout's identity. Note that $\mathcal{B}$ does not know the factorization of N, so it is not able to compute $b^{-1} \pmod{\varphi(N)}$.

(c) Show that if we extend the domain of $H_{N,e,u}$ to $\mathbb{Z}_N \times \{0, \ldots, e\}$, then the function is no longer collision-resistant.

Problem 6: 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. To receive the extra credit for this problem, you must submit your homework to Gradescope (with the provided template) and properly assign *all* problems to their respective pages.

Optional Feedback [0 points]. 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 so far?