CS 388H: Cryptography **Spring 2025**

Homework 3: Symmetric and Public-Key Cryptography

Due: March 5, 2025 at 11:59pm (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/sp25/static/homework.tex

You must submit your problem set via Gradescope (accessible through Canvas).

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 names of all of your collaborators with your submission. Refer to the official course policies for the full details.

Problem 1: Hash-then-Encrypt [24 points]. An old version of 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 $(\mathsf{Encrypt}_{\mathrm{CBC}}, \mathsf{Decrypt}_{\mathrm{CBC}})$ 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 $(\mathsf{Encrypt}, \mathsf{Decrypt})$:

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

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

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

(b) Show that $(\mathsf{Encrypt}, \mathsf{Decrypt})$ 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.

Problem 2: The Importance of Independent Keys [25 points]. In lecture, we saw several constructions that relied on the use of two *independent* keys. In this problem, we will show that independent keys are essential for security in many of these settings. Let $F\colon \{0,1\}^\lambda \times \{0,1\}^\ell \to \{0,1\}^\ell$ be a secure PRP, and let $F_{\mathrm{CBC}}\colon \{0,1\}^\lambda \times (\{0,1\}^\ell)^{\leq L} \to \{0,1\}^\ell$ be the *raw-CBC MAC* from lecture (where $L \geq 3$) built from F. In lecture, we said that raw-CBC is a secure MAC for fixed-length messages (and more generally, *prefix-free messages*).

(a) Recall the encrypted CBC-MAC construction. To sign a message m, we first compute the rawCBC MAC $\tau' = F_{\mathrm{CBC}}(k_1, m)$. The final MAC is then $\tau = F(k_2, \tau')$, where k_2 is an independent key. Suppose we implemented this construction but set $k_1 = k_2$. Prove that this scheme is insecure. It suffices for the adversary to query for a tag on a single message before outputting its forgery (but you are welcome to use more queries if it is easier).

(b) Suppose we use "encrypt-then-MAC" to construct an authenticated encryption scheme for a *fixed-length* message space $\{0,1\}^\ell$ (i.e., one-block messages) by combining randomized counter-mode encryption with raw-CBC MAC, except we use the *same* key for both the encryption scheme and the MAC. Namely, an encryption of $m \in \{0,1\}^\ell$ consists of the tuple (IV, c, t) where $\mathsf{IV} \xleftarrow{R} \{0,1\}^\ell$, $c \leftarrow F(k, \mathsf{IV}) \oplus m$, and $t \leftarrow F_{\mathrm{CBC}}(k, (\mathsf{IV}, c))$. Show that the resulting scheme is neither CPA-secure *nor* provides ciphertext integrity (i.e., construct two separate adversaries).

Remark: Observe that if we had used independent keys for the encryption scheme and the MAC scheme, then this scheme does provide authenticated encryption since counter-mode is CPA-secure and raw-CBC MAC is a secure MAC for a fixed-length message space. The combination of counter-mode encryption together with encrypted CBC-MAC is also known as the CCM mode of operation. This is another mode that provides authenticated encryption (when used with *independent* keys).

Problem 3: DDH in $\mathbb{Z}_p^*$ [25 points]. Let $p > 2$ be a prime. In this problem, we will show that the DDH assumption does *not* hold in $\mathbb{Z}_p^*$ (recall from class that assumptions like CDH or discrete log are believed to hold over $\mathbb{Z}_p^*$). Throughout this problem, you can use (without proof) the fact that a non-zero polynomial of degree d has at most d roots over $\mathbb{Z}_p$.

(a) A square root of z is an element $y \in \mathbb{Z}_p^*$ where $y^2 = z \bmod p$. If z has a square root in $\mathbb{Z}_p^*$, we say it is a "quadratic residue" modulo p. Show that there are exactly $(p-1)/2$ quadratic residues in $\mathbb{Z}_p^*$.

(b) Using the result from Part (a), show that z is a quadratic residue modulo p **if and only if** $z^{(p-1)/2} = 1 \bmod p$.

(c) Let g be a generator of $\mathbb{Z}_p^*$. Take any exponent $\alpha \in \mathbb{Z}$. Using the result from Part (b), show that $g^\alpha \in \mathbb{Z}_p^*$ is a quadratic residue **if and only if** α is an even integer.

(d) Using the results from Parts (b) and (c), show that the DDH assumption is false over $\mathbb{Z}_p^*$. You should describe an explicit algorithm, show it is efficient (i.e., polynomial-time in the input length), and compute its advantage.

Problem 4: Time Spent [1 point]. How long did you spend on this problem set? This is for calibration purposes, and the response you provide does not affect your score.

Optional Feedback. 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?