6.875: Cryptography and Cryptanalysis **October 27, 2021**

Last updated: November 4, 2021

Problem Set 4

Instructor: Vinod Vaikuntanathan

TAs: Lali Devadas, Aparna Gupte, Sacha Servan-Schreiber

Instructions.

- **When:** This problem set is due on **November 10, 2021** before **11pm ET**.
- **How:** You should use LaTeX to type up your solutions (you can use our LaTeX template from the course webpage). Solutions should be uploaded on Gradescope as a single pdf file.
- **Acknowledge your collaborators:** Collaboration is permitted and encouraged in small groups of at most three. You must write up your solutions *entirely on your own* and *acknowledge your collaborators.*
- **Reference your sources:** If you use material from outside the lectures, you must reference your sources (papers, websites, wikipedia, ...).
- **When in doubt, ask questions:** Use Piazza or the TA office hours for questions about the problem set. See the course webpage for the timings.

Problem 1. Commitment issues!

A commitment scheme $(\langle \mathcal{S}, \mathcal{R} \rangle, \mathsf{Verify})$ for a message space $\mathcal{M}$ and security parameter λ consists of an interactive protocol between a PPT sender $\mathcal{S}$ and a PPT receiver $\mathcal{R}$ as well as an efficient algorithm Verify, satisfying correctness, hiding, and binding defined below. We denote running the interactive protocol between the sender $\mathcal{S}$ with input $m \in \mathcal{M}$ and the receiver $\mathcal{R}$ with no input by

$$[(c,d)_{\mathcal{S}},\ (c)_{\mathcal{R}}] \leftarrow \langle S(1^\lambda, m), R(1^\lambda) \rangle\ ,$$

where (c, d) is the output of the sender and (c) is the output of the receiver. Verify takes as input m, c, d and returns yes if d is a valid opening of the commitment c for the message m and no otherwise.

Definition 1 (Correctness)**.** *A commitment scheme $(\langle \mathcal{S}, \mathcal{R} \rangle, \mathsf{Verify})$ with message space $\mathcal{M}$ and security parameter λ satisfies correctness if for all $m \in \mathcal{M}$,*

$$\Pr\Big[\ [(c,d)_{\mathcal{S}},\ (c)_{\mathcal{R}}] \leftarrow \langle \mathcal{S}(1^\lambda, m), \mathcal{R}(1^\lambda) \rangle \ \ :\ \ \mathsf{Verify}(m, c, d) = \mathsf{yes}\ \Big] = 1.$$

Definition 2 (Hiding)**.** *A commitment scheme $(\langle \mathcal{S}, \mathcal{R} \rangle, \mathsf{Verify})$ with message space $\mathcal{M}$ and security parameter λ is said to be perfectly hiding if for all (possibly malicious; possibly unbounded) $\mathcal{R}^*$ and all messages $m_0, m_1 \in \mathcal{M}$:*

$$\mathsf{view}_{\mathcal{R}^*}(\langle \mathcal{S}(1^\lambda, m_0), \mathcal{R}^*(1^\lambda) \rangle) \equiv \mathsf{view}_{\mathcal{R}^*}(\langle \mathcal{S}(1^\lambda, m_1), \mathcal{R}^*(1^\lambda) \rangle)$$

where $\mathsf{view}_{\mathcal{R}^*}$ is everything $\mathcal{R}^*$ sees while interacting with $\mathcal{S}$, i.e. all messages sent between $\mathcal{S}$ and $\mathcal{R}^*$ and $\mathcal{R}^*$'s internal randomness.

If for all (possibly malicious) PPT recipients $\mathcal{R}^*$, the two distributions are computationally indistinguishable, then we say the commitment scheme is computationally hiding and denote it as:

$$\mathsf{view}_{\mathcal{R}^*}(\langle \mathcal{S}(1^\lambda, m_0), \mathcal{R}^*(1^\lambda) \rangle) \approx_c \mathsf{view}_{\mathcal{R}^*}(\langle \mathcal{S}(1^\lambda, m_1), \mathcal{R}^*(1^\lambda) \rangle).$$

Definition 3 (Binding)**.** *A commitment scheme* $(\langle \mathcal{S}, \mathcal{R} \rangle, \mathsf{Verify})$ *with message space* $\mathcal{M}$ *and security parameter* λ *is said to be* *statistically binding* *if for all (possibly malicious; possibly unbounded)* $\mathcal{S}^*$ *and all messages* $m \neq m' \in \mathcal{M}$*:*

$$\Pr\left[[(c,d,d')_{\mathcal{S}^*}, (c)_{\mathcal{R}}] \leftarrow \langle \mathcal{S}^*(1^\lambda), \mathcal{R}(1^\lambda) \rangle \ : \ \begin{array}{l} \mathsf{Verify}(m,c,d) = \mathsf{yes}; \\ \mathsf{Verify}(m',c,d') = \mathsf{yes} \end{array} \right] \leq \mathsf{negl}(\lambda).$$

If the statement holds for all (possibly malicious) PPT senders $\mathcal{S}^*$*, then we say the commitment scheme is* *computationally binding**.*

(a) **Prove that a commitment scheme cannot be simultaneously perfectly hiding and statistically binding.**

(b) **Construct a** ***computationally hiding and statistically binding*** **commitment scheme with message space** $\mathcal{M}$ **based on the Decisional Diffie-Hellman (DDH) assumption in** $\mathbb{Z}_p^*$ **(where** $p = 2q+1$ **such that** q **is also prime). (You can assume e.g.,** $\mathcal{M} = \mathbb{Z}_{p-1}$ **or** $\mathcal{M} = \mathbb{Z}_p^*$**.) Prove your construction is correct,** ***computationally hiding, and statistically binding*** **under the DDH assumption.**

(c) **Construct a** ***perfectly hiding and computationally binding*** **commitment scheme based on the hardness of the discrete logarithm problem in** $\mathbb{Z}_p^*$ **(where** $p = 2q+1$ **such that** q **is also prime). (You can assume e.g.,** $\mathcal{M} = \mathbb{Z}_{p-1}$ **or** $\mathcal{M} = \mathbb{Z}_p^*$**.) Prove your construction is correct,** ***perfectly hiding, and computationally binding*** **under the discrete logarithm assumption.**

Problem 2. Back to MACs

Alice and Bob want to design a simple secret-key message authentication code (MAC) using hash functions. They learned in 6.875 that pseudorandom functions can be used to construct MACs, but they want to try something different. They define $\Pi = (\mathsf{Gen}, \mathsf{MAC}, \mathsf{Verify})$ as follows, using a hash function $h : \{0,1\}^\lambda \to \{0,1\}^{\ell(\lambda)}$:

$\mathsf{Gen}(1^\lambda)$	$\mathsf{MAC}(sk, m \in \{0,1\}^\lambda)$	$\mathsf{Verify}(sk, m, \sigma)$
1 : $sk \xleftarrow{R} \{0,1\}^\lambda$	1 : $\sigma \leftarrow h(sk \oplus m)$	1 : $t \leftarrow h(sk \oplus m)$
2 : **return** sk	2 : **return** σ	2 : **if** $\sigma = t$: **return** 1
		3 : **else** : **return** 0

(a) **For this part, assume that** h **is a random oracle. That is, it is a** ***public random function*** **that all the algorithms (that is,** Gen, MAC **and** Verify**) as well as the adversary have oracle access to. Give a proof in the random oracle model that** Π **is an EUF-CMA secure MAC for** λ**-bit messages.**

(b) **Alice and Bob like the simplicity of the scheme, but they have philosophical disagreements on what security in the random oracle model**

actually means when h is replaced with SHA-3 (a popular but messy hash function you haven't seen in class) in the real world. They start thinking about using collision-resistant hash functions in place of the random oracle, with the goal of coming up with a proof of security that does not resort to the strangeness of random oracles. They consider the following scheme. Let $\mathcal{H}_\lambda = \left\{h : \{0,1\}^\lambda \to \{0,1\}^{\ell(\lambda)}\right\}$ be a collision-resistant hash function family.

$$
\begin{array}{l}
\underline{\mathsf{Gen}(1^\lambda)} \\
h \xleftarrow{R} \mathcal{H}_\lambda \\
\text{publish } h \text{ on bulletin board} \\
sk \xleftarrow{R} \{0,1\}^n \\
\textbf{return } sk
\end{array}
\qquad
\begin{array}{l}
\underline{\mathsf{MAC}(sk, m \in \{0,1\}^\lambda)} \\
\sigma \leftarrow h(sk \oplus m) \\
\textbf{return } \sigma
\end{array}
\qquad
\begin{array}{l}
\underline{\mathsf{Verify}(sk, m, \sigma)} \\
t \leftarrow h(sk \oplus m) \\
\textbf{if } \sigma = t : \textbf{return } 1 \\
\textbf{else } : \textbf{return } 0
\end{array}
$$

Either prove that Π is an EUF-CMA secure MAC whenever $\mathcal{H}$ is a CRHF family, or provide a counterexample.

Problem 3. Upgrading Lamport signatures

Recall Lamport's signature scheme from class, based on a OWF $f : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$, that produces an $(\ell_1 \cdot n)$-bit signature for an n-bit message:

$$
\begin{array}{l}
\underline{\mathsf{Gen}(1^\lambda)} \\
x_{1,0}, \ldots, x_{n,0} \xleftarrow{R} \{0,1\}^{\ell_1} \\
x_{1,1}, \ldots, x_{n,1} \xleftarrow{R} \{0,1\}^{\ell_1} \\
sk := (x_{1,0}, \ldots, x_{n,0}, x_{1,1}, \ldots, x_{n,1}) \\
vk := (y_{1,0}, \ldots, y_{n,0}, y_{1,1}, \ldots, y_{n,1}), \ \text{where } y_{i,c} = f(x_{i,c}) \\
\textbf{return } (sk, vk)
\end{array}
$$

$$
\begin{array}{l}
\underline{\mathsf{Sign}(sk, m \in \{0,1\}^n)} \\
\textbf{parse } sk = (x_{1,0}, \ldots, x_{n,0}, x_{1,1}, \ldots, x_{n,1}) \\
\textbf{return } \sigma := (x_{1,m_1}, \ldots, x_{n,m_n})
\end{array}
\qquad
\begin{array}{l}
\underline{\mathsf{Verify}(vk, m \in \{0,1\}^n, \sigma)} \\
\textbf{parse } \sigma := (\sigma_1, \ldots, \sigma_n) \\
\textbf{if } \forall i \in [n], f(\sigma_i) \stackrel{?}{=} y_{i,m_i} : \textbf{return } 1 \\
\textbf{else } : \textbf{return } 0
\end{array}
$$

In this problem, we will look at a stronger definition of one-time unforgeability known as *one-time strong unforgeability* which states that not only is the adversary unable to produce a signature on a different message, but also that she is unable to produce a *different* signature σ^* on the same message it requested a signature on.

Definition 4 (One-time strong unforgeability)**.**
Let (Gen, Sign, Verify) *be a digital signature scheme with message space $\mathcal{M}$ and key space $\mathcal{K}$ with security parameter λ. This scheme is one-time strongly unforgeable if for all pair of PPT algorithms $(\mathcal{A}_1, \mathcal{A}_2)$, there exists a negligible function* negl *such that for all λ*

$$
\Pr\left[\begin{array}{l}
(sk, vk) \leftarrow \mathsf{Gen}(1^\lambda); \\
(m, \mathsf{state}) \leftarrow \mathcal{A}_1(vk); \\
\sigma \leftarrow \mathsf{Sign}(sk, m); \\
\sigma^* \leftarrow \mathcal{A}_2(\sigma, \mathsf{state})
\end{array} : \begin{array}{l}
\sigma^* \neq \sigma; \\
\mathsf{Verify}(vk, m, \sigma^*) = 1
\end{array}\right] \leq \mathsf{negl}(\lambda).
$$

(a) **Show an attack on the one-time strong unforgeability of Lamport's scheme. That is, construct a OWF f such that the Lamport signature scheme using f is not one-time strongly unforgeable.**

(b) **What additional property of the one-way function will make Lamport's scheme one-time strongly unforgeable? State the property and prove one-time strong unforgeability. (Keep the additional requirement on the OWF as minimal as you can.)**

Problem 4. ZK Proof of 1-out-of-2 QR Recall the quadratic residue problem described in class: Given a composite number $N = pq$ where p and q are two λ-bit primes, determine if a value $a \in \mathbb{Z}_N^*$ is of the form $a = b^2 \bmod N$ for some $b \in \mathbb{Z}_N^*$.

The quadratic residuosity assumption states that determining if $a \in \mathsf{QR}_N$ is computationally hard. A simple (but not zero-knowledge) proof that a is a quadratic residue is simply the value b. A verifier can efficiently check that $a = b^2 \bmod N$.

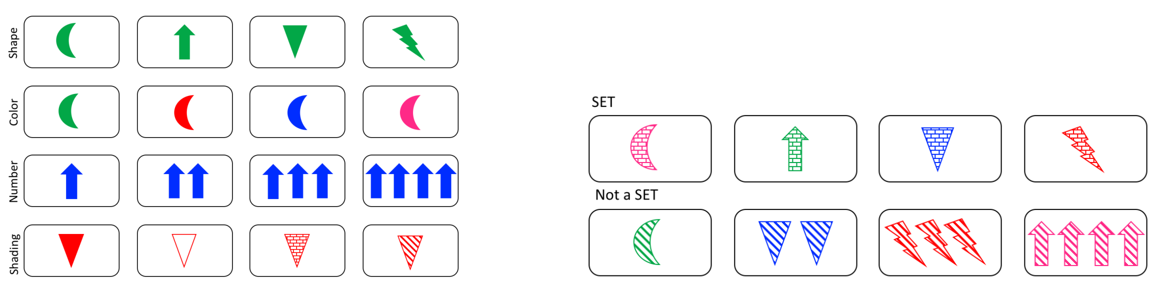
We will now explore a more interesting variant of this idea: proving, without leaking information about y_0 or y_1, that one of two values y_0, y_1 is a quadratic residue mod N.

(a) **As a warmup, provide a honest-verifier 2-message zero-knowledge protocol for proving that exactly one of y_0 and y_1 is a quadratic residue (and the other is not).**

(b) **Construct a malicious-verifier zero-knowledge 3-message protocol for proving that at least one of y_0 and y_1 is a quadratic residue mod N. Remember, you need to prove: *completeness*, *soundness*, and *zero-knowledge*.**

Problem 5. Zero Knowledge Proof System for Set

Set[1] is a card game. The object of the game is to identify a SET of n cards from n^2 cards. Each card has n features, and each feature has n possible values. A SET consists of n cards with the property that $\lfloor \frac{n}{2} \rfloor$ out of the n features are the same on each card, and $\lceil \frac{n}{2} \rceil$ of the features are different on each card. See an example with $n = 4$ below.



(a) Set Features

(b) Example of a SET and not a SET

[1]We modify the rules of the original game called Set, so please read the game instructions.

Design an honest-verifier zero-knowledge proof system for Set, i.e., for the language of Set instances (that is, collections of n^2 labeled cards) that contain a SET. Your protocol should have perfect completeness and soundness error $1 - \delta(n)$ for a non-negligible function δ.