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 <u>correctness</u>, <u>hiding</u>, and <u>binding</u> 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 <u>correctness</u> 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 <u>perfectly hiding</u> 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 <u>computationally hiding</u> 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.**

Solution

Suppose we have a commitment scheme that is correct and perfectly hiding. We will show how to break statistical binding as a malicious, unbounded sender $\mathcal{S}^*$. Pick some $m, m' \in \mathcal{M}$. Let

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

Because the scheme is perfectly hiding, we have that the distributions of the receiver's view while running the commitment protocol on either message is identical, i.e.

$$\mathsf{view}_{\mathcal{R}}(\langle \mathcal{S}(1^\lambda, m), \mathcal{R}(1^\lambda)\rangle) \equiv \mathsf{view}_{\mathcal{R}}(\langle \mathcal{S}(1^\lambda, m'), \mathcal{R}(1^\lambda)\rangle).$$

We can conclude that

$$\Pr[[(c', d')_{\mathcal{S}^*}, (c')_{\mathcal{R}}] \leftarrow \langle \mathcal{S}(1^\lambda, m'), \mathcal{R}(1^\lambda)\rangle : c' = c] > 0.$$

By correctness, we then have that

$$\mathsf{Verify}(m, c, d) = \mathsf{yes} \quad \text{and} \quad \mathsf{Verify}(m', c, d') = \mathsf{yes}$$

so the scheme is not statistically binding. As such, we conclude that there cannot exist a commitment scheme that is both 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 (where $p = 2q + 1$ such that q is also prime). (You can assume e.g., $\mathcal{M} = \mathcal{M} = \mathbb{Z}_p^*$.) Prove your construction is correct, *computationally hiding, and statistically binding* under the DDH assumption.**

Solution

We define the following commitment scheme.

$\mathcal{S}(1^\lambda, m \in \mathbb{Z}_q)$		$\mathcal{R}(1^\lambda)$

Commitment

$p \leftarrow$ first λ-bit prime		$p \leftarrow$ first λ-bit prime
$g \leftarrow$ canonical generator of QR_p		$g \leftarrow$ same generator of QR_p
$d \xleftarrow{R} \mathbb{Z}_q$		
$h \leftarrow$ random generator of QR_p		
$c := (h, g^d, g^m h^d) \in (\mathsf{QR}_p)^3$		
	$\xrightarrow{\quad c \quad}$	

Opening

	$\xrightarrow{\quad m, d \quad}$	

$\mathsf{Verify}(m, c, d)$

parse $c = (h, a, b)$

$\alpha := g^d \bmod p$

$\beta := g^m h^d \bmod p$

return yes if $(a, b) \stackrel{?}{=} (\alpha, \beta)$
and no otherwise.

Correctness. If the sender commits to $m \in \mathbb{Z}_q$ then $c = (h, \underbrace{g^d}_{a}, \underbrace{g^m h^d}_{b})$ for some random d and where g, h are random generators. We then have that if $a = g^d = \alpha$ and $b = g^m h^d = \beta$ and so the receiver accepts a valid opening.

Computational hiding. We first prove that the above scheme is computationally hiding. Assume towards contradiction that it is *not* computationally hiding. Then, there exists a PPT $\mathcal{A}$ (i.e., the malicious receiver) such that for some non-negligible δ and some messages $m_0 \neq m_1 \in \mathcal{M}$

$$\Pr\left[\begin{array}{l} b \xleftarrow{R} \{0,1\}; \\ \mathsf{view}_{\mathcal{A}}^{m_b} \leftarrow \langle \mathcal{S}(1^\lambda, m_b), \mathcal{A}(1^\lambda) \rangle; \\ b' \leftarrow \mathcal{A}(1^\lambda, \mathsf{view}_{\mathcal{A}}^{m_b}) \end{array} : \; b' = b \right] \geq \frac{1}{2} + \delta(\lambda).$$

Note that for our scheme, $\mathsf{view}_{\mathcal{R}}(\langle \mathcal{S}(1^\lambda, m), \mathcal{R}(1^\lambda) \rangle) = c$. Construct PPT $\mathcal{B}$ which breaks the DDH assumption as follows. We will let $\mathcal{B}$ act as the sender and $\mathcal{A}$ act as the recipient.

$\mathcal{B}(\mathsf{QR}_p, g, g^x, g^y, g^z)$

$b \xleftarrow{R} \{0,1\}$

$c \leftarrow (g^x, g^y, g^{m_b} g^z)$

$b' \leftarrow \mathcal{A}(1^\lambda, c)$

return 1 (DDH) if $b = b'$ and 0 (not DDH) otherwise.

Solution (continued...)

Analysis: Suppose that $\mathcal{B}$ receives as input $(\mathsf{QR}_p, g, g^x, g^y, g^{xy})$—i.e., a DDH tuple. Then the commitment $c = (g^x, g^y, g^{m_b+xy})$ follows the exact distribution of the above scheme (and hence the input expected by $\mathcal{A}$) because $c = (h, g^y, g^{m_b}h^y)$ for $h = g^x$. Hence, we have that $\mathcal{A}$ succeeds with non-negligible advantage $\delta(\lambda)$ which transfers to the advantage of $\mathcal{B}$ in breaking DDH. On the other hand, if $\mathcal{B}$ receives as input $(\mathsf{QR}_p, g, g^x, g^y, g^z)$—i.e., a uniformly random tuple—then $\mathcal{A}$ receives an invalid commitment $c = (g^x, g^y, g^{m_b+z})$. This is *crucially* distributed independently of m_b given that z is random and independent of x and y. Therefore, $\mathcal{A}$'s advantage is 0 and $\mathcal{B}$ outputs correctly with probability $\frac{1}{2}$. The overall advantage of $\mathcal{B}$ is therefore $\frac{1}{2}\delta(\lambda)$ which is non-negligible.

Statistical binding. Suppose that the scheme is **not** statistically binding. Then, we have that there exists $c = (h, a, b), m \neq m', d, d'$ such that

$$\begin{aligned} a &= g^m h^d = g^{m+xd} = g^{m'} h^{d'} = g^{m'+xd'} \bmod p, \\ b &= g^d = g^{d'} \bmod p. \end{aligned}$$

By assumption that $m \neq m'$ we get that $d \neq d'$. However, this is a contradiction since $g^d = g^{d'}$ and g is a generator, thus $d = d'$. Therefore, we conclude that there does not exist d' that opens c to message m', making the scheme statistically binding.

(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., $\mathcal{M} = \mathcal{M} = \mathbb{Z}_p^*$.) Prove your construction is correct, *perfectly hiding, and computationally binding* under the discrete logarithm assumption.**

Solution

$\mathcal{S}(1^\lambda, m \in \mathbb{Z}_q)$ $\qquad\qquad$ $\mathcal{R}(1^\lambda)$

....Commitment..

$p \leftarrow$ first λ-bit prime $\qquad\qquad$ $p \leftarrow$ first λ-bit prime

$g \leftarrow$ random generator of QR_p

$h \leftarrow$ random generator of QR_p

$$\xleftarrow{\quad g, h \quad}$$

check if g, h are generators of QR_p

$d \xleftarrow{R} \mathbb{Z}_q$

$c := (h, g^m h^d) \in (\mathsf{QR}_p)^2$

$$\xrightarrow{\quad c \quad}$$

....Opening..

$$\xrightarrow{\quad m, d \quad}$$

$\mathsf{Verify}(m, c, d)$

parse $c = (h, a)$

$\alpha := g^m h^d$

return yes if $a \stackrel{?}{=} \alpha$

and no otherwise.

Correctness. If the sender commits to $m \in \mathbb{Z}_q$ then $c = (h, \underbrace{g^m h^d}_{a})$ for some random d and where g, h are random generators chosen by the receiver. We then have that if $a = g^m h^d = \alpha$ so the receiver accepts a valid opening.

Perfect hiding. We first prove that the above scheme is perfectly hiding. For our scheme, $\mathsf{view}_{\mathcal{R}}(\langle \mathcal{S}(1^\lambda, m), \mathcal{R}(1^\lambda)\rangle) = (h, a = g^m h^d) \in (\mathsf{QR}_p)^2$. Fix arbitrary messages $m_0, m_1 \in \mathcal{M}$ and malicious, unbounded receiver $\mathcal{R}^*$. For all $h^* \in \mathsf{QR}_p$, let p_{h^*} be the probability that $\mathcal{R}^*$ sends h^* as its first message. Then for any $h^*, a^* \in \mathsf{QR}_p$, we have

$$\begin{aligned}
&\Pr\left[[(c = (h, a), d)_{\mathcal{S}}, (c)_{\mathcal{R}^*}] \leftarrow \langle \mathcal{S}(1^\lambda, m_0), \mathcal{R}^*(1^\lambda)\rangle : (h, a) = (h^*, a^*)\right] \\
&= p_{h^*} \cdot \Pr\left[d \xleftarrow{R} \mathbb{Z}_q : m_0 + \log h^* \cdot d = \log a^*\right] \\
&= p_{h^*} \cdot \Pr\left[d \xleftarrow{R} \mathbb{Z}_q : d = \frac{\log a^* - m_0}{\log h^*}\right] \\
&= p_{h^*} \cdot \Pr\left[d \xleftarrow{R} \mathbb{Z}_q : d = \frac{\log a^* - m_1}{\log h^*}\right] \\
&= \Pr\left[[(c = (h, a), d)_{\mathcal{S}}, (c)_{\mathcal{R}^*}] \leftarrow \langle \mathcal{S}(1^\lambda, m_1), \mathcal{R}^*(1^\lambda)\rangle : (h, a) = (h^*, a^*)\right]
\end{aligned}$$

where we use log to denote the discrete log with base g. Thus we can conclude that

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

Solution (continued...)

Computational binding. We prove that the scheme is computationally binding assuming the hardness of the discrete logarithm problem in QR_q. Suppose, towards contradiction, that the scheme is not computationally binding. Then, there exists a PPT $\mathcal{A}$ (i.e., the sender) such that for some non-negligible function $\delta(\lambda)$ and pair of messages $m_0 \neq m_1 \in \mathcal{M}$:

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

We will use $\mathcal{A}$ to solve the discrete logarithm problem in QR_p. Fix any m and m' and construct PPT $\mathcal{B}$ which breaks the DL assumption as follows.

$\mathcal{B}(\mathsf{QR}_p, g, h = g^x)$

1 : Run $\mathcal{A}(1^\lambda)$ and simulate the receiver $\mathcal{R}$ with input h

2 : $\mathcal{A}$ outputs (c, d, d')

3 : $x' \leftarrow (d' - d) \cdot (m_0 - m_1)^{-1} \bmod q$

4 : **return** x'

Analysis: If $\mathcal{A}$ succeeds in outputting (c, d, d') then $d' = d + x(m - m')$ where $h = g^x$. As such, we have that $x = (d' - d) \cdot (m - m')^{-1} \bmod q$. Therefore, if $\mathcal{A}$ succeeds with probability $\delta(\lambda)$, then $\mathcal{A}$ succeeds in recovering the discrete logarithm of g^x with probability $\delta(\lambda)$, which is non-negligible. By contrapositive, if the discrete logarithm problem is hard, then the scheme is computationally binding.

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, $\mathsf{Gen}, \mathsf{MAC}$ 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.

<u>**Solution**</u>

Suppose towards contradiction that Π is not EUF-CMA secure, so we have some PPT $\mathcal{A}$ which makes some amount of queries to the $\mathsf{MAC}(sk, \cdot)$ oracle, after which it will output (with non-negligible probability) a forgery (m^*, σ^*) such that $\mathsf{Verify}(sk, m^*, \sigma^*) = 1$, i.e., $\sigma^* = h(sk \oplus m^*)$ and m^* was never queried to the $\mathsf{MAC}(sk, \cdot)$ oracle. We will consider two cases.

Case 1. $\mathcal{A}$ learns $sk \oplus m^*$ and queries the random oracle on $sk \oplus m^*$ itself. However, since h is a random oracle, from $\mathcal{A}$'s perspective $sk \oplus m^*$ is a uniformly random value independent from all the outputs $h(sk \oplus m)$ it has seen. Since there are 2^λ possible values for $sk \oplus m^*$ that all occur with equal probability (over the choice of sk), we have

$$\Pr\left[\begin{array}{l} sk \leftarrow \mathsf{Gen}(1^\lambda); \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{MAC}(sk, \cdot)}(1^\lambda) \end{array} : \begin{array}{l} m^* \notin Q; \\ \sigma^* = h(sk \oplus m^*) \end{array}\right] \leq \frac{1}{2^\lambda} = \mathsf{negl}(\lambda).$$

Case 2. $\mathcal{A}$ doesn't learn $sk \oplus m^*$, but manages to guess $h(sk \oplus m)$. However, since h is a random oracle, from $\mathcal{A}$'s perspective $h(sk \oplus m^*)$ is a uniformly random value independent from all the values it has seen. Since there are $2^{\ell(\lambda)}$ possible values for $h(sk \oplus m^*)$ that all occur with equal probability, independent to all the inputs/query responses to $\mathcal{A}$, we have

$$\Pr\left[\begin{array}{l} sk \leftarrow \mathsf{Gen}(1^\lambda); \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{MAC}(sk, \cdot)}(1^\lambda) \end{array} : \begin{array}{l} m^* \notin Q; \\ \sigma^* = h(sk \oplus m^*) \end{array}\right] \leq \frac{1}{2^{\ell(\lambda)}}.$$

Finally, we claim that $\frac{1}{2^{\ell(\lambda)}}$ must be negligible. Suppose towards contradiction that it is not. Selecting any two inputs uniformly at random $x_1, x_2 \in \{0,1\}^\lambda$ gives us a collision with probability $\Pr[h(x_1) = h(x_2)] \geq \frac{1}{2^{\ell(\lambda)}}$. Then if $\frac{1}{2^{\ell(\lambda)}}$ were non-negligible, that would contradict the fact that h is a CRHF. Thus, Π is a secure fixed-length MAC for λ-bit messages, in the random oracle model.

(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.**

$\mathsf{Gen}(1^\lambda)$	$\mathsf{MAC}(sk, m \in \{0,1\}^\lambda)$	$\mathsf{Verify}(sk, m, \sigma)$
1 : $h \xleftarrow{R} \mathcal{H}_\lambda$	1 : $\sigma \leftarrow h(sk \oplus m)$	1 : $t \leftarrow h(sk \oplus m)$
2 : publish h on bulletin board	2 : **return** σ	2 : **if** $\sigma = t$: **return** 1
3 : $sk \xleftarrow{R} \{0,1\}^n$		3 : **else** : **return** 0
4 : **return** sk		

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

Solution

Define $h'(x) = x_1 \mid\mid h(x_{[2:|x|]})$. We claim that h is a CRHF and that Π is not EUF-CMA secure when implemented using h. First we will show that h is collision-resistant. Suppose not, so some PPT adversary $\mathcal{B}$ can produce a collision x, x' for h'. Then

$$h'(x) = h'(x') \implies x_1 \mid\mid h(x_{[2:|x|]}) = x'_1 \mid\mid h(x'_{[2:|x'|]}).$$

We also have that $h(x_{[2:|x|]}) = h(x'_{[2:|x'|]})$ and that $x_{[2:|x|]} \neq x'_{[2:|x'|]}$, since $x_1 = x'_1$ but $x \neq x'$. Thus $x_{[2:|x|]}, x'_{[2:|x'|]}$ is a collision for h, but this contradicts the collision resistance of h, because a PPT adversary $\mathcal{A}$ can simply run $\mathcal{B}$ to get x, x' and return $x_{[2:|x|]}, x'_{[2:|x'|]}$.

Now, we just need to show that Π is not a EUF-CMA secure when implemented using h'. Define the forger $\mathcal{A}$ as follows:

$\mathcal{A}^{\mathsf{MAC}(sk,\cdot)}(1^\lambda)$

$m \xleftarrow{R} \{0,1\}^n, \sigma \leftarrow \mathsf{MAC}_{(sk,m)}$

$m' := \overline{m_1} \mid\mid m_{[2:|m|]}$

$\sigma' := \overline{\sigma_1} \mid\mid \sigma_{[2:|m|]}$

return (m', t')

Clearly this runs in polynomial time. By construction, $m' \neq m$ and so m' has not been queried by to the $\mathsf{MAC}(sk, \cdot)$ oracle. By the definition of h', we have that

$$\begin{aligned} h'(sk \oplus m') &= (sk \oplus m'_1) \mid\mid h'(sk \oplus m'_{[2:|m'|]}) \\ &= (sk \oplus \overline{m_1}) \mid\mid h(sk \oplus m_{[2:|m|]}) \\ &= \overline{(sk \oplus m_1)} \mid\mid h(sk \oplus m_{[2:|m|]}) \\ &= \overline{\sigma_1} \mid\mid \sigma_{[2:|m|]} = \sigma'. \end{aligned}$$

Therefore (m', σ') is a forgery with probability 1, so Π is not EUF-CMA secure.

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:

$\mathsf{Gen}(1^\lambda)$

$x_{1,0}, \ldots, x_{n,0} \xleftarrow{R} \{0,1\}^{\ell_1}$

$x_{1,1}, \ldots, x_{n,1} \xleftarrow{R} \{0,1\}^{\ell_1}$

$sk := (x_{1,0}, \ldots, x_{n,0}, x_{1,1}, \ldots, x_{n,1})$

$vk := (y_{1,0}, \ldots, y_{n,0}, y_{1,1}, \ldots, y_{n,1})$, where $y_{i,c} = f(x_{i,c})$

return (sk, vk)

$\mathsf{Sign}(sk, m \in \{0,1\}^n)$

parse $sk = (x_{1,0}, \ldots, x_{n,0}, x_{1,1}, \ldots, x_{n,1})$

return $\sigma := (x_{1,m_1}, \ldots, x_{n,m_n})$

$\mathsf{Verify}(vk, m \in \{0,1\}^n, \sigma)$

parse $\sigma := (\sigma_1, \ldots, \sigma_n)$

if $\forall i \in [n], f(\sigma_i) \stackrel{?}{=} y_{i,m_i}$: **return** 1

else : **return** 0

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 $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *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.**

Solution

Let $f : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$ be an arbitrary one-way function. Define $f' : \{0,1\}^{\ell_1+1} \to \{0,1\}^{\ell_2}$ by $f'(x) = f(x_{[1:\ell_1]})$. f' is also a OWF: given a preimage x' for $f'(x)$, $x'_{[1:\ell_1]}$ is a preimage for $f(x_{[1:\ell_1]})$, which is a correctly distributed challenge in the OWF game for f.

Lamport signatures with f' are clearly not one-time strongly unforgeable: Given a signature σ for m, the adversary can produce $\sigma_{[1:(\ell_1+1)n-1]} \mid\mid \overline{\sigma_{(\ell_1+1)n}}$, a valid and different signature for m.

(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.)**

Solution

If the OWF is collision-resistant, then it's computationally intractable to find a different preimage, so no adversary can maul the signature. Suppose not, so we have some PPT algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ which can break the one-time strong unforgeability of the Lamport signature scheme with a collision-resistant OWF f. We will construct a PPT adversary $\mathcal{B}$ which uses $\mathcal{A}$ to break the collision-resistance of f as follows:

$\mathcal{B}(1^\lambda, f)$

$x_{1,0}, \ldots, x_{n,0} \xleftarrow{R} \{0,1\}^{\ell_1}$

$x_{1,1}, \ldots, x_{n,1} \xleftarrow{R} \{0,1\}^{\ell_1}$

$vk := (y_{1,0}, \ldots, y_{n,0}, y_{1,1}, \ldots, y_{n,1}),\ \text{where } y_{i,c} = f(x_{i,c})$

$(m, \mathsf{state}) \leftarrow \mathcal{A}_1(vk)$

$\sigma := (x_{1,m_1}, \ldots, x_{n,m_n})$

$\sigma^* \leftarrow \mathcal{A}_2(\sigma, \mathsf{state})$

for $i \in [n]$:

  if $\sigma_i \neq \sigma_i^*$: **return** (σ_i, σ_i^*)

When $\mathcal{A}$ produces $\sigma^* \neq \sigma$ such that $\mathsf{Verify}(vk, m, \sigma^*) = 1$, then $\mathcal{B}$ successfully produces a collision for f, since there must exist some i such that $\sigma_i \neq \sigma_i^*$ and $f(\sigma_i) = f(\sigma_i^*) = y_{i,m_i}$. Since $\mathcal{A}$ forges with nonnegligible probability, $\mathcal{B}$ breaks the collision-resistance of f with nonnegligible probability, but this is a contradiction. Thus the Lamport scheme with a collision-resistant OWF f is one-time strongly unforgeable.

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).**

Solution

We use the honest-verifier ZK proof of not QR from class on the product y_0y_1. Note that since

$$y_0y_1 \notin \mathsf{QR}_N \iff (y_0 \notin \mathsf{QR}_N \wedge y_1 \in \mathsf{QR}_N) \vee (y_0 \in \mathsf{QR}_N \wedge y_1 \notin \mathsf{QR}_N),$$

this is exactly what we want to prove. Proofs of completeness, soundness, and zero-knowledge are the same as from class.

Prover(N, p, q, y_0, y_1)		**Verifier**(N, y_0, y_1)
		$r \xleftarrow{R} \mathbb{Z}_N^*$
		$b \xleftarrow{R} \{0,1\}$
		$z := r^2(y_0y_1)^b \pmod N$
	$\xleftarrow{\quad z \quad}$	
if $z \in QR_N : b' = 0$		
else $: b' = 1$		
	$\xrightarrow{\quad b' \quad}$	
		$b' \stackrel{?}{=} b$

Figure 1: Zero-knowledge proof system for exactly-1-out-of-2 quadratic residues.

(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*.**

Solution

Prover(N, p, q, y_0, y_1)		**Verifier**(N, y_0, y_1)
if $y_0 \notin QR_N : \beta = 0$		
else $\beta = 1$		
$b_\beta \xleftarrow{R} \{0,1\}$		
$r_\beta \xleftarrow{R} \mathbb{Z}_N^*$		
$z_\beta := r_\beta^2 y_\beta^{b_\beta} \pmod N$		
$r_{1-\beta} \xleftarrow{R} \mathbb{Z}_N^*$		
$z_{1-\beta} := r_{1-\beta}^2 y_{1-\beta} \pmod N$		
	$\xrightarrow{z_0, z_1}$	
		$c \leftarrow \{0,1\}$
	$\xleftarrow{c}$	
$\pi_\beta := r_\beta$		
$b_{1-\beta} := c \oplus b_\beta$		
$x_{1-\beta} := \sqrt{y_{1-\beta}} \pmod N$		
$\pi_{1-\beta} := r_{1-\beta} x_{1-\beta}^{1-b_{1-\beta}} \pmod N$		
	$\xrightarrow{b_0, b_1, \pi_0, \pi_1}$	
		$\pi_0^2 \stackrel{?}{=} z_0 y_0^{-b_0} \pmod N$
		$\pi_1^2 \stackrel{?}{=} z_1 y_1^{-b_1} \pmod N$
		$b_0 \oplus b_1 \stackrel{?}{=} c$

Figure 2: Zero-knowledge proof system for at-least-1-out-of-2 quadratic residues.

Solution (continued...)
Completeness.

$$\pi_\beta^2 = r_\beta^2 = (r_\beta^2 y_\beta^{b_\beta}) y_\beta^{-b_\beta} = z_\beta y_\beta^{-b_\beta} \mod N$$
$$\pi_{1-\beta}^2 = (r_{1-\beta} x_{1-\beta}^{1-b_{1-\beta}})^2 = r_{1-\beta}^2 (x_{1-\beta}^2)^{1-b_{1-\beta}} = (r_{1-\beta}^2 y_{1-\beta}) y_{1-\beta}^{-b_{1-\beta}} = z_{1-\beta} y_{1-\beta}^{-b_{1-\beta}} \mod N$$
$$b_0 \oplus b_1 = b_\beta \oplus b_{1-\beta} = b_\beta \oplus (c \oplus b_\beta) = c \oplus (b_\beta \oplus b_\beta) = c$$

Soundness. Suppose $y_0, y_1 \notin \mathsf{QR}_N$. We will show that regardless of the prover's choice of z_0, z_1, with probability $1/2$ (over the choice of c) we have that $z_0 y_0^{-b_0} z_1 y_1^{-b_1} \notin \mathsf{QR}_N$, so one of the verifier's checks will fail.

- Case 1. $z_0 z_1 \in \mathsf{QR}_N$. If $c = 1$, then $b_\gamma = 0 \implies y_\gamma^{-b_\gamma} \in \mathsf{QR}_N$ and $b_{1-\gamma} = 1 \implies y_\gamma^{-b_{1-\gamma}} \notin \mathsf{QR}_N$, so $y_0^{b_0} y_1^{b_1} \notin \mathsf{QR}_N \implies z_0 y_0^{-b_0} z_1 y_1^{-b_1} \notin \mathsf{QR}_N$.
- Case 2. $z_0 z_1 \notin \mathsf{QR}_N$. If $c = 0$, then either $b_0 = b_1 = 0 \implies y_0^{-b_0} y_1^{-b_1} \in \mathsf{QR}_N$ or $b_0 = b_1 = 1 \implies y_0^{-b_0} y_1^{-b_1} \in \mathsf{QR}_N$, so regardless we have $z_0 y_0^{-b_0} z_1 y_1^{-b_1} \notin \mathsf{QR}_N$.

Zero-knowledge. Define our simulator $\mathcal{S}$ as follows:

$\mathcal{S}(N, y_0, y_1)$

1 : $c^* \xleftarrow{R} \{0,1\}$
2 : $b_0 \xleftarrow{R} \{0,1\},\ b_1 \xleftarrow{R} c^* \oplus b_0$
3 : $r_0 \xleftarrow{R} \mathbb{Z}_N^*,\ r_1 \xleftarrow{R} \mathbb{Z}_N^*$
4 : $z_0 := r_0^2 y_0^{b_0}, z_1 := r_1^2 y_1^{b_1}$
5 : send z_0, z_1 to verifier
6 : receive c from verifier
7 : **if** $c \neq c^*$: rewind to line 1
8 : **return** $(z_0, z_1, c, b_0, b_1, \pi_0 := r_0, \pi_1 := r_1)$

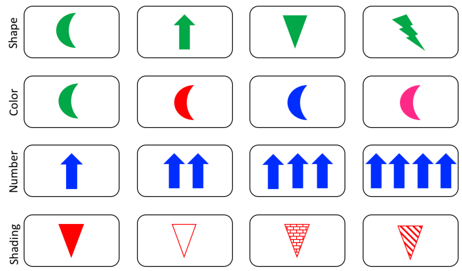
Fix arbitrary y_0, y_1 such that at least one is a quadratic residue $\mod N$. Let $\beta = 0$ if $y_0 \notin \mathsf{QR}_N$ and $\beta = 1$ otherwise (the simulator does not compute this value; we are just using it as notation to show the desired property). Consider the distribution $(z_0, z_1, c, b_0, b_1, \pi_0 := r_0, \pi_1 := r_1) \leftarrow \mathcal{S}(N, y_0, y_1)$.
Since $y_{1-\beta}$. We also have that that $z_\beta = r_\beta^2 y_\beta^{b_\beta}$ exactly as in our protocol. $z_{1-\beta}$ is computed differently by the simulator than in our protocol, but is still correctly distributed uniformly randomly in QR_N. Lastly, we have that $\pi_0^2 = r_0^2 = z_0 y_0^{-b_0}$ and $\pi_1 = r_1^2 = z_1 y_1^{-b_1}$ as in our protocol.

Problem 5. Zero Knowledge Proof System for Set

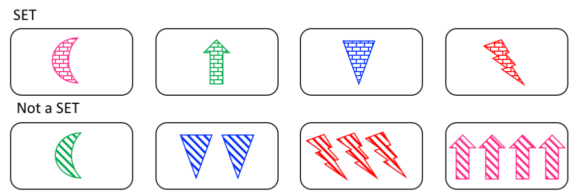
Set[1] is a card game. The object of the game is to identify a SET of n cards from n^2 cards. Each card has n features, and each feature has n possible values. A SET consists of n cards with the property that $\lfloor \frac{n}{2} \rfloor$ out of the n features are the same on each card, and $\lceil \frac{n}{2}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

Design an honest-verifier zero-knowledge proof system for Set, i.e., for the

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



(a) Set Features



(b) Example of a SET and not a SET

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.

Solution

We design a proof system $\Pi = (P, V)$ for the language of solvable Set instances. Let $x = (c_1, \dots c_{n^2})$ be a common Set instance, where each card consists with a list of n features with a numeric value in $[n]$. In other words, for each $i \in [n_2]$, $c_i = (f_1, \dots, f_n)$, and for each $j \in [n]$ $f_j \in [n]$.

1. First message is a commitment to $\tilde{x}$ by the prover P:

   - The prover P selects uniform random $n+1$ permutations with the domain $[n]$. That is,
   $$\pi_0, \dots, \pi_n \leftarrow \{\pi : [n] \overset{1:1}{\to} [n]\}.$$
   - Then, the prover P permute the list of features in each card according to the permutation π_0, and apply to the j-th feature the j-th permutation π_j. That is, each card c_i is permuted to $\tilde{c}_i$ as follows
   $$\tilde{c}_i = (\pi_1(f_{\pi_0(1)}), \dots, \pi_n(f_{\pi_0(n)})).$$
   - Next, the prover selects a uniform permutation with the domain $[n^2]$, $\sigma : [n^2] \overset{1:1}{\to} [n^2]$, and permutes the cards according to it. That is $\tilde{x} = (\tilde{c}_{\sigma(1)}, \dots, \tilde{c}_{\sigma(n^2)})$.
   - Finally, the prover commits (bitwise) to $\tilde{x}$ using a statistically binding commitment scheme.

2. Second message is a uniform random challenge $b \in \{0, 1\}$ by the verifier V

3. Third message is a challenge respond by the prover

   - **Case 1** (challenge message $b = 0$): The prover P's response consists of decommitments to $\tilde{x}$, and all $n+2$ permutations $\pi_0, \dots, \pi_n$ and σ.
   - **Case 2** (challenge message $b = 1$): The prover P's response consists of decommitments to a SET S. Namely, a subset $S \subset \tilde{x}$ such that $|S| = n$, $\lfloor \frac{n}{2} \rfloor$ out of the n features have the same value on all cards in S, and $\lceil \frac{n}{2} \rceil$ out of n of the features have different value on each card in S.

4. The verifier V accepts if

   - **Case 1** (challenge message $b = 0$): The committed $\tilde{x}$ is a permuted version of x according to the $n+2$ permutations that the prover sent in the previous round. That is, $\tilde{x} = (\tilde{c}_{\sigma(1)}, \dots, \tilde{c}_{\sigma(n^2)})$ and $\tilde{c}_i = (\pi_1(f_{\pi_0(1)}), \dots, \pi_n(f_{\pi_0(n)}))$ where $\tilde{c}_i$ is a permuted version of c_i and $x = (c_1, \dots, c_{n^2})$
   - **Case 2** (challenge message $b = 1$): The set S that the prover sent in the previous round is a valid decommitment to a SET according to the Set card game. Thus, this protocol has soundness 1/2.

Solution (continued...)
Next we prove that the proof system (P, V) is complete, sound and zero-knowledge.

Completeness Completeness follows from the fact that the SET property is preserved under the permutation done by the honest prover P.

Soundness Soundness follows from the fact that with all but negligible probability, any first message m_1 by the prover P^* corresponds to *at most* one string x^* under the commitment scheme (by the statistical binding property of the commitment scheme). Since $x \notin L$, we know that either x^* is not a "valid permutation" of x (as defined in the protocol), in which case P^* will fail the challenge $b = 0$, or x^* contains no SET (because any valid permutation of x cannot contain a SET), in which case P^* will fail the challenge $b = 1$.

Zero Knowledge We show that for every PPT algorithm V^* there exists a PPT algorithm S (simulator) such that for all x in the language of solvable Set instances, $\text{view}_{(\text{P},\text{V}^*)}(x)$ and $S(x)$ are computationally indistinguishable.
The simulator S works as follows:

1. Select a uniform random bit $b \in \{0, 1\}$.
   - **Case 1** (b=0) Simulate an honest first message of the honest prover P and send to V^*, i.e., send a commitment to $\tilde{x}$ using a statistically binding commitment scheme where $\tilde{x}$ is a permuted version of x according to $n + 2$ random permutations.
   - **Case 1** (b=1) Send to V^* a commitment to n^2 cards $(c'_1, \ldots, c'_{n^2})$ where a random subset of size n of them is a random SET, and the other cards are zero cards.
2. Get a challenge message b^* from V^*
3. If $b = b^*$, answer and output the transcript. Namely,
   - **Case 1** ($b^* = 0$) Send decommitments to $\tilde{x}$ and $n + 2$ permutations that consists with the first message.
   - **Case 2** ($b^* = 1$) Send decommitments to the random SET within the first message.
4. Else ($b \neq b^*$), rewind (i.e., go to step 1) at most n-times.

Solution (continued...)

We now prove that $S(x) \overset{c}{\approx} \text{view}_{(\mathrm{P},\mathrm{V}^*)}(x)$.

First, we show that for every V^*, in one iteration of the loop, the probability that $b^* = b$ is $1/2 + \mathsf{negl}(\lambda)(n)$, where b is the bit sampled by the simulator and b^* is the output bit of V^* after receiving the first message. If we show this, then we know that the simulator outputs a transcript with high probability, that is $1 - 2^{-n}$.

Assume for contradiction that there exists a PPT algorithm V^* that causes the simulator S to rewind with probability $\mu(n)$, i.e., with input[a] x and first message[b] C^*, output a bit b^* that predict if C^* is a commitment of a permuted version of x or not with probability μ. We construct a PPT algorithm A that uses V^* and breaks the hiding property of the commitment scheme.

- A selects a Set game instance x (i.e., x consists of n^2 cards), and sets m_0 to be a permuted version of x according to $n+2$ random permutations (as done by the honest prover P), and sets m_1 to be a list n^2 cards where a random subset of size n of them is a random SET, and the other cards are zero cards.
- A sends m_0, m_1 to the challenger.
- The challenger send sto A a commitment $C^* = \mathsf{Commit}(m_b)$ for a uniform random b.
- A runs V^* with input x and first message C^*, and gets from V^* a bit $b^* \in \{0,1\}$.
- A output b^*.

Note that A runs V^* with input and first message from the same distribution that the simulator S generates. Therefore, the probability $b^* = b$ is μ, and by the hiding property of the commitment scheme it is $1/2 + \mathsf{negl}(\lambda)(n)$. So, we conclude that the simulator outputs a transcript with probability close to 1. In fact, the above reduction shows that

$$(x, C^*, b^*, \rho_{V^*})_{C^* \leftarrow \mathsf{Commit(m_0)}} \overset{c}{\approx} (x, C^*, b^*, \rho_{V^*})_{C^* \leftarrow \mathsf{Commit(m_1)}} \tag{1}$$

where ρ_{V^*} is the random coins of V^*.

Second, we show that the transcript output by the simulator is computationally indistinguishable from the view of V^* when interact with the honest unbounded prover P.

Case 1(Transcript when $b^* = 0$) In this case S simulates perfectly the honest prover P messages, and so

$$S(x) = (x, C^*, 0, \vec{cr}^{(0)} = (r_1, \ldots, r_{n^2}, \pi_0, \ldots, \pi_n, \sigma)) \equiv \text{view}_{(\mathrm{P},\mathrm{V}^*)}(x).$$

Case 2(Transcript when $b^* = 1$) In this case S simulates a computationally indistinguishable first message (by the commitment hiding property), and third message is a decommitment to a random subset that is a random SET as in the real interaction, and so

$$S(x) = (x, C^*, 1, \vec{cr}^{(1)} = (r_1, \ldots, r_n)) \overset{c}{\approx} \text{view}_{(\mathrm{P},\mathrm{V}^*)}(x).$$

Combining these cases with (1), we conclude that

$$(x, C^*, b^*, \vec{cr}^{(b^*)}, \rho_{V^*})_{C^* \leftarrow \mathsf{Commit(m_0)}} \overset{c}{\approx} (x, C^*, b^*, \vec{cr}^{(b^*)}, \rho_{V^*})_{C^* \leftarrow \mathsf{Commit(m_1)}},$$

completing the proof of indistinguishability.

[a] Input is a Set game instance, a list of n^2 cards.

[b] First message is a commitment of either a permuted version of the input x, or a list of n^2 cards with a random subset of a random SET.