Foundations of Cryptography, Fall 2025
Problem Set 5
Due Monday, December 8

Total Number of Points: 100

Collaboration Policy: Collaboration is permitted and encouraged in small groups of at most three students. You are free to collaborate in discussing answers, but you must write up solutions on your own, and must specify in your submission the names of any collaborators. Do not copy any text from your collaborators; the writeup must be entirely your work. Do not write down solutions on a board and copy it verbatim; again, the writeup must be entirely your own words and your own work and should demonstrate clear understanding of the solution. **Solutions should be typeset in LaTeX.** You may make use of material referenced on the course website, provided that you clearly acknowledge all sources/tools used. Of course, scavenging for solutions from prior years is forbidden.

On the Use of LLMs: You may use AI however you wish to deepen your understanding of the lecture material. Upload the notes, talk to your AI about them, ask for more explanation or examples; it's all fine. **You may not use AI/LLMs in any way to work on your homework.** You may not upload assignments, ask for hints, ask how certain concepts from the lectures might be applied to specific homework problems, or upload your assignments to check for correctness or clarity or anything else. You may not include any AI-generated content whatsoever in your homework submissions. If it becomes clear that you have used an AI tool when working on your homework (either directly by making edits or to ask for hints/solutions), we may mark your grade down to reflect that.

1 Tree-based CRHFs (25 points)

In this problem, we will revisit the question of collision-resistant hash functions. Recall the definition of a CRHF:

Definition 1. *A collision-resistant hash family is associated with a key generation algorithm* Gen *and a set of functions* $\mathcal{H} = \{\mathcal{H}_\lambda\}_{\lambda\in\mathbb{N}}$*, where* $\mathcal{H}_\lambda = \{h_k\}_{k\in\mathcal{K}_\lambda}$ *(where* $\mathcal{K}_\lambda$ *is the keyspace/output space of* Gen*), with the following properties:*

1. ***Efficient:*** Gen *is PPT and for any* $\lambda \in \mathbb{N}$ *and* $k \in \mathcal{K}_\lambda$*,* h_k *is deterministic and computable in polynomial time.*
2. ***Compressing:*** *There exists* $n = n(\lambda) \le \mathrm{poly}(\lambda)$ *and* $m = m(\lambda) < n(\lambda)$ *such that for every* $k \in \mathcal{K}_\lambda$*,* $h_k : \{0,1\}^n \to \{0,1\}^m$*.*
3. ***Collision-resistant:*** *For every polynomial-size adversary* $\mathcal{A}$ *and all* $\lambda \in \mathbb{N}$*,*

$$\Pr_{k\leftarrow\mathsf{Gen}(1^\lambda)}[h_k(x) = h_k(x') \text{ and } x \neq x' : (x,x') = \mathcal{A}(1^\lambda, k)] \le \mathrm{negl}(\lambda).$$

Let $(\mathsf{Gen}, \mathcal{H})$ be a collision-resistant hash family with $n = 2\lambda, m = \lambda$, and let $p = p(\lambda) = \mathrm{poly}(\lambda)$. Construct a collision-resistant hash family $(\mathsf{Gen}, \mathcal{H}' = \{\mathcal{H}'_\lambda\}_{\lambda\in\mathbb{N}})$ with the following properties:

1. For every $\lambda \in \mathbb{N}$ and every $k \in \mathcal{K}_\lambda$, the function $h'_k \in \mathcal{H}'_\lambda$, maps inputs of length $p \cdot \lambda$ to outputs of length λ; i.e.,
$$h'_k : \{0,1\}^{p\cdot\lambda} \to \{0,1\}^\lambda.$$
Moreover, given k, h'_k is an efficiently computable deterministic function.
2. For any index $i \in [p]$ and message $m \in \{0,1\}^{p\cdot\lambda}$, there is a *local opening* of size $O((\log p)\cdot\lambda)$ which suffices to verify the i'th block of m. More formally, there is a pair of poly-time algorithms $(\mathsf{Open}, \mathsf{Ver})$ with the following syntax:
   - Open takes as input a hash key $k \in \mathcal{K}_\lambda$, a message $m \in \{0,1\}^{p\cdot\lambda}$, and an index $i \in [p]$, and outputs a local opening string $L \in \{0,1\}^{O((\log p)\cdot\lambda)}$.
   - Ver takes as input a hash key $k \in \mathcal{K}_\lambda$, a hash value $\mathsf{hash} \in \{0,1\}^\lambda$, a local opening $L \in \{0,1\}^{O(\log p\cdot\lambda)}$, an index $i \in [p]$, and the purported ith message block $\psi \in \{0,1\}^\lambda$, and outputs a bit $v \in \{0,1\}$.

   $(\mathsf{Open}, \mathsf{Ver})$ should satisfy the following properties:

   (a) **Correctness of local opening:** For all $\lambda \in \mathbb{N}$, $k \in \mathcal{K}_\lambda$, $m \in \{0,1\}^{\lambda\cdot p}$, and $i \in [p]$,
   $$\Pr[\mathsf{Ver}(k, \mathsf{hash}, L, i, m_{[(i-1)\lambda+1:i\lambda]}) = 1 : \mathsf{hash} = h_k(m), L = \mathsf{Open}(k, m, i)] = 1.$$

   (b) **Computational binding of local opening:** For all poly-size adversaries $\mathcal{A}$, all indices $i \in [p]$, and all $\lambda \in \mathbb{N}$,
   $$\Pr_{k\leftarrow\mathsf{Gen}(1^\lambda)}\left[{}^{m_0\neq m_1 \ \wedge}_{\forall b\in\{0,1\}, \mathsf{Ver}(k,\mathsf{hash},L_b,i,m_b)=1} : \ (\mathsf{hash}, L_0, m_0, L_1, m_1) = \mathcal{A}(1^\lambda, k)\right] \le \mathrm{negl}(\lambda).$$

Together, these properties allow a prover to succinctly hash a large number of values and open to any one of them with small communication.

Concretely, you should describe $(\mathsf{Gen}, \mathcal{H}', \mathsf{Open}, \mathsf{Ver})$**, prove that** $(\mathsf{Gen}, \mathcal{H}')$ **is a CRHF, and prove that** $(\mathsf{Open}, \mathsf{Ver})$ **is correct and computationally binding.** You may assume that $p > 1$ and that p is a power of 2.

Hint: Look at the problem title for some inspiration on how to design your hash functions!

2 More Secret Sharing Schemes (25 points)

In class, we saw the notion of threshold secret sharing. Given n users (numbered 1 through n), define the (t, n)-threshold access structure to be the collection of sets

$$\mathcal{A}_{\mathsf{threshold}} = \{S \subseteq [n] : |S| \geq t\}.$$

The property of a secret sharing scheme, then, is that when a set $T \subseteq [n]$ of users come together, they can reconstruct the secret from their shares if and only if $T \in \mathcal{A}_{\mathsf{threshold}}$.

In this problem, we will generalize secret sharing to other access structures. In general, an access structure $\mathcal{A}$ is a monotone collection of subsets of $[n]$. Here, monotonicity refers to the condition that if $\mathcal{A}$ contains a set T, it also contains all supersets of T. (Do you see why this has to be the case?)

1. **Adding Weights (10 points).** Let $w \in \mathbb{N}^n$ be an arbitrary *weight vector* with positive integer entries w_i between 1 and poly(n). Show that there is an efficient (i.e. polynomial time) secret sharing scheme with the access structure
$$\mathcal{A}_1 = \{S \subseteq [n] : \sum_{i \in S} w_i \geq t(n)\}.$$

2. **A Majority of Majorities (15 points).** Let $k(n) < n$ be any (polynomial-time computable) function of n, and suppose that the set $[n]$ is partitioned into k sets $S_1, \ldots, S_k$ of size $\frac{n}{k}$. Suppose that we want a subset of players $S \subseteq [n]$ to be able to reconstruct the secret if and only if a *majority* of the sets $S_1, \ldots, S_k$ have *a majority* of players present in S. The corresponding access structure can be described as
$$\mathcal{A}_2 = \{S \subseteq [n] : |\{i \in [n] : |S_i \cap S| > |S_i|/2\}| > k/2\}.$$

   Show that we can efficiently (i.e. in polynomial time) share a secret according to this reconstruction constraint/access structure.

3 Reusable Zero Knowledge (25 points)

In this problem, we will explore the question of constructing multi-theorem NIZKs. Recall that in pset 4, we (essentially) showed that any NIZK argument which is non-adaptive, single-theorem zero-knowledge is also a (non-interactive) non-adaptive, single-theorem witness-indistinguishable argument (NIWI). We also know that NIWI arguments can also be repeated with the same common reference string, giving a non-adaptive, multi-theorem NIWI argument. We will now see how to lift such a NIWI argument to a non-adaptive multi-theorem NIZK argument.[1]

Definition 2. *A non-interactive (computational) zero-knowledge argument (NIZK) for a language $\mathcal{L} \in \mathsf{NP}$ is a triple of PPT algorithms (Gen, P, V) with the following syntax:*

- $\mathsf{Gen}(1^\lambda)$*: Takes as input the security parameter 1^λ and outputs a common reference string* crs*.*
- $P(\mathsf{crs}, x, w)$*: Takes as input* crs*, an instance x, and a corresponding witness w, and outputs a proof π.*
- $V(\mathsf{crs}, x, \pi)$*: Takes as input* crs*, an instance x, and a proof π and outputs a bit 0 or 1, indicating whether it rejects or accepts the proof.*

In addition, (Gen, P, V) should satisfy the following properties:

1. ***Completeness:*** *For every $\lambda \in \mathbb{N}$ and every $(x, w) \in \mathcal{R}_\mathcal{L}$,*

$$\Pr[(V(\mathsf{crs}, x, P(\mathsf{crs}, x, w)) = 1] \geq 1 - \mathrm{negl}(\lambda),$$

   where the probability is over $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$ and over the randomness of the prover P.

2. ***(Adaptive) Soundness:*** *For every poly-size (cheating) prover P^* there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$,*

$$\Pr[x \notin \mathcal{L} \ \wedge \ V(\mathsf{crs}, x, \pi) = 1 : \ (x, \pi) = P^*(\mathsf{crs})] \leq \mu(\lambda),$$

   where the probability is over $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$.

3. ***(Non-Adaptive, Single/Multi-Theorem) Zero-Knowledge:*** *If (Gen, P, V) is single-theorem zero-knowledge, then there exists a PPT simulator $\mathcal{S}$, such that for every polynomial time instance generator algorithm $\mathcal{I}$, that on input 1^λ outputs $(x, w) \in \mathcal{R}_\mathcal{L}$,*

$$(\mathsf{crs}, x, P(\mathsf{crs}, x, w)) \approx S(1^\lambda, x),$$

   where $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$ and $(x, w) \leftarrow \mathcal{I}(1^\lambda)$.

   If (Gen, P, V) is multi-theorem zero-knowledge, then there exists a PPT simulator $\mathcal{S}$, such that for every polynomial $p = p(\lambda)$, every polynomial-time instance generator algorithm $\mathcal{I}$, that on input 1^λ outputs $(x_1, w_1), \ldots, (x_p, w_p) \in \mathcal{R}_\mathcal{L}$,

$$(\mathsf{crs}, \{(x_i, P(\mathsf{crs}, x_i, w_i))\}_{i=1}^p) \approx S(1^\lambda, \{x_i\}_{i=1}^p),$$

   where $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$ and $\{(x_i, w_i)\}_{i=1}^p \leftarrow \mathcal{I}(1^\lambda)$.

Fix any NP-complete language $\mathcal{L}$, and let $G : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be any PRG which stretches inputs of length λ to outputs of length 2λ. Observe that the following language is in NP:

$$\mathcal{L}' = \{(x, y) : x \in \mathcal{L} \vee \exists s \in \{0,1\}^\lambda : G(s) = y\} \in \mathsf{NP}.$$

Suppose $\Pi = (\mathsf{Gen}, P, V)$ is a (non-adaptive) multi-theorem NIWI argument for $\mathcal{L}$. **Construct an argument system $\Pi' = (\mathsf{Gen}', P', V')$ which is a (non-adaptive) multi-theorem NIZK argument for $\mathcal{L}$ and prove that it satisfies all required properties.**

Hint: Try splitting the common reference string into two parts (related to the two part-condition of $\mathcal{L}'$).

[1] We did not explicitly define multi-theorem NIWI, but it is the straightforward definition: for any tuple of instances $(x_1, \ldots, x_p)$ and witnesses $\{w_{i,1}\}_{i=1}^p$, $\{w_{i,2}\}_{i=1}^p$ where $(x_i, w_{i,1}), (x_i, w_{i,2}) \in \mathcal{R}_\mathcal{L}$, $(\mathsf{crs}, \{x_i, P(\mathsf{crs}, x_i, w_{i,1})\}_{i=1}^p)$ is computationally indistinguishable from $(\mathsf{crs}, \{x_i, P(\mathsf{crs}, x_i, w_{i,2})\}_{i=1}^p)$.

4 Committing to Polynomials (25 points)

In this problem, we will explore commitments where the goal is to commit and open to a polynomial $f \in \mathbb{Z}_p[x]$ of bounded degree. We could use a standard commitment scheme to do this, but in some situations, instead of opening to the entire polynomial, we want to open only to its evaluation at some queried point.

Definition 3 (Polynomial Commitment Scheme)**.** *A polynomial commitment scheme consists of four* PPT *algorithms* $\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify}$ *with the following syntax.*

- $\mathsf{Setup}(1^\lambda, d) \to (\mathsf{pk}, \mathsf{sk})$*: takes as input the security parameter and a polynomial degree bound* d *and outputs a private-key secret-key pair.*
- $\mathsf{Commit}(\mathsf{pk}, f) \to \mathsf{com}$*: takes as input a public key and a polynomial* $f \in \mathbb{Z}_p$ *of degree at most* d*, described in terms of its coefficients* $f = (f_0, \ldots, f_d)$*, and produces a commitment* com*.*
- $\mathsf{Open}(\mathsf{pk}, f, u) \to (v, \pi)$*: takes as input* pk, f *and an evaluation point* u*, and outputs the evaluation* v *along with a "proof" that* v *is the evaluation of* f *at* u*.*
- $\mathsf{Verify}(\mathsf{pk}, \mathsf{com}, u, v, \pi) \to \{\top, \bot\}$*: takes as input* $\mathsf{pk}, \mathsf{com}$ *and* u, v, π *and either accepts or rejects.*

These algorithms satisfy the following properties:

- ***Correctness****: For any* $\mathsf{pk} \leftarrow \mathsf{Setup}(^\lambda, t)$*, polynomial* $f \in \mathbb{F}_p$ *with degree at most* d*, for any* $\mathsf{com} \leftarrow \mathsf{Commit}(\mathsf{pk}, f)$*, for any* $u \in \mathbb{F}_p$*, and* $(v, \pi) \leftarrow \mathsf{Open}(\mathsf{pk}, f, u)$*, it must hold that* $v = f(u)$ *and* $\mathsf{Verify}(\mathsf{pk}, \mathsf{com}, u, v, \pi) = \top$ *with probability* 1*.*
- ***Binding****: For all adversaries* $\mathcal{A}$*, there is a negligible function* negl *such that*

$$\Pr_{\substack{\mathsf{pk} \leftarrow \mathsf{Setup}(1^\lambda, d) \\ (\mathsf{com}, u, v, v', \pi, \pi') \leftarrow \mathcal{A}(\mathsf{pk})}} \left[\mathsf{Verify}(\mathsf{pk}, \mathsf{com}, u, v, \pi) = \top \ \wedge \ \mathsf{Verify}(\mathsf{pk}, \mathsf{com}, u, v', \pi') = \top \ \wedge \ v \neq v' \right] \leq \mathrm{negl}(\lambda).$$

Remark 1. *Note that the commitment definition here does not require hiding – this is because polynomial commitments are typically used to construct succinct arguments, where hiding is not always required. The construction we will give allows for a succinct (in fact, single group element!) commitment which is still binding at any particular evaluation point.*

We will now see a construction of such a commitment scheme. First, we introduce some additional structure in the groups that we will be using. Let G be groups of prime order p with generator g. Then group G is called *bilinear* if it is also equipped with an efficiently computable (symmetric) bilinear operation $e : G \times G \to G_T$ that maps pairs $(g_1, g_2) \in G \times G$ to an element of some other group G_T of the same prime order p, such that the following properties hold.[2]

- **Bilinearity**: For every $x, y \in \mathbb{Z}_p$, it holds that $e(g^x, g^y) = e(g, g)^{xy}$.
- **Non-degenerate**: For generator $g \in F$, we have that $e(g, g)$ generates G_T.

Construction. Let $e : G \times G \to G_T$ be an efficiently computable bilinear map for groups G, G_T of prime order p.

- $\mathsf{Setup}(1^\lambda, d)$: Sample a random $\mathsf{sk} := \alpha \leftarrow \mathbb{Z}_p^*$ and generate $\mathsf{pk} := (g, g^\alpha, g^{\alpha^2} \ldots, g^{\alpha^d})$
- $\mathsf{Commit}(\mathsf{pk}, f)$: Parse $\mathsf{pk} = (g_0, \ldots, g_d)$ and $f = (f_0, \ldots, f_d)$, and compute and output

$$\mathsf{com} := \prod_{i=0}^{t} g_i^{f_i} = g^{f(\alpha)}.$$

[2]Technically, all of these groups and orders are indexed by the security parameter λ, but we have omitted them from the notation to avoid clutter.

- $\mathsf{Open}(\mathsf{pk}, f, u)$: Parse $\mathsf{pk} = (g_0, \ldots, g_t)$, and define the polynomial f' and compute π as follows:

$$f'(x) := \frac{f(x) - f(u)}{x - u}$$
$$\pi := \prod_{i=0}^{t} g_i^{f'_i} = g^{f'(\alpha)}.$$

  where $f' = (f'_0, f'_1, \ldots, f'_d)$.

- $\mathsf{Verify}(\mathsf{pk}, \mathsf{com}, u, v, \pi)$: Accept if

$$e(\mathsf{com}, g) = e(\pi, g^{\alpha - u}) \cdot e(g, g)^v$$

  and reject otherwise.

Prove that this scheme satisfies correctness and binding, assuming the following strengthening of the Diffie Hellman assumption holds over G.

Definition 4 (t-strong Diffie Hellman Assumption)**.** *Let $\alpha \leftarrow \mathbb{Z}_p^*$ be a random element. Let g be a generator of the group G. For every poly-size adversary $\mathcal{A}$, there is a negligible function* negl *such that, given as input the tuple $(g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^t}) \in G^{t+1}$, for any $c \in \mathbb{Z}_p \setminus \{-\alpha\}$, it holds that*

$$\Pr_{\alpha \leftarrow \mathbb{Z}_p^*}[\mathcal{A}(g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^t}) \to (c, g^{(\alpha+c)^{-1}})] \leq \mathrm{negl}(\lambda).$$