From Hidden-Bits Generator to NIZKs

Shen Dong

SJTU

March 27, 2024

Outline

Contents

NIZKs in Hidden-bits Model

Definition

A pair of PPT $(\mathcal{P}, \mathcal{V})$ is a non-adaptive NIZK proof system for a language $L \in \mathrm{NP}$ in the "hidden-bits" model when followings hold:

1. Completeness: For all $x \in L$ where $|x| = k$ and its all witnesses w:

$$\Pr\left[b \leftarrow \{0,1\}^{\mathrm{poly}(k)}; (\Pi, I) \leftarrow \mathcal{P}(b, x, w) : \mathcal{V}\left(\{b_i\}_{i \in I}, I, x, \Pi\right) = 1\right] = 1.$$

2. Soundness: $\forall$ (unbounded) $\mathcal{P}^*$, the following is negligible:

$$\Pr\left[b \leftarrow \{0,1\}^{\mathrm{poly}(k)}; (x, \Pi, I) \leftarrow \mathcal{P}^*(b) : \mathcal{V}\left(\{b_i\}_{i \in I}, I, x, \Pi\right) = 1 \wedge x \notin L\right].$$

3. Zero-knowledge: $\exists$ PPT Sim such the following ensembles are computationally indistinguishable for any PPT A :

$$\left\{(x, w) \leftarrow A\left(1^k\right); b \leftarrow \{0,1\}^{\mathrm{poly}\ (k)}; (\Pi, I) \leftarrow \mathcal{P}(b, x, w) : \left(\{b_i\}_{i \in I}, I, x, \Pi\right)\right\}$$

$$\left\{(x, w) \leftarrow A\left(1^k\right); \left(\{b_i\}_{i \in I}, I, \Pi\right) \leftarrow \mathrm{Sim}(x) : \left(\{b_i\}_{i \in I}, I, x, \Pi\right)\right\}$$

NIZKs in CRS Model

Definition

A pair of PPT $(\mathcal{P}, \mathcal{V})$ is a non-adaptive NIZK proof system for a language $L \in \mathrm{NP}$ in the CRS model if followings hold:

1. Completeness: For all $x \in L$ where $|x| = k$ and all witnesses w for x,
$$\Pr\left[r \leftarrow \{0,1\}^{\text{poly }(k)}; \Pi \leftarrow \mathcal{P}(r,x,w) : \mathcal{V}(r,x,\Pi) = 1\right] = 1.$$
2. Soundness: For any (unbounded) algorithm $\mathcal{P}^*$, the following is negligible:
$$\Pr\left[r \leftarrow \{0,1\}^{\text{poly}(k)}; (x,\Pi) \leftarrow \mathcal{P}^*(r) : \mathcal{V}(r,x,\Pi) = 1 \wedge x \notin L\right].$$
3. Zero-knowledge: There exists a PPT algorithm Sim such the following ensembles are computationally indistinguishable for all PPT A :
$$\left\{(x,w) \leftarrow A\left(1^k\right); r \leftarrow \{0,1\}^{\text{poly(k)}}\ ; \Pi \leftarrow \mathcal{P}(r,x,w) : (r,x,\Pi)\right\}$$
$$\left\{(x,w) \leftarrow A\left(1^k\right); (r,\Pi) \leftarrow \mathrm{Sim}(x) : (r,x,\Pi)\right\}$$

Feige-Lapidot-Shamir NIZK for NP

CRS: images of a one-way trapdoor permutation
Hidden-bits string: hard-core bits of the respective pre-images.

$$
\begin{aligned}
&\mathcal{P}\left(r = r_0|\cdots|r_{p(k)}, x, w\right), \quad r_i \in \{0,1\}^k \\
&\quad \left(f, f^{-1}\right) \leftarrow \mathrm{Gen}\left(1^k\right); \\
&\quad \text{For } i = 1 \text{ to } p(k) \text{ do} \\
&\quad\quad b_i = r_0 \cdot f^{-1}\left(r_i\right); \\
&\quad (\Pi, I) \leftarrow \mathcal{P}'\left(b_1 \ldots b_{p(k)}, x, w\right); \\
&\quad \text{Output } \left(\Pi, I, \left\{f^{-1}\left(r_i\right)\right\}_{i\in I}, f\right) \\
&\mathcal{V}\left(r, x, \left(\Pi, I, \{z_i\}_{i\in I}, f\right)\right) \\
&\quad \text{For all } i \in I \\
&\quad\quad \text{If } f\left(z_i\right) = r_i \text{ then} \\
&\quad\quad\quad \text{let } b_i = r_0 \cdot z_i; \\
&\quad\quad \text{else stop and output } 0; \\
&\quad \text{Output } \mathcal{V}'\left(\{b_i\}_{i\in I}, I, x, \Pi\right)
\end{aligned}
$$

FLS: Sketch of Proof

1. completeness: trivially from the completeness of $\mathcal{P}'$ because $\mathcal{P}$ runs $\mathcal{P}'$ as a subroutine.
2. The soundness of $\mathcal{P}'$ is 2^{-2k}, but an unbounded malicious prover might use many trapdoor permutations to find a "bad" f for cheating. Because the generation of f, there are at most 2^k. Soundness bounded by 2^{-k}.
3. Construct a simulator $\mathcal{SIM}$ from $\mathcal{SIM}'$. $(\{b_i\}_{i\in I}, I, \Pi) \leftarrow \mathcal{SIM}'$, so $\mathcal{SIM}$ generate f to computes $r_i,\ i \in I$ and generates $r_j \overset{\$}{\leftarrow} \{0,1\}^k$ for $j \notin I$,output $(r = r_0|\ldots|r_{p(k)}, (\Pi, I, \{z_i\}_{i\in I}, f))$.

Contents

Hidden-bits Generator

Definition

A Hidden-Bits Generator (HBG) is given by a set of PPT algorithms (Setup, GenBits, Verify) satisfying ***statistical binding and computationally hiding****:*

- *Setup* $\left(1^{\lambda}, 1^{k}\right)$ *: Outputs a common reference string crs.*
- *GenBits(crs): Outputs a triple (com,* $r, \{\pi_i\}_{i\in[k]}\Big)$*, where* $r \in \{0,1\}^k$.
- *Verify(crs, com,* $i, r_i, \pi_i)$ *: Outputs accept or reject, where* $i \in [k]$.

Properties

Correctness: We require that for every polynomial $k = k(\lambda)$ and for all $i \in [k]$, we have:

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, \mathsf{com}, i, r_i, \pi_i) = \mathsf{accept} \;:\; \begin{array}{cl} \mathsf{crs} & \leftarrow \mathsf{Setup}(1^\lambda, 1^k) \\ (\mathsf{com}, r, \pi_{[k]}) & \leftarrow \mathsf{GenBits}(\mathsf{crs}) \end{array}\right] = 1.$$

Succinct Commitment: We require that there exists some set $\mathcal{COM}(\lambda)$ and some constant $\delta < 1$ such that $|\mathcal{COM}(\lambda)| \leq 2^{k^\delta \text{ poly}(\lambda)}$, and such that for all crs output by $Setup\left(1^\lambda, 1^k\right)$ and all com output by $GenBits(crs)$ we have com $\in \mathcal{COM}(\lambda)$. Furthermore, we require that for all com $\notin \mathcal{COM}(\lambda)$, $Verify(crs, com, \cdot, \cdot)$ always outputs reject.

Properties

Statistical Binding: $\exists$ (inefficient) deterministic algorithm Open s.t. for every polynomial $k = k(\lambda)$, on input 1^k, crs and com, Open outputs r such that for every (potentially unbounded) cheating prover $\widetilde{\mathcal{P}}$:

$$\Pr\left[\begin{array}{ll} & r_i^* \neq r_i \\ \wedge & \mathsf{Verify}(\mathsf{crs}, \mathsf{com}, i, r_i^*, \pi_i) = \mathsf{accept} \end{array} : \begin{array}{cl} \mathsf{crs} & \leftarrow \mathsf{Setup}(1^\lambda, 1^k) \\ (\mathsf{com}, i, r_i^*, \pi_i) & \leftarrow \widetilde{\mathcal{P}}(\mathsf{crs}) \\ r & \leftarrow \mathsf{Open}(1^k, \mathsf{crs}, \mathsf{com}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Computationally Hiding: We require that for all polynomial $k = k(\lambda)$ and $I \subseteq [k]$, the two following distributions are computationally indistinguishable:

$$(\mathrm{crs}, \mathrm{com}, I, r_I, \pi_I, r_{\bar{I}})$$

$$\overset{\mathrm{c}}{\approx}$$

$$\left(\mathrm{crs}, \mathrm{com}, I, r_I, \pi_I, r'_{\bar{I}}\right), \;\; r' \xleftarrow{\$} \{0,1\}^k$$

Designated-Verifier Hidden-bits Generator

Definition

We define the Designated-Verifier version of a Hidden-Bits Generator (DV-HBG) similarly, but with the following differences:
- Setup $\left(1^{\lambda}, 1^{k}\right)$ *: outputs* (crs, td), *td trapdoor associated to crs;*
- *Verify (crs, td, com,* i, r_i, π_i *) takes the trapdoor td as an additional input, and outputs accept or reject as before;*

Statistical Binding: the cheating prover $\widetilde{\mathcal{P}}$ can now make a polynomial number of oracle queries to $Verify\,(\mathrm{crs}, \mathrm{td}, \cdots)$. $\forall \widetilde{P}$:

$$\Pr\left[\begin{array}{ll} & r_i^* \neq r_i \\ \wedge & \mathsf{Verify}(\mathsf{crs}, \mathsf{td}, \mathsf{com}, i, r_i^*, \pi_i) = \mathsf{accept} \end{array} : \begin{array}{cl} (\mathsf{crs}, \mathsf{td}) & \leftarrow \mathsf{Setup}(1^{\lambda}, 1^{k}) \\ (\mathsf{com}, i, r_i^*, \pi_i) & \leftarrow \widetilde{\mathcal{P}}^{\mathsf{Verify}(\mathsf{crs}, \mathsf{td}, \cdots)}(\mathsf{crs}) \\ r & \leftarrow \mathsf{Open}(1^{k}, \mathsf{crs}, \mathsf{com}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

Computational Hiding: we require indistinguishability given associated td:

$$(\mathrm{crs}, \mathrm{td}, \mathrm{com}, I, r_I, \pi_I, r_{\bar{I}}) \overset{c}{\approx} (\mathrm{crs}, \mathrm{td}, \mathrm{com}, I, r_I, \pi_I, r'_{\bar{I}})$$

where $(\mathrm{crs}, \mathrm{td}) \leftarrow Setup\left(1^{\lambda}, 1^{k}\right), \left(com, r, \pi_{[k]}\right) \leftarrow GenBits(\mathrm{crs}),\ r' \overset{\$}{\leftarrow} \{0,1\}^{k}$.

HBG implies public verifiable NIZK in CRS model

Theorem

Suppose there exists a Hidden-Bits Generator, then there exists a publicly verifiable NIZK.
Suppose there exists a designated-verifier Hidden-Bits Generator (DVHBG), then there exists a reusable designated-verifier NIZK (reusable DV-NIZK).

Construction

Consider the following candidate NIZK $(Setup^{ZK}, \mathcal{P}, \mathcal{V})$ in the CRS model:

1. $Setup^{\text{ZK}}\left(1^\lambda, 1^n\right)$:
$$\begin{aligned} & crs^{\text{BG}} \leftarrow Setup^{\text{BG}}\left(1^\lambda, 1^k\right) \\ & s \xleftarrow{\$} \{0,1\}^k \\ & \text{output: } crs = \left(crs^{\text{BG}}, s\right) \end{aligned}$$
2. $\mathcal{P}(\text{crs}, x, w)$:
$$\begin{aligned} & \left(com, r^{\text{BG}}, \pi_{[k]}\right) \leftarrow GenBits\left(crs^{\text{BG}}\right) \\ & r_i = r_i^{\text{BG}} \oplus s_i \forall i \in [k] \\ & \text{invoke } \left(I \subseteq [k], \pi^{\text{HB}}\right) \leftarrow \mathcal{P}^{\text{HB}}(r, x, w) \\ & \text{output: } \Pi = \left(I, \pi^{\text{HB}}, com, r_I, \pi_I\right) \end{aligned}$$
3. $\mathcal{V}\left(crs, x, \Pi = \left(\left(I, \pi^{\text{HB}}, com, r_I, \pi_I\right)\right)\right)$: $r_i^{\text{BG}} = r_i \oplus s_i,\ \forall i \in [k]$. Accept if $\forall i \in I, Verify\left(crs^{\text{BG}}, com, i, r_i^{\text{BG}}, \pi_i\right)$ accepts, and if $\mathcal{V}^{\text{HB}}\left(I, r_I, x, \pi^{\text{HB}}\right)$ also accepts.

Contents

LWE Trapdoor

- $D_{\mathbb{Z},\sigma}$ as discrete Gaussian distribution over $\mathbb{Z}$ with parameter $\sigma \in \mathbb{R}^+$.
- For a matrix $\mathrm{A} \in \mathbb{Z}_q^{n\times m}$, and a vector $\mathbf{v} \in \mathbb{Z}_q^n$, $\mathbf{A}_\sigma^{-1}(\mathbf{v})$ as a random variable $\mathbf{x} \leftarrow D_{\mathbb{Z},\chi}^m$ conditioned on $\mathrm{Ax} = \mathbf{v} \bmod q$.
- Gaussian Tail Bound: security parameter λ and $\sigma = \sigma(\lambda)$ be a Gaussian width parameter. Then for all polynomials $n = n(\lambda)$, there exists a negligible function $\mathrm{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$:

$$\Pr\left[\|\mathbf{v}\|_\infty > \sqrt{\lambda}\sigma : \mathbf{v} \leftarrow D_{\mathbb{Z},\sigma}^m\right] = \mathrm{negl}(\lambda).$$

- TrapGen $(1^n, q, m) \to (\mathrm{A}, \mathrm{td_A})$: the trapdoor-generation algorithm outputs a matrix $\mathrm{A} \in \mathbb{Z}_q^{n\times m}$ together with a trapdoor $\mathrm{td_A}$.
- SamplePre $(\mathrm{A}, \mathrm{td_A}, \mathbf{v}, s) \to \mathbf{u}$: the preimage-sampling algorithm outputs a vector $\mathbf{u}$ s.t. $Au = v \mod q$ and $u \leftarrow D_{\mathbb{Z},\sigma}^m$.
- Trapdoor quality: The trapdoor $\mathrm{td_A}$ output by TrapGen $(1^n, q, m)$ is a τ-trapdoor where $\tau = O(\sqrt{n \log q \log n})$, meaning max norm of columns.

LWE HBG Construction: Setup

Setup $\left(1^\lambda, 1^k\right) \to \mathrm{crs}$:

1. Run $(\mathbf{A}_i, \mathrm{td}_i) \leftarrow \mathrm{TrapGen}\left(1^n, q, m\right)$ for all $i \in [k]$.
2. Choose random $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n \times L}$.
3. Sample $\mathbf{s}_i \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}_i \xleftarrow{\$} \bar{D}_{\mathbb{Z},\sigma}^m$, $d_i \xleftarrow{\$} \mathbb{Z}_q$ for all $i \in [k]$.
4. Compute $\mathbf{v}_i^\top = \mathbf{s}_i^\top \mathbf{A}_i + \mathbf{e}_i^\top$ for all $i \in [k]$.
5. Sample $\mathbf{W}_i \xleftarrow{\mathrm{R}} \mathrm{SamplePre}(\mathbf{A}_i, \mathrm{td}_i, \mathbf{U}, \sigma)$ for all $i \in [k]$.
6. Output $\mathrm{crs} = \{\mathbf{U}, \mathbf{A}_i, \mathbf{W}_i, \mathbf{v}_i, d_i\}_{i \in [k]}$.

LWE HBG Construction: Genbits

$\mathrm{GenBits}(\mathrm{crs}) \to (\mathrm{com}, \mathbf{r}, (\pi_1, \dots, \pi_k))$:

1. Sample $t \xleftarrow{R} \left[-2^{0.5\lambda}, 2^{0.5\lambda}\right]^L$.
2. Compute $\pi_i = \mathbf{W}_i \mathbf{t}$ for all $i \in [k]$.
3. Set $r_i = \left\lfloor \mathbf{v}_i^\top \pi_i + d_i \right\rceil$ for all $i \in [k]$.
4. Set $\mathrm{com} = \mathbf{U}\mathbf{t}$.
5. Output $(\mathrm{com},\ \mathbf{r},\ (\pi_1, \dots, \pi_k))$.

LWE HBG Construction: Verify

Verify(crs, com, i, β, π) $\to b$:

1. Check if $\|\pi\|_\infty \leq \text{TestBound}$; output 0 if this does not hold, where $\text{TestBound} = \sigma\sqrt{\lambda} \cdot L \cdot 2^{.5\lambda}$.
2. Check if $\text{com} \stackrel{?}{=} \mathbf{A}_i \pi$; output 0 if this does not hold.
3. Check if $\beta \stackrel{?}{=} \lfloor \mathbf{v}_i^\top \pi + d_i \rceil$; output 0 if does not hold.
4. Check if $\beta \stackrel{?}{=} \lfloor \mathbf{v}_i^\top \pi + d_i \pm \text{RoundingBound} \rceil$; output 1 if both hold and 0 otherwise.

Here we set $\text{TestBound} = \sigma\sqrt{\lambda} \cdot L \cdot B$, $\text{RoundingBound} = \sigma\sqrt{\lambda} \cdot m \cdot \text{TestBound}$.

Binding Security

We set $\text{TestBound} = \sigma\sqrt{\lambda} \cdot L \cdot B$, $\text{RoundingBound} = \sigma\sqrt{\lambda} \cdot m \cdot \text{TestBound}$.

$$\tau = O(\sqrt{n \log q \log n}) = O(\sqrt{n\lambda \log n}, \ \sigma \geq \tau \cdot \omega(\sqrt{\log n})$$
$$\therefore \mathbf{W}_i \leq \sigma\sqrt{\lambda}, |\mathbf{t}| \leq B, \text{length } = L.$$

$$\mathbf{v}_i^\top \pi_i + d_i = \left(\mathbf{s}_i^\top \mathbf{A}_i + \mathbf{e}_i^\top\right) \pi_i + d_i = \mathbf{s}_i^\top \mathbf{A}_i \pi_i + \mathbf{e}_i^\top \pi_i + d_i = \mathbf{s}_i^\top \operatorname{com} + \mathbf{e}_i^\top \pi_i + d_i$$

Checks if the output flips in the range of $\pm$RoundingBound.
Correctness error is with negligible probability.

Single Bit Hiding

$\text{SetupHiding}\left(1^\lambda, 1^k\right) \to \text{crs}$:

1. Run $(\mathbf{A}'_i, \text{td}_i) \leftarrow \text{TrapGen}\left(1^{n+1}, q, m\right)$ and parse $\mathbf{A}'_i = \begin{bmatrix} \mathbf{A}_i \\ \mathbf{v}_i^\top \end{bmatrix}$ where $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{v}_i$ is a vector for all $i \in [k]$.
2. Choose random $\mathbf{U} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times L}$.
3. Sample $\mathbf{u}_i \xleftarrow{\text{R}} \mathbb{Z}_q^L$ for all $i \in [k]$ and $d_i \xleftarrow{\text{R}} \mathbb{Z}_q$ for all $i \in [k]$.
4. Set $\mathbf{U}'_i = \begin{bmatrix} \mathbf{U} \\ \mathbf{u}_i^\top \end{bmatrix}$ for all $i \in [k]$.
5. Sample $\mathbf{W}_i \xleftarrow{\text{R}} \text{SamplePre}\left(\mathbf{A}'_i, \text{td}_i, \mathbf{U}'_i, \sigma\right)$ for all $i \in [k]$.
6. Output $\text{crs} = \{\mathbf{U}, \mathbf{A}_i, \mathbf{W}_i, \mathbf{v}_i, d_i\}_{i \in [k]}$.

Indistinguishability

The single hiding bit model setup can be proved indistinguishable with a sequence of games which each is negligibly close to the adjacent one.

Lemma

(Smudging Lemma) Let B_1, B_2 *be two polynomials over the integers and let* $D = \{D(\lambda)\}_\lambda$ *be any* B_1*-bounded distribution family. Let* $U = \{U(\lambda)\}_\lambda$ *and* $U(\lambda)$ *denote the uniform distribution over integers* $[-B_2(\lambda), B_2(\lambda)]$*. The family of distributions* D *and* U *is statistically indistinguishable,* $D + U \approx_s U$*, if there existsa negligible negl(* $\cdot$*) such that for all* $\lambda \in \mathbb{N}, B_1(\lambda)/B_2(\lambda) \leq \mathrm{negl}(\lambda)$*.*

Hiding Security

Assume$\mathbf{c}$ s.t. $\mathbf{W}_i\mathbf{c} = \mathbf{0}$ for all $i \neq i^*$, but $\mathbf{u}_{i^*}^\top \mathbf{c} = \lfloor q/2 \rfloor$.
The challenger in the GenBits algorithm uses $\mathbf{t} + \delta\mathbf{c}$ instead of just $\mathbf{t}$ as before, $\delta \xleftarrow{\$} \{0,1\}$, $\mathbf{t} \xleftarrow{\$} \left[-2^{0.5\lambda}, 2^{0.5\lambda}\right]$ which are statistically close.
$\delta = 0/1$? for $i \neq i^*$ the same, but the output value $\mathbf{r}_{i^*}$ will either be flipped or not depending on whether δ is 0 or 1 .

$$\begin{aligned}
\left\lfloor \mathbf{v}_{i^*}^\top \pi_{i^*} + d_{i^*} \right\rceil &= \left\lfloor \mathbf{v}_{i^*}^\top \mathbf{W}_{i^*}(\mathbf{t} + \delta\mathbf{c}) + d_{i^*} \right\rceil \\
&= \left\lfloor \mathbf{u}_{i^*}^\top (\mathbf{t} + \delta\mathbf{c}) + d_{i^*} \right\rceil \\
&= \left\lfloor \mathbf{u}_{i^*}^\top \mathbf{t} + d_{i^*} + \delta\lfloor q/2 \rfloor \right\rceil \\
&= \left\lfloor \mathbf{u}_{i^*}^\top \mathbf{t} + d_{i^*} \right\rceil \oplus \delta
\end{aligned}$$

The last equation holds with all but $1/q$ probability.
Since δ does not appear anywhere else in the proof, the bit is hidden.

Estabilish $\mathbf{c}$

There exists a sequence of vectors $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{5\lambda} \in \{-1, 0, 1\}^L$ and unique indices ind $_0$, ind $_1$, ..., ind $_{5\lambda} \in [k]$ with the following properties:

- $\mathbf{W}_i \mathbf{h}_j = \mathbf{0}$ for all $i \neq i^*$
- For all $j \in [0, 5\lambda]$ we have $\mathbf{h}_j$ [ind $_j$] $= 1$
- For all $j, j' \in [0, 5\lambda]$ where $j' > j$ we have $\mathbf{h}_{j'}$ [ind $_j$] $= 0$.
- The vectors $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{5\lambda}$ are linearly independent.

FindVectors$\Big(\{\mathbf{W}_i\}_{i \neq i^*}\Big)$

1. Initialize set $S = \emptyset$.
2. For $j = 0$ to 5λ
   - Let $\mathbf{h}_j$ be the lexiographically first vector in $\{-1, 0, 1\}^L$ such that: (1) $\mathbf{W}_i \mathbf{h}_j = \mathbf{0}$ for all $i \neq i^*$, (2) at least one entry of $\mathbf{h}_j$ is 1 and (3) for all $z \in S$ we have $\mathbf{h}_j[z] = 0$.
   - Set ind$_j$ to be the smallest $z \in [L]$ such that $\mathbf{h}_j = 1$.
   - Add the index ind in $_j$ to the set S.
3. Output the sequence $\mathbf{h}_0, \ldots, \mathbf{h}_{5\lambda}$ and ind$_0$, ..., ind$_{5\lambda}$

References

Reusable Designated-Verifier NIZKs for all NP from CDH
https://eprint.iacr.org/2019/235
A New Approach for Non-Interactive Zero-Knowledge from Learning with Errors https://eprint.iacr.org/2024/340