NTNU | Norwegian University of Science and Technology

Concurrently Secure Blind Lattice Signatures from Raccoon

Lucjan Hanzlik, Mark Manulis, Marzio Mula, Alan Pulval-Dady,
Tjerand Silde, and Daniel Slamanig

IACR ePrint 2026/1084

Blind Signatures

- A two-party signing protocol between a Signer and a User

- The Signer does not learn the message they are signing
  - Blindness
- The Signer can not link a signature to any given session
  - Unlinkability
- The User does not learn the signing key being used to sign
  - Unforgeability

Applications

- Blind signatures can be used as anonymous credentials
- There are also constructions for e-cash and e-voting
- A DLOG version is used in Privacy Pass by Cloudflare
- An RSA version is used for private VPNs by Google
- Important for privacy in the new EU Digital Identity Wallet

Concurrent Security

- Blind signatures are often implemented in systems where many users can initiate signing sessions at the same time
- To ensure unforgeability when opening many sessions with a single malicious party, we need concurrent security
- Some schemes are only proven secure in the sequential setting, limiting the signer to one session at any given time

Blind Schnorr

Wallet Databases with Observers

(Extended Abstract)

David Chaum
CWI
The Netherlands

Torben Pryds Pedersen*
Aarhus University
Denmark

CRYPTO 1992

Blind Schnorr

$\text{User}(X, m)$		$\text{Server}(x)$
		$\bar{r} \leftarrow\!\!\$\ \mathbb{Z}_p$
		$\bar{R} := \bar{r}G$
	$\xleftarrow{\bar{R}}$	
$\alpha, \beta \leftarrow\!\!\$\ \mathbb{Z}_p$		
$R := \bar{R} + \alpha G + \beta X$		
$c := \mathrm{H}(R, m)$		
$\bar{c} := c + \beta$		
	$\xrightarrow{\bar{c}}$	
		$\bar{s} := \bar{c}x + \bar{r}$
	$\xleftarrow{\bar{s}}$	
check $\bar{s}G \stackrel{?}{=} \bar{c}X + \bar{R}$		
$s := \bar{s} + \alpha$		
return (R, s)		

The ROS Attack

On the (in)security of ROS

Fabrice Benhamouda[1], Tancrède Lepoint[2], Julian Loss[3], Michele Orrù[4], and Mariana Raykova[5]

EUROCRYPT 2021, ePrint 2020/945

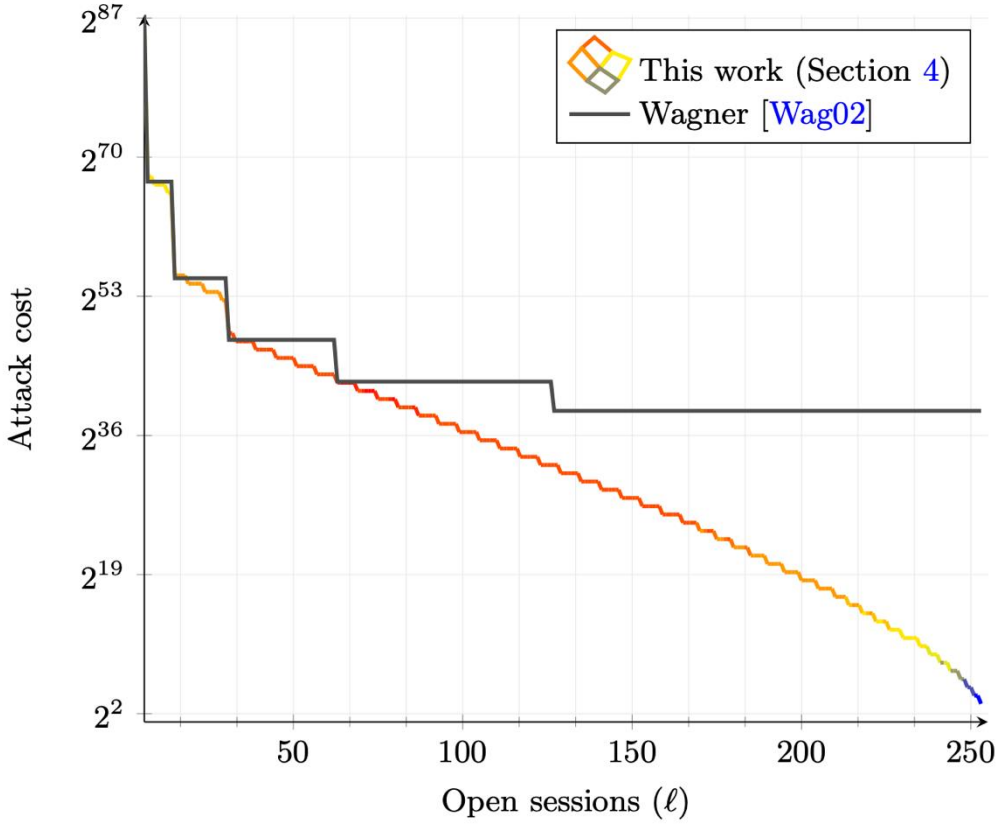
The ROS Attack

which is defined as follows. Given a prime number p and access to a random oracle $\mathrm{H}_{\mathrm{ros}}$ with range in $\mathbb{Z}_p$, the ROS problem (in dimension ℓ) asks to find $(\ell+1)$ vectors $\hat{\boldsymbol{\rho}}_i \in \mathbb{Z}_p^\ell$ for $i \in [\ell+1]$, and a vector $\mathbf{c} = (c_1, \ldots, c_\ell)$ such that:

$$\mathrm{H}_{\mathrm{ros}}(\hat{\boldsymbol{\rho}}_i) = \langle \hat{\boldsymbol{\rho}}_i, \mathbf{c} \rangle \qquad \text{for all } i \in [\ell+1].$$

This problem was originally studied by Schnorr [Sch01] in the context of blind signature schemes. Using a solver for the ROS problem, Wagner showed that the unforgeability of the Schnorr and Okamoto–Schnorr blind signature schemes can be attacked in subexponential time whenever more than $\mathsf{polylog}(\lambda)$ signatures are issued concurrently. In this work, we revisit the ROS problem and its applications. We make the following

The ROS Attack



The Clause Blind Schnorr

Blind Schnorr Signatures and Signed ElGamal Encryption in the Algebraic Group Model

Georg Fuchsbauer[1], Antoine Plouviez[2], and Yannick Seurin[3]

EUROCRYPT 2020, ePrint 2019/877

The Clause Blind Schnorr

$\mathsf{CBISch.Sign}((p, \mathbb{G}, G, \mathsf{H}), x)$		$\mathsf{CBISch.User}(((p, \mathbb{G}, G, \mathsf{H}), X), m)$
$r_0, r_1 \leftarrow\!\!\$\ \mathbb{Z}_p$ $R_0 := r_0 G\,; R_1 := r_1 G$	$\xrightarrow{R_0, R_1}$	$\alpha_0, \beta_0, \alpha_1, \beta_1 \leftarrow\!\!\$\ \mathbb{Z}_p$ $R'_0 := R_0 + \alpha_0 G + \beta_0 X$ $R'_1 := R_1 + \alpha_1 G + \beta_1 X$ $c'_0 := \mathsf{H}(R'_0, m)$ $c'_1 := \mathsf{H}(R'_1, m)$ $c_0 := c'_0 + \beta_0 \bmod p$ $c_1 := c'_1 + \beta_1 \bmod p$
$b \leftarrow\!\!\$\ \{0,1\}$ $s := r_b + c_b x \bmod p$	$\xleftarrow{c_0, c_1}$ $\xrightarrow{b, s}$	**if** $sG \neq R_b + c_b X$ **then return** $\perp$ $s' := s + \alpha_b \bmod p$
return 1		**return** $\sigma := (R'_b, s')$

Concurrently Secure Blind Schnorr

Concurrently Secure Blind Schnorr Signatures

Georg Fuchsbauer and Mathias Wolf

EUROCRYPT 2024, ePrint 2022/1676

Concurrently Secure Blind Schnorr

$\underline{\mathsf{PBSch.Sign}(sk, prd)}$		$\underline{\mathsf{PBSch.User}(vk, prd, m)}$
$(q, \mathbb{G}, G, crs, ek, x) :\subseteq sk$		$(q, \mathbb{G}, G, \mathsf{H}, crs, ek, X) :\subseteq vk$
		$\alpha, \beta \leftarrow\!\!\$\ \mathbb{Z}_q\ ;\ \rho \leftarrow\!\!\$\ \mathcal{R}_{ek}$
	$\xleftarrow{C}$	$C := \mathsf{PKE.Enc}(ek, (m, \alpha, \beta); \rho)$
$r \leftarrow\!\!\$\ \mathbb{Z}_q\ ;\ R := rG$	$\xrightarrow{R}$	$R' := R + \alpha G + \beta X$
		$c := (\mathsf{H}(R', X, m) + \beta) \bmod q$
		$\theta := (X, R, c, C, prd, ek)$
		$w := (m, \alpha, \beta, \rho)$
	$\xleftarrow{c, \pi}$	$\pi \leftarrow \mathsf{NArg.Prove}(crs, \theta, w)$
$\theta := (xG, R, c, C, \mathrm{prd}, ek)$		
if $\mathsf{NArg.Vfy}(crs, \theta, \pi) = 0$:		
return 0		
$s := (r + cx) \bmod q$	$\xrightarrow{s}$	**if** $sG \neq R + cX$: **return** $\perp$
		$s' := (s + \alpha) \bmod q$
return 1		**return** $\sigma := (R', s')$

Challenges for Lattices

- Rejection sampling; get the right distributions
- Norm bounds; ensure hardness of MSIS/MLWE
- Masking values; Gaussian/small, not uniform
- Zero-knowledge proofs; more complex vs DLOG

Usual Approaches

- Design a new ad-hoc blind identification scheme
- Fischlin-transform: com + sign + zkp of sign
- New ad-hoc assumptions (f-ISIS, one-more ...)
- Compute a blind version of a normal signature

Raccoon Lattice Signature Scheme

Raccoon

A Side-Channel Secure Signature Scheme

Rafael del Pino
PQShield, FR

Thomas Espitau
PQShield, FR

Shuichi Katsumata
PQShield, UK
AIST, JP

Mary Maller
PQShield, UK
Ethereum Foundation, UK

Fabrice Mouhartem
PQShield, FR

Thomas Prest
PQShield, FR

Mélissa Rossi
ANSSI, FR

Markku-Juhani Saarinen
PQShield, UK
Tampere University, FI

Raccoon Lattice Signature Scheme

Schnorr.Keygen() $\to$ sk, vk

1. Sample uniform sk, set $\mathsf{vk} = g^{\mathsf{sk}}$

Schnorr.Sign(sk, msg) $\to$ sig

1. Sample r
2. $w = g^r$
3. $c = H(w, \mathtt{msg})$
4. $z = r + c \cdot \mathsf{sk}$
5. Output $\mathtt{sig} = (c, z)$

Schnorr.Verify(vk, msg, sig)

1. $w' = g^z \cdot \mathsf{vk}^{-c}$
2. Assert $H(\mathbf{w}', \mathtt{msg}) = c$

Raccoon.Keygen() $\to$ sk, vk

1. Sample short sk, set $\mathsf{vk} = \begin{bmatrix}\mathbf{A} & 1\end{bmatrix} \cdot \mathsf{sk}$

Raccoon.Sign(sk, msg) $\to$ sig

1. Sample a short $\mathbf{r}$
2. $\mathbf{w} = \begin{bmatrix}\mathbf{A} & 1\end{bmatrix} \cdot \mathbf{r}$
3. $c = H(\mathbf{w}, \mathtt{msg})$
4. $\mathbf{z} = \mathbf{r} + c \cdot \mathsf{sk}$
5. Output $\mathtt{sig} = (c, \mathbf{z})$

Raccoon.Verify(vk, msg, sig)

1. $\mathbf{w}' = \begin{bmatrix}\mathbf{A} & 1\end{bmatrix} \cdot \mathbf{z} - c \cdot \mathsf{vk}$
2. Assert $H(\mathbf{w}', \mathtt{msg}) = c$

BRaccoon

Signer($\mathsf{vk} = (\mathbf{A}, \mathbf{t} = \mathbf{As}), \mathsf{sk} = \mathbf{s}, \mathsf{ek}$)		User($\mathsf{vk} = (\mathbf{A}, \mathbf{t}), \mathsf{msg}, \mathsf{ck}, \mathsf{ek}, \mathsf{dk}$)
		Sample bounded $\mathbf{r} \leftarrow\!\!\$\ \mathsf{D}_{\mathbf{w}}$;
		$\mathsf{C}_{\mathsf{msg},\mathbf{r}} \leftarrow \mathsf{Com}(\mathsf{ck}, \mathbf{r}, \mathsf{msg})$
	$\mathsf{C}_{\mathsf{msg},\mathbf{r}}$ ⟵	
Abort if π_1 **does not verify**		
Sample bounded $\mathbf{r}' \leftarrow\!\!\$\ \mathsf{D}_{\mathbf{w}'}$; $\mathbf{w}' \leftarrow \mathbf{Ar}'$		
	$\mathbf{w}'$ ⟶	
		$\mathbf{w} \leftarrow \mathbf{w}' + \mathbf{Ar}$; $c \leftarrow \mathsf{H}_c(\mathsf{vk}, \mathbf{w}, \mathsf{msg})$
		$\mathsf{ctx}_c \leftarrow \mathsf{Enc}_1(\mathsf{ek}, c)$;
		Compute NIZK π_1
	ctx_c, π_1 ⟵	
Abort if π_1 **does not verify**		
$\mathsf{ctx}_{\mathbf{z}'} \leftarrow \mathsf{Enc}_2(\mathsf{ek}, \mathbf{r}') + \mathsf{ctx}_c \cdot \mathbf{s}$;		
Compute NIZK π_2		
	$\mathsf{ctx}_{\mathbf{z}'}, \pi_2$ ⟶	
		Abort if π_2 **does not verify**
		$\mathbf{z}' \leftarrow \mathsf{Dec}(\mathsf{dk}, \mathsf{ctx}_{\mathbf{z}'})$; $\mathbf{z} \leftarrow \mathbf{z}' + \mathbf{r}$
		Output signature $\mathsf{sig} = (c, \mathbf{z})$

Lattice Assumptions

Definition 3 (**Hint-**MLWE (H-MLWE) [KLSS23]). *Let k, ℓ, Q be positive integers, D_1 and D_2 be probability distributions over $\mathcal{R}_q$, and C be a subset of $\mathcal{R}_q$. The Hint-*MLWE *problem* $\mathsf{H\text{-}MLWE}_{k,\ell,\mathsf{D}_1,\mathsf{D}_2,Q}$ *then asks an adversary $\mathcal{A}$ to distinguish between the following two cases:*

1. $(\mathbf{A}, \mathbf{As}, (c_i, \mathbf{z}_i)_{i\in[Q]})$ *for* $\mathbf{A} \leftarrow \mathcal{R}_q^{k\times(\ell+k)}$
2. $(\mathbf{A}, \mathbf{b}, (c_i, \mathbf{z}_i)_{i\in[Q]})$ *for* $\mathbf{A} \leftarrow \mathcal{R}_q^{k\times(\ell+k)}, \mathbf{b} \leftarrow \mathcal{R}_q^{k\times(\ell+k)}$,

where $\mathbf{s} \leftarrow \mathsf{D}_1^{\ell+k}, c_i \leftarrow C$ *for* $i \in [Q]$, *and* $\mathbf{z}_i := c_i \cdot \mathbf{s} + \mathbf{y_i}$ *where* $\mathbf{y}_i \leftarrow \mathsf{D}_2^{\ell+k}$ *for* $i \in [Q]$.

Definition 4 (SelfTargetMSIS [DKL+18]). *Let k, ℓ be positive integers and $0 < \eta \ll q$. Let* $\mathsf{H}\colon \mathcal{R}_q^k \times \{0,1\}^* \to \mathcal{C}$ *be a cryptographically secure hash function modeled as a random oracle. Then, given* $(\mathbf{A}, \mathbf{t}) \in \mathcal{R}_q^{k\times\ell} \times \mathcal{R}_q^k$, *the Self-Target MSIS problem asks an adversary $\mathcal{A}$ to find* $(\mathbf{z}, \mu) \in \mathcal{R}_q^\ell \times \{0,1\}^*$ *such that* $0 < \|\mathbf{z}\|_2 \le \eta$ *and* $\mathsf{H}([\mathbf{A} \mid \mathbf{t}] \cdot \mathbf{r}, \mu) = c$, *where* $\mathbf{r} = [\mathbf{z}\ c]^\intercal$. *$\mathcal{A}$ is said to have advantage* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{SelfTargetMSIS}}(\lambda)$ *in solving* $\mathsf{SelfTargetMSIS}_{k,\ell,\eta}$ *if*

$$\Pr\left[\begin{matrix}\mathsf{H}([\mathbf{A} \mid \mathbf{t}] \cdot \mathbf{r}, \mu) = c \\ \wedge\ 0 < \|\mathbf{z}\|_2 \le \eta\end{matrix} \;\middle|\; \mathbf{A} \leftarrow \mathcal{R}_q^{k\times\ell};\ \left(\mathbf{r} = \begin{bmatrix}\mathbf{z}\\ c\end{bmatrix}, \mu\right) \leftarrow \mathcal{A}^{\mathsf{H}}(\mathbf{A}, \mathbf{t})\right] \ge \mathsf{Adv}_{\mathcal{A}}^{\mathsf{SelfTargetMSIS}}(\lambda).$$

Raccoon*

$\mathsf{User}_1(\mathsf{pp}, \mathsf{vk}, \mathsf{msg}, \mathsf{ek}, \mathsf{dk})$

1: $(\mathbf{r}, \mathbf{e}) \leftarrow \mathsf{D}_{\mathbf{w}}^{\ell} \times \mathsf{D}_{\mathbf{w}}^{k}$
2: $C_{\mathsf{msg},\mathbf{r},\mathbf{e}} := \mathsf{CS.Com}(\mathsf{pp}, (\mathbf{r}, \mathbf{e}, \mathsf{msg}))$
3: $\mathsf{state}_{\mathsf{U}} := (\mathbf{r}, \mathbf{e}, \mathsf{msg}, \mathsf{vk}, \mathsf{ek}, \mathsf{dk})$
4: **return** $(C_{\mathsf{msg},\mathbf{r},\mathbf{e}}, \mathsf{state}_{\mathsf{U}})$

$\mathsf{Signer}_1(\mathsf{pp}, \mathsf{ek}, \mathsf{sk}, \mathsf{vk}, C)$

1: $(\mathbf{r}', \mathbf{e}') \leftarrow \mathsf{D}_{\mathbf{w}'}^{\ell} \times \mathsf{D}_{\mathbf{w}'}^{k}$
2: $\mathbf{w}' := \mathbf{A}\mathbf{r}' + \mathbf{e}'$
3: $\mathsf{state}_{\mathsf{S}} := (\mathsf{ek}, \mathsf{sk}, \mathsf{vk}, \mathbf{r}', \mathbf{e}')$
4: **return** $(\mathbf{w}', \mathsf{state}_{\mathsf{S}})$

$\mathsf{User}_2(\mathsf{pp}, \mathsf{state}_{\mathsf{U}}, \mathbf{w}')$

1: $\mathbf{w} := \lfloor \mathbf{w}' + \mathbf{A}\mathbf{r} + \mathbf{e} \rceil_{\nu_{\mathbf{w}}}$
2: $\mu := \mathsf{H}(\mathsf{vk}, \mathsf{msg})$
3: $c := \mathsf{H}_c(\mu, \mathbf{w})$
4: $\mathsf{ctx}_c := \mathsf{HE.Enc}(\mathsf{pp}, \mathsf{ek}, c)$
5: $\pi_1 := \mathsf{NIZK}_{\mathsf{L}_1}.\mathsf{Prove}(\mathsf{pp}, \mathsf{msg}, \mathbf{r}, \mathbf{e}, \mathbf{w}, \mu)$
6: $\mathsf{state}_{\mathsf{U}} := \mathsf{state}_{\mathsf{U}} || (c, \mathbf{w})$
7: **return** $(\mathsf{state}_{\mathsf{U}}, \pi_1, \mathsf{ctx}_c)$

$\mathsf{Signer}_2(\mathsf{pp}, \mathsf{state}_{\mathsf{S}}, \pi_1, \mathsf{ctx}_c)$

1: **if** $\mathsf{NIZK}_{\mathsf{L}_1}.\mathsf{Verify}(\pi_1) = 0$ **then**
2: **abort**
3: $\mathsf{ctx}_{\mathbf{r}'} := \mathsf{HE.Enc}(\mathsf{pp}, \mathsf{ek}, \mathbf{r}')$
4: $\mathsf{ctx}_{\mathbf{z}'} := \mathsf{HE.Eval}(\mathsf{pp}, \mathsf{ek}, f, (\mathsf{ctx}_c, \mathbf{s}, \mathsf{ctx}_{\mathbf{r}'}))$
5: $\pi_2 := \mathsf{NIZK}_{\mathsf{L}_2}.\mathsf{Prove}(\mathsf{pp}, \mathbf{r}', \mathbf{e}', \mathbf{s}, \mathbf{e}_{\mathsf{kg}})$
6: **return** $(\pi_2, \mathsf{ctx}_{\mathbf{z}'})$

$\mathsf{User}_3(\mathsf{pp}, \mathsf{state}_{\mathsf{U}}, \pi_2, \mathsf{ctx}'_z)$

1: **if** $\mathsf{NIZK}_{\mathsf{L}_2}.\mathsf{Verify}(\pi_2) = 0$ **then**
2: **abort**
3: $\mathbf{z}' := \mathsf{HE.Dec}(\mathsf{pp}, \mathsf{dk}, \mathsf{ctx}_{\mathbf{z}'})$
4: $\mathbf{z} := \mathbf{z}' + \mathbf{r}$
5: $\mathbf{y} := \lfloor \mathbf{A}\mathbf{z} - 2^{\nu_t} c\mathbf{t} \rceil_{\nu_{\mathbf{w}}}$
6: $\mathbf{h} := \mathbf{w} - \mathbf{y}$
7: **return** $\mathsf{sig} = (c, \mathbf{z}, \mathbf{h})$

Security Analysis

Theorem 4 (One-More Unforgeability). *Consider an adversary $\mathcal{A}$ breaking the one-more unforgeability of our protocol* BRaccoon *in Figure 1. Then there exist PPT adversaries $\mathcal{R}, \mathcal{B}, \mathcal{C}, \mathcal{D}$, $\mathcal{E}, \mathcal{V}$ such that $\mathcal{R}$ breaks the* EUF-CMA *security of* Raccoon* *with $Q_s(\lambda)$ queries to the signing oracle $\mathcal{O}^{\mathsf{Raccoon}^\star}$, $\mathcal{B}$ breaks the setup indistinguishability of the commitment scheme* CS*, $\mathcal{C}$ breaks the extractability of* CS*, $\mathcal{D}$ breaks the zero-knowledge property of π_2, $\mathcal{E}$ breaks the soundness of π_1, and $\mathcal{V}$ breaks the circuit privacy of* HE*. In particular, for any security parameter λ, we have:*

$$
\begin{aligned}
\mathcal{A}^{\mathsf{BRaccoon}}_{\mathrm{OMUF}}(\mathcal{A}) \leq\ & \mathsf{Adv}^{\mathsf{MLWE}_{\mathrm{q},\mathrm{n},\mathrm{k}_{\mathrm{ext}},\mathrm{l}_{\mathrm{ext}},\mathrm{D}_{\mathrm{ext}}}}_{\mathcal{B}}(\lambda) + \frac{Q_{\mathsf{setup}}}{2^{\kappa}} \\
& + Q_s \cdot \mathsf{Adv}^{\mathsf{Ext}}_{\mathsf{CS},\mathcal{C}}(\lambda) \\
& + Q_s \cdot \mathsf{Adv}^{\mathsf{ZK}}_{\pi_2,\mathcal{D}}(\lambda) \\
& + Q_s \cdot \mathsf{Adv}^{\mathsf{SND}}_{\pi_1,\mathcal{E}}(\lambda) \\
& + Q_s \cdot \mathsf{Adv}^{\mathsf{CP}}_{\mathsf{HE},\mathcal{V}}(\lambda) \\
& + \mathsf{Adv}^{\text{EUF-CMA}}_{\mathsf{Raccoon},\mathcal{R}}(\lambda).
\end{aligned}
$$

Security Analysis

Theorem 5 (Blindness). *Let $\mathcal{A}$ be any* PPT *adversary that wins the blindness experiment instantiated with our blind signature in Figure 1 with advantage* $\mathsf{Adv}^{\mathsf{BLD}}_{\mathsf{BRaccoon},\mathcal{A}}(\lambda)$. *Then there exist PPT adversaries $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{K}$ and $\mathcal{S}$ such that, for the security parameter λ, we have*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{BLD}}_{\mathsf{BRaccoon},\mathcal{A}}(\lambda) \leq\ & \mathsf{Adv}^{\mathsf{ZK}}_{\pi_1,\mathcal{B}}(\lambda) + \mathsf{Adv}^{\mathsf{HID}}_{\mathsf{CS},\mathcal{C}}(\lambda) \\ & + \mathsf{Adv}^{\mathsf{KS}}_{\pi_2,\mathcal{K}}(\lambda) + \mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{HE},\mathcal{S}}(\lambda) + \epsilon_{\mathsf{Stat}}\end{aligned}$$

NIZKs

- We can use the LNP proof system for all lattice relations, which can be combined with LaBRADOR for compactness
- But we need some kind of post-quantum zk-SNARK to prove the evaluation of the hash function in move three
- We use the Poseidon zero-knowledge-friendly hash
- We estimate the number of constraints using Aurora

Scheme	Non-standard assumptions	vk (KB)	Comm. (KB)	sig (KB)	Sec. (bits)	#Moves	Q_s
Proof of knowledge of signature (Fischlin approach)							
[AKSY22]	OM-ISIS	0.8	1.4	45.2	109	2	2^{64}
[dPK22, HSS24]	–	≈ 56	≈ 1000	≈ 100	128	2	2^{64}
[LNP22a]	–	1331	16384	150	128	2	2^{18}
[BLNS23b, LSS24]	ISIS-f[5]	1	22	27	128	2	2^{64}
[JRS23, AGJ+24]	–	48	45	79.6	128	2	2^{32}
[BLNS23a]	–	1	63	22	128	2	2^{20}
[JS25a, JS25b]	–	36.0	53.2	36.3	125	3	2^{32}
Blinded identification scheme (ad-hoc construction)							
[HKLN20]	–	443.7	34037.3	7915.5	128	3	7
[KLR25]	–	≈ 122750	≈ 9000	≈ 350	128	4	2^{64}
Our scheme (regular signature)							
BRaccoon$_{\mathsf{LNP}}$	Raccoon	10	3110 3150 3260	28.00 31.78 43.10	128	4	2^{20} 2^{32} 2^{64}
BRaccoon$_{\mathsf{LNPLaB}}$	Raccoon	10	847	28.00 31.78 43.10	128	4	2^{20} 2^{32} 2^{64}

Thanks! Questions?