BlackLemon: Snake-eye Resistance with Short Secrets

rat4
https://blacknet.ninja

February 10, 2025

Abstract

Snake-eye resistance is a recently proposed property of public key encryption that arises in advanced protocols such as oblivious message retrieval. In a nutshell, it is connected to the already established notion of robustness. We discuss a generic approach to augment a suitable PKE with snake-eye resistance.

1 Introduction

How a lightweight cryptocurrency wallet could retain privacy? Private information retrieval, introduced in 1990s, answers a half of the question: if transaction identifiers are already known, PIR can be used. An answer to the other half was recently formalised as oblivious message detection[1]. It is a protocol that employs homomorphic encryption to encrypt a secret key and perform decryption over this homomorphism. Because the secret key and the result of decryption stay in encrypted form, the privacy is preserved. The interplay of two encryptions might be better shown step by step. Alice encrypts a transaction with Bob's public key and broadcasts it. Bob encrypts his secret key with homomorphic encryption and gives it to Carol. Carol tries to decrypt all transactions with Bob's key and replies with a compact digest. Bob decrypts the digest that contains his transactions.

Questions of practicality followed the theoretical framework. A line of research[2][3][4][5] (some others are left out of scope) had improved performance, however the most efficient schemes were shown to be vulnerable to a snake-eye attack.

2 Preliminary

2.1 Notation

Let δ a probability of successful attack, $\mathcal{A}$ an output of adversary. Probability distributions $\mathcal{U}, \chi_{key}, \chi_{err}$ are uniform, sparse ternary, discrete Gaussian. Denote $\mathbb{Z}/q\mathbb{Z}$ as $\mathbb{Z}_q$, and a power-of-two cyclotomic ring $\mathbb{Z}_q[X]/(X^n+1)$ as $\mathcal{R}_q$. If number of dimensions $\kappa \leq \ell \leq n$ equals n, reinterpret $\mathbb{Z}_q^n$ as $\mathcal{R}_q$. Short and long refer to the infinity norm. Shorthands sk, pk, ct, pt, m are for secret key, public key, ciphertext, plaintext, message.

2.2 Ring-LWE

We will use the non-dual form of Ring-LWE.[6]

Ring-LWE Sample is a pair of form: $(a, b = a * s + e) \in \mathcal{R}_q \times \mathcal{R}_q$ where $a \leftarrow \mathcal{U}, s \leftarrow \chi_{key}, e \leftarrow \chi_{err}$.

Decision Ring-LWE problem is to distinguish between Ring-LWE samples and uniformly random samples.

Search Ring-LWE problem is to recover s from Ring-LWE samples.

3 Snake-Eye Attack

In snake-eye attack[1], the goal of adversary is to output ciphertexts that are likely to decrypt to the same plaintext for independent and honestly generated secret keys. Formally,

$$\begin{aligned} &\Pr\big[sk_1 \leftarrow \text{keygen}(), sk_2 \leftarrow \text{keygen}(), \\ &\quad \forall ct \in \mathcal{A} : \text{decrypt}(sk_1, ct) = \text{decrypt}(sk_2, ct) \neq \perp \big] \leq \delta \end{aligned} \tag{1}$$

3.1 LWE

Typically, LWE decryption procedure looks like[7]:

$$\text{decode}(b - a * s) \tag{2}$$

Where s is a secret key, and ciphertext comprises a and b. Decoding is done via Babai round-off procedure.

By setting $a = 0$ the adversary can make the result of decryption independent of secret key. In [4] it was shown that exclusion of such trivial ciphertexts is insufficient in the general case, where short secrets can be preferred for efficiency.

3.2 NTRU

In modernised variants of NTRU[8] decryption procedure looks like:

$$\text{decode}(c * f)$$

Where f is a secret key, and ciphertext comprises c. Decoding is done via reduction modulo a small integer p.

The situation with snake-eyes is roughly similar to LWE. Whilst our idea is ought to be equally applicable to NTRU, it hasn't been formally checked.

4 Related Work

As originally noted in [1], it is possible to apply a zero-knowledge proof to the problem. In more detail, this method requires a proof that public key was properly generated, because $ct \approx \chi_{err}$ is a snake-eye. LaBRADOR[9] is among compactest proofs, yet the size is on the order of 50 KiB (even without

zero-knowledge property). This is a bit too much to show a cryptocurrency address as a text or barcode.

As discussed in [4], LWE with uniform secrets can be snake-eye resistant. Consequently, the authors proposed LWEmongrass that is a LWE-based encryption with hybrid secret distribution, wherein only a few secret elements are uniform for snake-eye resistance and mostly are short for efficiency. We note that in principle Ring-LWE with uniform secrets should be snake-eye resistant likewise unstructured LWE, albeit it would carry on the same efficiency issue of long decoding range (owing to the regularity lemma[7]).

5 BlackLemon

We start with observation that Ring-LWE doesn't seem to be directly compatible with hybrid secret distribution due to the (nega)cyclic structure (also see *Remark 5.5* in [4]). A next question could be whether Module-LWE is appropriate, since it partially destructures Ring-LWE.

Instead, we consider an indirect approach, and then apply it to sRLWE[3]. The cheapest operation over homomorphic encryption is addition. With this in mind, return to equation 2 and notice that summing a ciphertext and a uniformly random value could "repel snakes". To suit equation 1, it is sufficient to be sampled once along secret key. Intuitively, this should not interfere with basic security of encryption. Being a non-blackbox transformation, it requires examination on a case-by-case basis, and in fact we spotted a hurdle with one of advanced properties (see linkability).

5.1 Augmented sRLWE

- **SecretKeyGen** sample $a \leftarrow \chi_{key}, \vec{b} \leftarrow \mathcal{U}$ and return $sk = (a, \vec{b}) \in \mathcal{R}_q \times \mathbb{Z}_q^\kappa$
- **PublicKeyGen(sk)** sample $a \leftarrow \mathcal{U}, e \leftarrow \chi_{err}$ and compute

$$pk = (a, b = a * sk.a + e, \vec{c} = sk.\vec{b}) \in \mathcal{R}_q \times \mathcal{R}_q \times \mathbb{Z}_q^\kappa \tag{3}$$

- **Encrypt(pk, $\vec{\mathbf{m}}$)** sample $u \leftarrow \chi_{key}, e_1 \leftarrow \chi_{err}, e_2 \leftarrow \chi_{err}$,
  let $\left[\vec{pt}[i] = \left\lfloor \frac{q}{2}\vec{m}[i] \right\rceil\right]_{i=1}^{\ell} \in \mathbb{Z}_q^\ell$
  and compute

$$ct = (a = pk.a * u + e_1, \vec{b} = pk.b * u + pk.\vec{c} + \vec{pt} + e_2) \in \mathcal{R}_q \times \mathbb{Z}_q^\ell \tag{4}$$

- **Decrypt(sk, ct)** compute

$$\vec{d} = ct.\vec{b} - ct.a * sk.a - sk.\vec{b} \in \mathbb{Z}_q^\ell \tag{5}$$

check

$$\forall i \in [\ell] : \vec{d}[i] \in [-r; r] \vee \left\lfloor \frac{2}{q} - \vec{d}[i] \right\rceil \in [-r; r] \tag{6}$$

round off

$$\left[\vec{m}[i] = \left\lfloor \frac{2}{q} \vec{d}[i] \right\rceil \right]_{i=1}^{\ell} \in \mathbb{Z}_2^\ell \tag{7}$$

5.2 Properties

- **Correcntness and completeness** follow from sRLWE.
  By rearranging equation 4, we get:

$$ct = (a = \underbrace{pk.a * u + e_1}_{\text{sRLWE.ct.a}}, b = \underbrace{pk.b * u + pt + e_2}_{\text{sRLWE.ct.b}} + pk.c)$$

  That is an invertible operation over sRLWE ciphertext.

- **IND-CPA and IK-CPA** security relies on Ring-LWE assumption.
  Rearrange equation 4 to get:

$$ct = (a = \underbrace{pk.a * u + e_1}_{\text{Ring}-\text{LWE}}, b = \underbrace{pk.b * u + e_2}_{\text{Ring}-\text{LWE}} + pk.c + pt)$$

  Both, decision Ring-LWE and search Ring-LWE problems are required to be hard.

- **Snake-eye resistance** is provided.
  Examine equation 5:

$$d = \underbrace{ct.b - ct.a}_{\mathcal{A}} * sk.a - sk.b$$

  Recall that by definition we require a honestly generated $sk.b \leftarrow \mathcal{U}(\mathbb{Z}_q^\kappa)$, and $\mathbb{Z}_q^\kappa$ is a cyclic group. Counting in equations 6 and 7, for any adversary:

$$\delta = \left(\frac{2r+1}{q} \right)^\kappa$$

- **Regeneration of public keys** is linkable.

  To begin with, LWE public keys are regenerable, meaning that multiple public keys may correspond to a single secret key. These additional public keys are unlinkable to each other, even if correspond to the same secret.

  The "repellent value" added by us conflicts with unlinkability. It has to contain some entropy to provide snake-eye resistance, and the same entropy is a source of linkability.

  For use in practice, we can consider a compromise. Full node wallets essentially don't need snake-eye resistance and hence could fix the new value to zero to get the original encryption with regenerable keys.

6 Evaluation

We augmented SophOMR[5] proof of concept for evaluation. The free software is available online:

`https://github.com/blacknet-ninja/blacklemon-poc`

Detection time increased insignificantly at the cost of detection key size.

Table 1: Breakdown of detection key

Component	Switch	Rotation	Multiplication	sRLWE	BlackLemon	Total
Size	4 MiB	136 MiB	58 MiB	19 MiB	19 MiB	236 MiB

Table 2: Comparison of public-key encryption

Scheme	sRLWE[3]	sRLWE + BlackLemon	LWEmongrass[4]	sRLWE + LaBRADOR[9]
Snake-eye resistance	x	✓	✓	✓
Regenerable pk	✓	x	✓	✓
Ciphertext	2.01 KiB	2.01 KiB	1.84 KiB	2.01 KiB
Compressed pk	2.04 KiB	2.04 KiB	4.49 KiB	$\approx$ 52 KiB
Public key	4.01 KiB	4.01 KiB	1.37 MiB	$\approx$ 54 KiB
Secret key	2.01 KiB	2.01 KiB	5.49 KiB	2.01 KiB

References

[1] Zeyu Liu and Eran Tromer. Oblivious Message Retrieval, 2021.

[2] Zeyu Liu, Eran Tromer, and Yunhao Wang. Group Oblivious Message Retrieval, 2023.

[3] Zeyu Liu, Eran Tromer, and Yunhao Wang. PerfOMR: Oblivious Message Retrieval with Reduced Communication and Computation, 2024.

[4] Zeyu Liu, Katerina Sotiraki, Eran Tromer, and Yunhao Wang. Snake-eye Resistance from LWE for Oblivious Message Retrieval and Robust Encryption, 2024.

[5] Keewoo Lee and Yongdong Yeo. SophOMR: Improved Oblivious Message Retrieval from SIMD-Aware Homomorphic Compression, 2024.

[6] Chris Peikert. How (Not) to Instantiate Ring-LWE, 2016.

[7] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A Toolkit for Ring-LWE Cryptography, 2013.

[8] Jeffrey Hoffstein, Jill Pipher, and Joseph Silverman. NTRU: A Ring-Based Public Key Cryptosystem, 1998.

[9] Ward Beullens and Gregor Seiler. LaBRADOR: Compact Proofs for R1CS from Module-SIS, 2022.