# Neural Leakage–based Cryptanalysis of LowMC with Linear Complexity

Kwangjo Kim

IACR Fellow, Emeritus Professor of KAIST,
President of International Research Institute for Cyber Security(IRCS), Korea
kkj@kaist.ac.kr

**Abstract.** MPC-in-the-Head protocols enable post-quantum digital signatures based solely on symmetric primitives, with PICNIC being a prominent example built on the LowMC block cipher. While existing analyses assume exact Boolean circuit semantics, recent advances in neural representations suggest that piecewise-linear implementations may introduce activation-boundary leakage. In this work, we investigate whether such leakage can be exploited in the context of LowMC and MPC-in-the-Head transcripts.
We propose a perturbation-based probing methodology that models neural leakage and reduces round-key recovery to independent binary hypothesis tests via majority voting. Exploiting the linear structure of the LowMC key schedule, we demonstrate that recovery of the first-round key enables efficient reconstruction of the master key with linear complexity. Experimental results confirm successful recovery of 128-, 192-, and 256-bit keys under the proposed model, highlighting a new dimension in symmetric cryptanalysis and the need to consider learning-based leakage in future designs.

**Keywords:** Perturbation-based Probing · PICNIC/LowMC · Linear Complexity · Key Recovery

## 1 Introduction

Post-quantum cryptography has motivated the exploration of diverse signature constructions beyond classical number-theoretic assumptions. Among these approaches, the MPC-in-the-Head (MPCitH) paradigm has emerged as a powerful tool for constructing zero-knowledge proofs and digital signatures from symmetric primitives. PICNIC is a post-quantum digital signature scheme based on zero-knowledge proofs MPCithH paradigm using symmetric primitives (hash + block cipher) rather than number-theoretic assumptions. LowMC [1] is the first FHE/MPC-friendly block cipher, and one of its applications is to the PICNIC [4, 3] signature scheme.

The PICNIC signature scheme is a prominent example of this approach. PICNIC proves knowledge of a secret key $k$ satisfying $C = \mathrm{LowMC}_k(P)$ using an MPCitH protocol combined with the Fiat–Shamir transformation.

Recent work on neural representations of cryptographic primitives suggests that Boolean circuits can be represented as piecewise-linear neural networks. Such representations introduce geometric structures that may exhibit activation-boundary leakage. To the first time of my knowledge, this paper explores whether such leakage could affect MPCitH transcripts.

A number of algebraic attacks on LowMC have been proposed under the assumption that an adversary is given only a single plaintext-ciphertext pair: mainly based on two algebraic techniques: linearization [2] by guessing, and the polynomial method [5].

However, Gérault et al. [6] proposed methodologies for implementing block ciphers such as AES only in a manner that is resilient against potential deep learning–based cryptanalysis, highlighting the emerging threat of neural network assisted attacks. In parallel, Kim [9] investigated the structural vulnerabilities of neural implementations of symmetric primitives, demonstrating that piecewise-linear representations may introduce exploitable leakage through activation boundaries. Motivated by these developments, this paper extends the neural leakage–based cryptanalytic framework beyond classical block ciphers and, for the first time, applies it to the LowMC family.

The remainder of this paper is organized as follows. Section 2 introduces the necessary preliminaries, including PICNIC, LowMC, and the Grain-based generator. Section 3 presents our neural leakage–based key recovery methodology and experimental results. Section 4 concludes the paper and discusses security implications.

## 2 Preliminaries

### 2.1 PICNIC Signature

PICNIC constructs signatures using a zero-knowledge proof of knowledge for the relation

$$C = \text{LowMC}_k(P)$$

for a given security parameter $k$: $k \leftarrow \{0,1\}^\lambda$ The public key is $(P, C)$ and the secret key is $k$.

The MPC-in-the-Head paradigm was introduced by Ishai et al [8]. It simulates a multi-party computation protocol internally.

- Secret-share witness, $w$
- Simulate MPC evaluation
- Commit to party views
- Reveal subset of views

Verification checks consistency of the opened views.

### 2.2 LowMC

Let $k \in \mathbb{F}_2^n$ denote the $n$-bit secret key. In LowMC, the round keys are derived via a sequence of public binary matrices generated deterministically (via the Grain SSG).

**Round-Key Derivation:** For each round $i = 0, 1, \ldots, r$, a public matrix

$$K_i \in \mathbb{F}_2^{n \times n}$$

is defined. The corresponding round key $rk_i \in \mathbb{F}_2^n$ is computed as

$$rk_i = K_i \cdot k,$$

where multiplication is over $\mathbb{F}_2$.

**Initial Whitening:** Before the first round, the plaintext $x \in \mathbb{F}_2^n$ is whitened using

$$x^{(0)} = x \oplus rk_0.$$

**Round Function with Whitening:** For each round $i = 1, \ldots, r$, the state is updated as

$$x^{(i)} = L_i\big(S(x^{(i-1)})\big) \oplus rk_i \oplus c_i,$$

where:

- $S(\cdot)$ is the non-linear layer (parallel 3-bit S-boxes),
- $L_i \in \mathbb{F}_2^{n \times n}$ is the linear layer matrix,
- $c_i \in \mathbb{F}_2^n$ is the round constant.

**Ciphertext.** The final ciphertext is

$$y = x^{(r)}.$$

**Remarks**

- The key schedule is *fully linear* over $\mathbb{F}_2$.
- All matrices $K_i$ are public and generated via the Grain-based SSG.
- There is no non-linear key expansion; security relies on the LowMC round structure rather than key schedule complexity.
- The whitening step ensures that the key influences both input and all intermediate rounds.

### 2.3 Grain Self-Shrinking Generator

PICNIC uses a Grain-based Self-Shrinking Generator (SSG) [7] combined with a self-shrinking rule to deterministically generate the public parameters of LowMC, including linear matrices, round-key matrices, and round constants.

**Underlying LFSR (Grain):** The generator is based on an 80-bit linear feedback shift register (LFSR) with state

$$(s_0, s_1, \ldots, s_{79}) \in \{0,1\}^{80}.$$

At each clock, the LFSR is updated as

$$s_{t+80} = s_{t+62} \oplus s_{t+51} \oplus s_{t+38} \oplus s_{t+23} \oplus s_{t+13} \oplus s_t.$$

Equivalently, the feedback polynomial over $\mathbb{F}_2$ is

$$p(x) = x^{80} + x^{62} + x^{51} + x^{38} + x^{23} + x^{13} + 1.$$

The output sequence $\{a_t\}_{t \geq 0}$ is taken from the LFSR (typically the last state bit at each clock).

**Self-Shrinking Rule:** The sequence $\{a_t\}$ is processed in pairs:

$$(a_{2i}, a_{2i+1}), \quad i = 0, 1, 2, \ldots$$

and produces output bits $\{z_j\}$ as follows:

$$z_j = \begin{cases} a_{2i+1}, & \text{if } a_{2i} = 1, \\ (\text{no output}), & \text{if } a_{2i} = 0. \end{cases}$$

Thus, the first bit of each pair acts as a control bit that determines whether the second bit is emitted.

**Usage of LowMC in PICNIC:** The resulting bit stream $\{z_j\}$ is consumed sequentially to fill:

- linear layer matrices,
- round-key matrices,
- round constants.

This provides a compact, deterministic, and pseudorandom method to define all public parameters of the LowMC block cipher used in PICNIC.

## 3 Our Method

We attempt to recover the PICNIC master key not by targeting the full algorithmic execution, but by focusing solely on the key expansion component. Specifically, we first recover the first round key, $rk_1$, and then invert the key expansion process to derive the original master key.

### 3.1 Recovering $rk_1$ at first

The attack reduces round-key recovery to independent binary hypothesis tests per coordinate, implemented as majority-vote classifiers over noisy threshold leakage, followed by linear inversion of the LowMC key schedule. This is not just heuristic, but binary hypothesis testing:

$$\begin{aligned} \text{Hypothesis } H_1: &\quad rk_1[i] = 1 \;\Rightarrow\; \mathbb{E}[y] = +\mu, \\ \text{Hypothesis } H_0: &\quad rk_1[i] = 0 \;\Rightarrow\; \mathbb{E}[y] = -\mu. \end{aligned}$$

Majority voting is approximately equals to sign test/ Neyman-Pearson detector.

$$\hat{b}_i = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} y_t \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Neural interpretation from this research angle each bit recovery equivalent to 1-neuron perception, ReLU boundary detection or threshold classifier.

The high-level operation tree is as below:

```
(1) Generate K1 (invertible GF(2))
    |
(2) Generate master_key
    |
(3) rk1 = K1 · master_key
    |
(4) Noisy Oracle Construction
    |
(5) [Diagnostics]
    |-- pairgenerate_sym()
    |-- pairgenerate_change()
(6) [Core Attack]
    |--recover_rk1()
           |-- majority vote per bit
    |
(7) [Linear Solve]
    master_hat = K1^{-1} · rk1_hat
    |
(8) Evaluation (HD, match, confidence)
```

The pseudocode of LowMC full key recovery with noisy oracle is described in **Algorithm** 1.

The recovering $rk_1$, `RecoverRK1()` is depicted in **Algorithm** 2:

**Algorithm 1** LowMC full key recovery with noisy oracle

Initialize parameters $N, \mu$, trials $T$
$K_1 \leftarrow$ random invertible matrix in $\mathbb{F}_2^{N \times N}$
$K_1^{-1} \leftarrow \text{GF2Inverse}(K_1)$
$k \leftarrow \text{RandBits}(N)$
$rk_1 \leftarrow K_1 \cdot k$
Construct oracle $\mathcal{O}$:

$$\mathcal{O}(i) = \begin{cases} 1 & \text{if } (\pm\mu + noise + probe) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

**for** $i = 0$ to $7$ **do**
    $(s_1, d_1) \leftarrow \text{PairSym}(\mathcal{O}, i)$
    or $(s_2, d_2) \leftarrow \text{PairChange}(\mathcal{O}, i)$
**end for**
$\hat{rk}_1 \leftarrow \text{RecoverRK1}(\mathcal{O}, T)$
$\hat{k} \leftarrow K_1^{-1} \cdot \hat{rk}_1$
$d_1 \leftarrow \text{HD}(rk_1, \hat{rk}_1)$
$d_2 \leftarrow \text{HD}(k, \hat{k})$
**return** $(\hat{k}, \hat{rk}_1, d_1, d_2)$

**Algorithm 2** Recovering round key $rk_1$

**Require:** Oracle $\mathcal{O}$, trials $T$
**Ensure:** $\hat{rk}_1$
**for** $i = 0$ to $N - 1$ **do**
    $(\hat{b}_i, c_i) \leftarrow \text{EstimateBit}(\mathcal{O}, i, T)$
    $\hat{rk}_1[i] \leftarrow \hat{b}_i$
**end for**
**return** $\hat{rk}_1$

**Algorithm 3** Estimating one round-key bit via majority voting

**Require:** Oracle $\mathcal{O}$, index $i$, trials $T$
**Ensure:** Estimated bit $\hat{b}_i$
$ones \leftarrow 0$
**for** $t = 1$ to $T$ **do**
    $y \leftarrow \mathcal{O}(i)$
    $ones \leftarrow ones + y$
**end for**
**if** $2 \cdot ones \geq T$ **then**
    $\hat{b}_i \leftarrow 1$
**else**
    $\hat{b}_i \leftarrow 0$
**end if**
confidence:

$$c_i = \left| \frac{ones}{T} - \frac{1}{2} \right|$$

**return** $(\hat{b}_i, c_i)$

The estimating one round-key bit via majority voting `EstimateBit()` is given in **Algorithm** 3:

The function `PairSym()` used in Step 8 of **Algorithm** 1 is used as **Algorithm** 4. You can select `PairChange()` depending on your variety, but skipped due to space limitation.

**Algorithm 4** PairSym()

**Require:** Oracle $\mathcal{O}$, index $i$, amplitude $a$
**for** $t = 1$ to $T$ **do**
  $y_1 \leftarrow \mathcal{O}(i, +a)$
  $y_2 \leftarrow \mathcal{O}(i, -a)$
  **if** $y_1 = y_2$ **then**
    $same \leftarrow same + 1$
  **else**
    $diff \leftarrow diff + 1$
  **end if**
**end for**
**return** $(same, diff)$

### 3.2 Recovering master key from $rk_1$

This implementation is at first to generate $rk_1$ from the external master key multiplied with random invertible matrix for key whitening stage and to recover embedded round key 0 by statistical estimation via majority voting i.e., Neyman-Pearson detector.

Even with perfect recovery of the first-round key via neural leakage, recovery of the master key from step 12 of **Algorithm** 1 depends critically on the rank of the linear key schedule. In rank-deficient cases, the adversary obtains a solution space rather than a unique key, highlighting a structural dependency between leakage exploitation and key schedule invertibility. After using Grain-SSG with full rank, we succeeded to recover the master key of LowMC-128, 192 and 256 as below with 100 % from recovered $rk_1$. This is our first achievement to recover any master key of LowMC (*i.e.,* key schedule of PICNIC) in linear complexity.

## 4 Conclusion

In this paper, we investigated the implications of neural leakage arising from piecewise-linear representations of Boolean circuits, and applied this perspective to the LowMC block cipher used in MPCitH protocols such as PICNIC and AIMer. By modeling activation-boundary effects through perturbation-based probing, we demonstrated that round-key information can be recovered via simple statistical hypothesis testing. Combined with the linear structure of the LowMC key schedule, this enables efficient reconstruction of the master key under suitable conditions with linear complexity.

Our results highlight a new dimension in symmetric cryptanalysis, where neural or hybrid implementations may introduce unintended leakage channels not captured by classical Boolean security models. This suggests that future designs of many MPC-friendly primitives and post-quantum signature schemes should explicitly consider robustness against learning-based and activation boundary attacks. Further work is needed to formalize such leakage models and to develop countermeasures ensuring security in neural or approximate computing environments.

## References


1. Albrecht, M., et al.: Lowmc: A low multiplicative complexity block cipher. In: ASIACRYPT (2016) 1
2. Banik, S., Barooti, K., Caforio, A., Vaudenay, S.: Memory-efficient single data-complexity attacks on LowMC using partial sets. Cryptology ePrint Archive, Report 2022/688 (2022), https://eprint.iacr.org/2022/688 1
3. Chase, M., Derler, D., Goldfeder, S., Orlandi, C.: Post-quantum zero-knowledge and signatures from symmetric-key primitives. IACR ePrint 2017/279 (2017) 1
4. Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., Slamanig, D., Zaverucha, G.: The picnic signature scheme. https://microsoft.github.io/Picnic/ (2020), design document v2.2 1
5. Dinur, I.: Cryptanalytic applications of the polynomial method for solving multivariate equation systems over GF(2). In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021. Lecture Notes in Computer Science, vol. 12696, pp. 374–403. Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_13 1
6. Gérault, D., Hambitzer, A., Ronen, E., Shamir, A.: How to securely implement cryptography in deep neural networks. Report 288, IACR Cryptology ePrint Archive (2025), https://eprint.iacr.org/2025/288.pdf 1
7. Hell, M., Johansson, T., Maximov, A., Meier, W.: Grain: A stream cipher for constrained environments. In: International Journal of Wireless and Mobile Computing (2007) 2
8. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: STOC (2007) 2
9. Kim, K.: Assessing geometric security of aes neural realizations: Linear-time key recovery via neural leakage (2026), https://eprint.iacr.org/2026/734 1