Attacking ECDSA with Nonce Leakage by Lattice Sieving: Bridging the Gap with Fourier Analysis-based Attacks

Yiming Gao[1,⋆], Jinghui Wang[1,⋆], Honggang Hu[1,2(✉)], and Binang He[1]

[1] School of Cyber Science and Technology
University of Science and Technology of China, Hefei 230027, China
{qw1234567,liqing21,hebinang}@mail.ustc.edu.cn
[2] Hefei National Laboratory, Hefei 230088, China
hghu2005@ustc.edu.cn

Abstract. The Hidden Number Problem (HNP) has found extensive applications in side-channel attacks against cryptographic schemes, such as ECDSA and Diffie-Hellman. There are two primary algorithmic approaches to solving the HNP: lattice-based attacks and Fourier analysis-based attacks. Lattice-based attacks exhibit better efficiency and require fewer samples when sufficiently long substrings of the nonces are known. However, they face significant challenges when only a small fraction of the nonce is leaked, such as 1-bit leakage, and their performance degrades in the presence of errors.

In this paper, we address an open question by introducing an algorithmic tradeoff that significantly bridges the gap between these two approaches. By introducing a parameter x to modify Albrecht and Heninger's lattice, the lattice dimension is reduced by approximately $(\log_2 x)/l$, where l represents the number of leaked bits. We present a series of new methods, including the interval reduction algorithm, several predicates, and the pre-screening technique. Furthermore, we extend our algorithms to solve the HNP with erroneous input. Our attack outperforms existing state-of-the-art lattice-based attacks against ECDSA. We obtain several records including 1-bit and less than 1-bit leakage on a 160-bit curve, while the best previous lattice-based attack for 1-bit leakage was conducted only on a 112-bit curve.

Keywords: ECDSA · Hidden Number Problem · Lattice Sieving · Lattice-based Attacks

1 Introduction

The Hidden Number Problem (HNP) was originally proposed by Boneh and Venkatesan as a number theoretic problem to investigate the bit security of the Diffie-Hellman key exchange scheme [12]. Their work was subsequently extended by Nguyen and Shparlinski to analyze the security of ECDSA with partial known

⋆ Yiming Gao and Jinghui Wang are the co-first authors of this work.

nonce leakage [29]. In the scenario that an attacker can obtain some information about the nonce used in each signature generation of ECDSA, it is possible to recover the secret key by solving the corresponding HNP instance. Currently, there are two types of attacks for solving the HNP: Fourier analysis-based attacks and lattice-based attacks.

The foundational principles of Fourier analysis-based attacks were initially introduced by Bleichenbacher [10], providing the basis for subsequent research [5,6,14,35]. These attacks have been considered to be more tractable to break HNP instances with limited known bits and even with errors. However, they demand a substantial number of samples and exhibit a high computational overhead [5,6,11]. The latest advance is obtained by Aranha et al., who successfully broke 192-bit ECDSA with less than 1-bit leakage [6].

In lattice-based attacks, the HNP is transformed into the Bounded Distance Decoding (BDD) Problem, which is a variant of the Closest Vector Problem (CVP). Using Kannan's embedding [23], it can be further transformed into the Unique Shortest Vector Problem (uSVP). The success of lattice-based attacks highly depends on whether the target vector corresponding to the secret is sufficiently short in the lattice. It is believed that the lattice-based attacks would become ineffective when only a small fraction of the nonce is revealed, particularly in the case of 1-bit leakage. Aranha et al. emphasized that exploiting a 1-bit nonce leakage to attack ECDSA is infeasible due to the underlying structure of the HNP lattices [5]. Additionally, lattice-based attacks have been considered to behave very poorly with noisy data, which poses constraints on practical side-channel attacks.

At EUROCRYPT 2021, Albrecht and Heninger extended the applicability of lattice-based attacks with their Sieving with Predicate (Sieve-Pred) algorithm [3] and the state-of-the-art lattice sieving library G6K [2]. The Sieve-Pred algorithm no longer treats sieving algorithms as black boxes for SVP. Instead, it uses a predicate to check all the vectors in the database output by sieving algorithms. The predicate they used involves scalar multiplication on the elliptic curve, which is a nonlinear operation and results in significant overhead. The nonlinear predicate was improved to a linear predicate by Xu et al. [38].

Lattice-based attacks and Fourier analysis-based attacks have their unique characteristics. Current lattice-based attacks are known for their minimal sample requirements and efficient processing. However, they are considered to be infeasible when dealing with challenging HNP instances. In contrast, Fourier analysis-based attacks can handle more difficult instances such as 1-bit leakage on a 192-bit curve [6], but demand a significantly larger sample size and computational time. This raises open questions [20]: *Can lattice-based attacks be enhanced by utilizing more samples? Is there a smooth tradeoff that can be characterized between these two types of algorithms?*

1.1 Contributions

In this work, we enhance the lattice-based attacks by utilizing more samples, significantly bridging the gap between lattice-based attacks and Fourier analysis-based attacks. The main idea of our attack is using more samples to obtain the specific ones employed in the lattice construction. This allows us to increase the lattice determinant while keeping the norm of the target vector roughly unchanged. Consequently, it results in a reduction of lattice dimension while still satisfying the enabling condition for the attack.

In practice, we successfully address the case of 1-bit leakage on a 160-bit curve, surpassing the best previous lattice-based attack for 1-bit leakage, which was only conducted on a 112-bit curve [38]. Moreover, despite the belief that lattice-based attacks are ineffective for erroneous input [6], we demonstrate the effectiveness of our new attack in handling erroneous input, and successfully break the 160-bit ECDSA with less than 1-bit leakage. Our main contributions are detailed as follows.

Improved Algorithms for Solving the HNP. Firstly, we propose a new lattice construction that introduces a parameter x to trade off the lattice dimension. Our modification is based on Albrecht and Heninger's lattice [3], where the hidden number α is transformed into k_0' using the elimination method and the recentering technique.

In our construction, we employ a decomposition technique in which k_0' is decomposed as $x \cdot \alpha_0 + \alpha_1$, with the condition that $|\alpha_1| \leq x/2$. The target lattice vector contains the information of α_0, which is expected to be included in the database output by sieving algorithms.

Compared to Albrecht and Heninger's lattice [3], our construction can offer a dimension reduction of approximately $(\log_2 x)/l$, where l represents the number of leaked bits. The reduction in lattice dimension leads to a significant efficiency advantage because sieving algorithms have exponential complexity. As a tradeoff, our algorithm needs a larger number of samples to select the specific ones used in the lattice construction. Moreover, we prove the existence of a constant $c > 0$ which serves as a lower bound for the success probability of our algorithm. This is a new theoretical finding not reported in the literature.

Secondly, in order to determine the unique hidden number, we propose an improved linear predicate that makes use of the linear constraints from $2\log_2 q$ HNP samples. Our predicate demonstrates superior efficiency compared to the non-linear predicate proposed by Albrecht and Heninger [3], which involves time-consuming scalar multiplication over elliptic curves. Our predicate also outperforms the linear predicate used by Xu et al. [38]. Their predicate requires knowledge of all elements of the candidate vector, whereas ours only requires the last two elements. Furthermore, we identify an issue with Xu et al.'s predicate that may cause it to return true for some incorrect candidates. We provide a corrected version of their predicate for a fair comparison and demonstrate that our approach achieves superior performance while maintaining the desired functionality.

Thirdly, a predicate for the decomposition technique is proposed. We design an interval reduction algorithm with expected time complexity $O(\log^2 x)$ to recover the remaining part α_1, instead of an exhaustive search over the range $[-x/2, x/2]$. Moreover, we also present a pre-screening technique to pre-select candidates. This technique can effectively eliminate most incorrect candidates before checking the predicate.

Modified Algorithms for Handling Errors. We define HNP with erroneous input to handle practical scenarios in side-channel attacks where errors may exist in the leaked nonce. We demonstrate the effectiveness of our lattice construction for solving this problem, and provide the estimation for the minimum lattice dimension. In this case, our construction offers a greater reduction in lattice dimension of more than $\log_2 x/l$. Furthermore, our new algorithms and techniques for solving the HNP are extended to address the case of erroneous input.

New Records of Lattice-based Attacks against ECDSA. We carry out experiments on lattice-based attacks against ECDSA with nonce leakage. Our attack demonstrates a significant efficiency advantage over previous works [3,34,38]. The most notable achievement is the successful key recovery for the ECDSA instance with 1-bit leakage on a 160-bit curve, which is considered to be extremely difficult by previous lattice-based approaches. Moreover, we also successfully conduct attacks for the case of less than 1-bit leakage on various elliptic curves, including a 160-bit curve. Our source code is publicly available on GitHub[1].

1.2 Comparison with Related Work

In Table 1, our work is compared with previous records of lattice-based attacks. Several new records are listed, including 1-bit and less than 1-bit leakage on a 160-bit curve. To the best of our knowledge, we carry out the first lattice-based attack against ECDSA with less than 1-bit leakage (the leaked 1-bit of the nonce is exact with a probability of less than 1). For the case of 4-bit leakage, we also make significant progress. While the previous record for 4-bit leakage was achieved on a 384-bit curve [3,34], we extend the attack to a 512-bit curve.

Table 1: Lattice-based attacks against ECDSA with nonce leakage

	Nonce leakage				
Modulus	4-bit	3-bit	2-bit	1-bit	<1-bit
112-bit	-	-	-	[38], Ours (faster)	Ours
128-bit	-	-	-	Ours	Ours
160-bit	-	[29]	[3,34]	Ours	Ours
256-bit	[3]	[3,34]	[38], Ours (faster)	-	-
384-bit	[3,34]	[38], Ours (faster)	-	-	-
512-bit	Ours	-	-	-	-

Among lattice-based attacks, the currently best one is the Sieve-Pred algorithm in [3]. The efficiency of the predicate is essential, as it is used to check the

[1] https://github.com/JinghuiWW/ecdsa-leakage-attack

vectors in the database generated by sieving algorithms. Our linear predicate outperforms both predicates proposed in [3] and [38]. Our attack demonstrates superior efficiency when targeting various ECDSA instances. For example, we break the 112-bit ECDSA with 1-bit leakage in 6 minutes which is approximately 43 times faster than the currently fastest attack in [38].

Our work can also be viewed as a comprehensive extension of the work conducted by Sun et al. [34], who recognized the connection between Fourier-based attacks and lattice-based attacks, and proposed a general framework to enhance lattice attacks with more samples. However, they did not consider reducing the lattice dimension, nor did they utilize sieving algorithms to handle more challenging cases, such as 256-bit ECDSA with 2-bit leakage or 1-bit leakage case.

Compared with their work, our algorithm requires significantly fewer samples and achieves higher success rates. For instance, when targeting 160-bit ECDSA with 2-bit leakage, their algorithm required approximately 2^{27} samples and achieved a success rate of 15%, while our algorithm only needs 411 samples with a success rate of approximately 100%. Moreover, according to their estimation, the time complexity of their algorithm is 2^{110} BKZ-30 operations for 160-bit ECDSA with 1-bit leakage, which is impractical. However, we break this instance in only 13.7 hours using approximately 2^{25} ECDSA samples.

Our work shares some similarities with Fourier analysis-based attacks. However, except for very difficult cases, our attack works more efficiently with far fewer samples. Another advantage of our approach is the capability to recover the entire secret key all at once, while Fourier analysis-based attacks can only recover a few bits of the secret key in a single execution.

2 Preliminaries

Let $\mathbb{N}^+$ be the set $\{1, 2, 3, \cdots\}$. The logarithm with base two is denoted as $\log(\cdot)$, and the Euclidean norm is denoted as $\|\cdot\|$. For any integer z, the unique integer x satisfying $0 \le x < q$ and $x \equiv z \mod q$ is denoted by $|z|_q$. A function $f(k)$ is called $\mathrm{negl}(k)$ if for every positive polynomial function $\mathrm{P}(k)$, there exists an integer $N_\mathrm{P} > 0$ such that for all $k > N_\mathrm{P}$, $|f(k)| < 1/\mathrm{P}(k)$.

2.1 Lattices and Hard Problems

Given a matrix $\boldsymbol{B} = (\boldsymbol{b}_0, \dots, \boldsymbol{b}_{d-1})^T \subset \mathbb{R}^{d\times d}$ with linearly independent rows, the lattice generated by the basis $\boldsymbol{B}$ is defined as $\mathcal{L}(\boldsymbol{B}) := \{\sum_i^d x_i \boldsymbol{b}_i : x_i \in \mathbb{Z}\}$. We define π_i as the projections orthogonal to the span of $\boldsymbol{b}_0, \dots, \boldsymbol{b}_{i-1}$, and the Gram-Schmidt orthogonalisation as $\boldsymbol{B}^* = (\boldsymbol{b}_0^*, \dots, \boldsymbol{b}_{d-1}^*)$, where $\boldsymbol{b}_i^* := \pi_i(\boldsymbol{b}_i)$. For any $0 \le l < r \le d$, the projected sublattice $\mathcal{L}_{[l:r]}$ is defined as the lattice with basis $\boldsymbol{B}_{[l:r]} := (\pi_l(\boldsymbol{b}_l), \dots, \pi_l(\boldsymbol{b}_{r-1}))$.

Let $\lambda_i(\mathcal{L})$ be the radius of the smallest ball centred at the origin containing at least i linearly independent lattice vectors. Then $\lambda_1(\mathcal{L})$ is the Euclidean norm of the shortest non-zero vector in lattice $\mathcal{L}$.

The Gaussian heuristic predicts the number of lattice points inside a measurable body $\mathcal{B} \subset \mathbb{R}^n$, and it tells us that the number $|\mathcal{L} \cap \mathcal{B}|$ of lattice points inside $\mathcal{B}$ is approximately equal to $\mathrm{Vol}(\mathcal{B})/\mathrm{Vol}(\mathcal{L})$. Applying it to the Euclidean d-ball, the prediction of $\lambda_1(\mathcal{L})$ can be obtained.

Definition 1 (Gaussian Heuristic (GH)). *We denote by* $\mathrm{GH}(\mathcal{L})$ *the expected first minimum of a lattice* $\mathcal{L}$. *For a full rank lattice* $\mathcal{L} \subset \mathbb{R}^d$, *it is given by*

$$\mathrm{GH}(\mathcal{L}) = \frac{\Gamma\left(1+\frac{d}{2}\right)^{1/d}}{\sqrt{\pi}} \cdot \mathrm{Vol}(\mathcal{L})^{1/d} \approx \sqrt{\frac{d}{2\pi e}} \cdot \mathrm{Vol}(\mathcal{L})^{1/d}.$$

The final step above is obtained from Stirling's formula, and we utilize this asymptotic estimation in our theoretical analysis. In practical attacks and calculations, we directly compute the value of the Gamma function.

Albrecht and Heninger formalized two lattice problems augmented with a predicate [3], which are defined as follows:

Definition 2 (α-Bounded Distance Decoding with Predicate($\mathrm{BDD}_{\alpha,\mathrm{f}(\cdot)}$)). *Given a lattice basis* $\boldsymbol{B}$, *a vector* $\boldsymbol{t}$ *and a parameter* $\alpha > 0$ *such that the Euclidean distance* $\mathrm{dist}(\boldsymbol{t}, \boldsymbol{B}) < \alpha \cdot \lambda_1(\mathcal{L}(\boldsymbol{B}))$, *find the lattice vector* $v \in \mathcal{L}(\boldsymbol{B})$ *satisfying* $f(\boldsymbol{v} - \boldsymbol{t}) = 1$ *that is closest to* $\boldsymbol{t}$.

Definition 3 (unique Shortest Vector Problem with Predicate ($\mathrm{uSVP}_{\mathrm{f}(\cdot)}$)). *Given a lattice basis* $\boldsymbol{B}$ *and a predicate* $f(\cdot)$, *find the shortest non-zero vector* $\boldsymbol{v} \in \mathcal{L}(\boldsymbol{B})$ *that satisfies* $f(\boldsymbol{v}) = 1$.

$\mathrm{BDD}_{\alpha,\mathrm{f}(\cdot)}$ can be solved using a $\mathrm{uSVP}_{\mathrm{f}(\cdot)}$ oracle due to Kannan embedding technique, by constructing the lattice

$$\mathbf{C} = \begin{bmatrix} \mathbf{B} & 0 \\ \mathbf{t} & \tau \end{bmatrix},$$

where τ is the embedding factor. If $\boldsymbol{v}$ is the closest vector to $\mathbf{t}$, then the short vector $(\mathbf{t} - \boldsymbol{v}, \tau)$ is contained in the lattice $\mathcal{L}(\mathbf{C})$.

2.2 Lattice Algorithms

BKZ. The Block Korkine-Zolotarey (BKZ) algorithm, which can be regarded as an extended version of the LLL algorithm [26], was first introduced by Schnorr and Euchner [33]. It uses an oracle that solves the SVP in the lattice with a block size of β, spanned by $\boldsymbol{B}_{[0,\beta-1]}$. The short vector is then recursively inserted into the lattice basis. A BKZ tour is initiated by calling an SVP-solver on consecutive blocks $\boldsymbol{B}_{[i,\min(i+\beta,d-1)]}$ for $i = 0, \ldots, d-2$. The algorithm will proceed with these tours until there are no further changes observed within a single tour. We abbreviate BKZ with a block size of β as BKZ-β.

Lattice Sieving. Sieving algorithms are not only asymptotically superior to enumeration techniques [23,17,33,19], but also showing better practical performance in higher lattice dimensions, due to the recent progress in both theory [24,9,8,21] and practice [18,15,25,2,16].

The first sieving algorithm was proposed by Ajtai et al. [1] in 2001. It starts with a list of lattice vectors $L \subset \mathcal{L}$ and searches for shorter sums and differences of these vectors. The shorter combinations then replace the original longer vectors in the database. This process iterates until the database contains a significant number of short vectors, with the expectation of eventually finding the shortest vector.

Lattice sieving algorithms can be categorized into provably sieving algorithms and heuristic sieving algorithms. Heuristic sieving algorithms fall in the practical regime because of lower time and memory complexities. They are analyzed under the heuristic that the points in L are independently and uniformly distributed in a thin spherical shell. Nguyen and Vidick [30] proposed the first practical sieving algorithm, utilizing a database of $(4/3)^{d/2+o(d)} = 2^{0.2075d+o(d)}$ vectors and running in time $2^{0.415d+o(d)}$. The time complexity was subsequently improved to $2^{0.292d+o(d)}$ through nearest neighbor search techniques [8]. Various sieving algorithms have been efficiently implemented in G6K [2] and its GPU-accelerated version, G6K-GPU [16].

2.3 Hidden Number Problem

In the Hidden Number Problem (HNP) [12], we have an n-bit sized public modulus q, and there is a secret integer $\alpha \in \mathbb{Z}_q$, referred to as the hidden number. For $i = 0, 1, \ldots, m-1$, t_i are uniformly random integers in $\mathbb{Z}_q$, and we are provided with the corresponding value a_i such that $|t_i \cdot \alpha - a_i|_q = k_i < q/2^l$. The problem is to recover the hidden number α when m samples (t_i, a_i) are given. We denote the above problem as $\mathrm{HNP}(n, l)$.

2.4 Breaking ECDSA with Nonce Leakage

ECDSA. The global parameters for an ECDSA signature include an elliptic curve $E(\mathbb{F}_p)$ and a generator point G on $E(\mathbb{F}_p)$ of a prime order q. The secret signing key is an integer $0 \leq sk < q$, and the public verifying key is a point $[sk]G$. To sign a message hash h, the signer first generates a random integer nonce $0 \leq k < q$, then computes the signature $(r, s) = (([k]G)_x, |k^{-1} \cdot (h + sk \cdot r)|_q)$, where x subscript represents the x coordinate of the point.

ECDSA as a HNP. In a side-channel attack against ECDSA, the adversary may know l least significant bits of the nonce k. If we write $k = k_{msb} \cdot 2^l + k_{lsb}$, where $0 \leq k_{msb} < q/2^l$ and $0 \leq k_{lsb} < 2^l$, then we can obtain the following equation based on $s = |k^{-1} \cdot (h + r \cdot sk)|_q$:

$$2^{-l}(k_{lsb} - s^{-1} \cdot h) + k_{msb} = 2^{-l} \cdot s^{-1} \cdot r \cdot sk \bmod q.$$

This can be regarded as a HNP instance with $(t_i, a_i) = (|2^{-l} \cdot s^{-1} \cdot r|_q, |2^{-l} \cdot (k_{lsb} - s^{-1} \cdot h)|_q)$ and hidden number $\alpha = sk$. In the case where the most significant bits of the nonce are leaked, the method of transformation into a HNP instance remains similar.

For convenience, we abbreviate $\mathrm{ECDSA}(n, l)$ as an n-bit ECDSA instance with l-bit leakage.

2.5 Solving the HNP with Lattices

To solve the HNP, in 1996, Boneh and Venkatesan [12] constructed the $(m+1)$-dimensional lattice generated by the rows of the following matrix:

$$\begin{bmatrix} q & 0 & \cdots & 0 & 0 \\ 0 & q & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & q & 0 \\ t_0 & t_1 & \cdots & t_{m-1} & 1/2^l \end{bmatrix}.$$

There exists a lattice vector $\boldsymbol{v} = (t_0 \cdot \alpha \bmod q, \ldots, t_{m-1} \cdot \alpha \bmod q, \alpha/2^l)$. This lattice vector is close to the target vector $\mathbf{t} = (a_0, \ldots, a_{m-1}, 0)$ since the distance $\|\boldsymbol{v} - \mathbf{t}\|$ can be bounded by $\sqrt{m+1} \cdot q/2^l$. If the distance is sufficiently small compared with other lattice vectors, the lattice vector can be found by solving the BDD problem using the nearest plane algorithm [7], or Kannan's embedding [23]. With Kannan's embedding, we construct the $(m+2)$-dimensional lattice basis

$$\begin{bmatrix} q & 0 & \cdots & 0 & 0 & 0 \\ 0 & q & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & q & 0 & 0 \\ t_0 & t_1 & \cdots & t_{m-1} & 1/2^l & 0 \\ a_0 & a_1 & \cdots & a_{m-1} & 0 & \tau \end{bmatrix},$$

where τ is the embedding number, which can be set to the upper bound of k_i, i.e., $\tau = q/2^l$. The target lattice vector becomes $\boldsymbol{v} = \pm(k_0, \ldots, k_{m-1}, \alpha/2^l, -\tau)$ with a bounded norm of $\sqrt{m+2} \cdot q/2^l$.

Recentering Technique. In the definition of HNP, $0 \leq k_i < q/2^l$, for $i = 0, 1, \ldots, m-1$. Since the lattice can work for any sign of k_i, we can make a variable substitution $k_i' = k_i - w$ where $w = q/2^{l+1}$. The target vector becomes $(k_0', k_1', \ldots, k_{m-1}', \alpha/2^l, -\tau)$ and has a much shorter length. This technique can bring a significant improvement in practice and is widely used in the lattice attacks on HNP [3,13,28,29,38].

Elimination Method. Given a set of HNP equations $a_i + k_i = t_i\alpha \bmod q$, where $i = 0, 1, \cdots, m-1$, we can eliminate the variable α by substituting $\alpha = |(a_0 + k_0)t_0^{-1}|_q$. This yields a new set of HNP equations. Incorporating the recentering technique, we have

$$a_i + w - (a_0 + w)t_0^{-1}t_i + k_i' = t_0^{-1}t_i k_0' \bmod q,$$

where $i = 1, \ldots, m-1$. This produces a transformed HNP instance $(t_i', a_i') = (t_0^{-1}t_i, a_i + w - (a_0 + w)t_0^{-1}t_i)$ with the transformed hidden number k_0'. Then the new lattice is generated by:

$$\begin{bmatrix} q & 0 & \cdots & 0 & 0 & 0 \\ 0 & q & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & q & 0 & 0 \\ t'_1 & t'_2 & \cdots & t'_{m-1} & 1 & 0 \\ a'_1 & a'_2 & \cdots & a'_{m-1} & 0 & \tau \end{bmatrix},$$

referred to as the Albrecht and Heninger's lattice [3]. The lattice dimension, denoted as d, is reduced to $m+1$. The target lattice vector becomes $\boldsymbol{v} = \pm(k'_1, \ldots, k'_{m-1}, k'_0, -\tau)$, with a bounded norm of $\sqrt{mw^2+\tau^2}$. To find the target vector using a black box SVP solver, we expect the target vector to be the shortest vector, i.e., $\|\boldsymbol{v}\| \leq \sqrt{mw^2+\tau^2} \leq \mathrm{GH}(\mathcal{L})$. Combining this with $d = m+1$, we can estimate the minimal lattice dimension. Moreover, instead of using the upper bound of $\|\boldsymbol{v}\|$, Albrecht and Heninger considered the expected squared norm [3], i.e., $\mathbb{E}[\|\boldsymbol{v}\|^2] = mw^2/3 + m/6 + w^2$. Then the minimal lattice dimension can be estimated as the minimal integer d satisfying $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] \leq \mathrm{GH}^2(\mathcal{L})$.

Sieving with Predicate. The sieving with predicate (Sieve-Pred) algorithm checks over the database generated by sieving algorithms using the predicate [3]. It does not treat the sieving algorithm as a black box SVP solver. This idea is inspired by Micciancio et al. [27], which suggests that the sieving algorithm not only outputs the shortest vector, but also provides all vectors with norm less than $\sqrt{4/3}\,\mathrm{GH}(\mathcal{L})$, under specific heuristic assumptions. Algorithm 1 is expected to find the target vector under Assumption 1 stated below.

Assumption 1 ([15]) *When a 2-sieve algorithm terminates, it outputs a database L containing all vectors with norm $\leq \sqrt{4/3}\,\mathrm{GH}(\mathcal{L})$.*

Theorem 1 ([3]). *Let $\mathcal{L} \subset \mathbb{R}^d$ be a lattice containing a vector v such that $\|\boldsymbol{v}\| \leq \sqrt{4/3}\,\mathrm{GH}(\mathcal{L})$. Under Assumption 1, Algorithm 1 is expected to find the minimal $\boldsymbol{v}$ satisfying $f(\boldsymbol{v}) = 1$ in $2^{0.292d+o(d)}$ steps and $(4/3)^{d/2+o(d)}$ calls to the predicate $f(\cdot)$.*

Algorithm 1: Sieving with Predicate

```
  Input: Lattice $\mathcal{L}(\boldsymbol{B})$, predicate $f(\cdot)$
  Output: $\boldsymbol{v}$ such that $\|\boldsymbol{v}\| \leq \sqrt{4/3}\,\mathrm{GH}(\mathcal{L})$ and $f(\boldsymbol{v}) = 1$ or $\perp$
1 $\boldsymbol{r} \leftarrow \perp$ ;
2 Run the sieving algorithm on $\mathcal{L}(\boldsymbol{B})$ and output list $L$;
3 for $\boldsymbol{v} \in L$ do
4     if $f(\boldsymbol{v}) = 1$ and ($r = \perp$ or $\|\boldsymbol{v}\| < \|\boldsymbol{r}\|$) then
5         $\boldsymbol{r} \leftarrow \boldsymbol{v}$;
6 return $\boldsymbol{r}$;
```

Thus, using the Sieve-Pred algorithm, the minimal lattice dimension can be estimated as the minimal integer d satisfying $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] \leq 4\,\mathrm{GH}^2(\mathcal{L})/3$.

3 Improved Algorithms

In this section, we propose several new algorithms for solving the HNP. We decompose the hidden number k'_0 as $x \cdot \alpha_0 + \alpha_1$, and introduce a new lattice construction that uses x to trade off the lattice dimension. Through theoretical analysis, we demonstrate that our lattice can offer a reduction of approximately $(\log x)/l$ compared to the lattice presented by Albrecht and Heninger [3]. While the target vector contains information about α_0, the lack of information about α_1 makes it impossible to directly compute the hidden number α. This makes previous predicates ineffective [3,38]. Due to the idea of Sieve-Pred algorithm [3], we need to search within the exponentially large database output by the sieving algorithms. Therefore, an efficient process that excludes incorrect candidates is essential. To address this issue, we propose a prescreening technique, an interval reduction algorithm, and a linear predicate. Moreover, we show that under Assumption 1, there exists a constant $c > 0$ such that the success probability of our algorithm is at least c.

3.1 New Lattice Construction Based on Decomposition Technique

The main idea behind our attack is using more HNP samples to obtain equations with small t'_i. Having many samples with small t'_i allows us to increase the lattice determinant, while keeping the norm of the target vector roughly unchanged. This enables us to reduce the lattice dimension, while still satisfying the enabling condition $\mathbb{E}[\|v\|^2] \leq 4/3\,\mathrm{GH}^2(\mathcal{L})$ for the attack.

Lattice Construction. Following Albrecht and Heningerąŕs lattice [3], we construct the lattice generated by

$$\begin{bmatrix} q & 0 & \cdots & 0 & 0 & 0 \\ 0 & q & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & q & 0 & 0 \\ x \cdot t'_1 & x \cdot t'_2 & \cdots & x \cdot t'_{m-1} & y & 0 \\ a'_1 & a'_2 & \cdots & a'_{m-1} & 0 & \tau \end{bmatrix},$$

where $x, y > 0$, and τ are undetermined coefficients. This modification increases Albrecht and Heninger's original lattice volume by a factor of y.

In [34], Sun et al. decomposed the hidden number α as $\alpha = 2^c \cdot \alpha_0 + \alpha_1$, where $0 \leq \alpha_1 < 2^c$. We propose a more generalized form of decomposition to the new hidden number: $k'_0 = x \cdot \alpha_0 + \alpha_1$, where $|\alpha_1| \leq x/2$. Here, x is an arbitrary positive integer and $|\alpha_1|$ is 1-bit smaller. Then, for $i = 1, \ldots, m-1$, we have $x \cdot t'_i \cdot \alpha_0 - a'_i \equiv k'_i - \alpha_1 \cdot t'_i \mod q$. The target vector in the lattice becomes

$$\boldsymbol{v} = \pm(k'_1 - \alpha_1 \cdot t'_1, \ldots, k'_{m-1} - \alpha_1 \cdot t'_{m-1}, y \cdot \alpha_0, -\tau).$$

To keep the norm of the target vector approximately the same as in Albrecht and Heningerąŕs lattice [3], we need to make the ratio $r = \mathbb{E}\left[(k'_i - \alpha_1 t'_i)^2\right] / \mathbb{E}\left[(k'_i)^2\right]$

close to 1. Now, we show how to achieve this. The following analysis is based on the assumption that α_1 is independent of k_i' and a_i'.

Recall that k_i' and t_i' are uniformly distributed in $[-w, w)$ and $[-q/2, q/2)$, respectively. By setting the upper bound of $|t_i'|$ to be B, t_i' becomes uniformly distributed in $[-B, B)$. Moreover, to make α_1 close to a uniform distribution over $[-x/2, x/2)$, the condition $x \ll w$ should be satisfied. Thus, we can set a theoretical upper bound $w/2^{10} = q/(2^{l+11})$ for x. Under these conditions, we have:

$$r = \frac{\mathbb{E}\left[(k_i' - \alpha_1 t_i')^2\right]}{\mathbb{E}\left[(k_i')^2\right]} = \frac{\mathbb{E}\left[(k_i')^2\right] - 2\mathbb{E}\left[\alpha_1\right]\mathbb{E}\left[k_i' t_i'\right] + \mathbb{E}\left[\alpha_1^2\right]\mathbb{E}\left[t_i'^2\right]}{\mathbb{E}\left[(k_i')^2\right]}$$
$$= 1 + \mathbb{E}\left[\alpha_1^2\right]\frac{\mathbb{E}\left[t_i'^2\right]}{\mathbb{E}\left[(k_i')^2\right]}.$$

Note that if an integer variable k is uniformly distributed over $[-w, w)$, then $\mathbb{E}\left[k^2\right] = w^2/3 + 1/6$. Thus, we have

$$\begin{aligned} r &= 1 + \left(\frac{x^2}{12} + \frac{1}{6}\right)\left(\frac{2B^2+1}{2w^2+1}\right) \\ &\le 1 + \left(\frac{x^2}{12} + \frac{1}{6}\right)\left(\frac{B^2}{w^2} + \frac{1}{2w^2+1}\right) \le 1 + 2^{-20} + \left(\frac{x^2}{12} + \frac{1}{6}\right)\frac{B^2}{w^2}. \end{aligned}$$

It can be seen that as B decreases, r approaches 1. However, the expected number of required samples increases by a factor of $q/(2B)$. To balance keeping r close to 1 and minimizing the number of required samples, we set $B = w/(2^3 x) = q/(2^{l+4}x)$. With this setting, we only need $2^{l+3}x$ times the original number of samples, and r is approximately $1+(x^2B^2+2B^2)/(12w^2) = 1+1/768+1/(384x^2)$, which is close to 1.

Next, we focus on the selection of parameters to optimize the performance of our attack. Given that $|k_0'|$ is bounded by w, we know that $|\alpha_0| \le w/x$. Then we have

$$\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] = (m-1)\frac{w^2}{3} + y^2\frac{w^2}{3x^2} + \tau^2.$$

Our goal is to minimize the ratio $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] / \mathrm{GH}^2(\mathcal{L})$, where

$$\mathrm{GH}^2(\mathcal{L}) = \frac{(m+1)}{2\pi e} \cdot q^{\frac{2(m-1)}{m+1}} \cdot y^{\frac{2}{m+1}} \cdot \tau^{\frac{2}{m+1}}.$$

Given x, according to the AMÍCGM inequality, we have

$$\begin{aligned} \mathbb{E}\left[\|\boldsymbol{v}\|^2\right] &= \underbrace{\frac{w^2}{3} + \cdots + \frac{w^2}{3}}_{m-1} + y^2\frac{w^2}{3x^2} + \tau^2 \\ &\ge (m+1)\left(\left(\frac{w^2}{3}\right)^{m-1} \cdot y^2\frac{w^2}{3x^2} \cdot \tau^2\right)^{1/(m+1)} \\ &= (m+1)\left(\frac{w^2}{3}\right)^{m/(m+1)} \cdot x^{-\frac{2}{m+1}} \cdot y^{\frac{2}{m+1}} \cdot \tau^{\frac{2}{m+1}}. \end{aligned}$$

Thus, $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] / \mathrm{GH}^2(\mathcal{L})$ attains its minimum value when $y = x$ and $\tau = w/\sqrt{3}$.

Reduction of Lattice Dimension. The new lattice construction can lead to a reduction in the lattice dimension, which is described in Theorem 2. This significantly improves the efficiency of lattice-based attacks, as the time complexity of sieving algorithms increases exponentially with an increase in the lattice dimension d.

Theorem 2. *For any positive integer x and the number of leaked bits l, the reduction of lattice dimension between our lattice and Albrecht and Heninger's lattice is given by*

$$\frac{2\log x}{2l + 3 - \log(\pi e)} \approx \frac{\log x}{l}.$$

Proof. According to Assumption 1, the lattice dimension is the minimal integer d satisfying $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] \leq 4/3 \cdot \mathrm{GH}^2(\mathcal{L})$. In Albrecht and Heninger's lattice [3], let $\tau = w/\sqrt{3}$, then we have

$$\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] = m\left(\frac{w^2}{3} + \frac{1}{6}\right) + \tau^2 \approx d \cdot \frac{w^2}{3},$$

$$\mathrm{GH}^2(\mathcal{L}) = \frac{d}{2\pi e} \cdot q^{\frac{2(d-2)}{d}} \cdot \left(\frac{w^2}{3}\right)^{\frac{1}{d}}.$$

Substituting $w = q/2^{l+1}$, and taking the logarithm, we have

$$d \geq \frac{2l + 2 + 2\log q + \log 3}{2l + 3 - \log(\pi e)}.$$

In our lattice, $\mathbb{E}[\|\boldsymbol{v}\|^2]$ remains approximately the same, but $\mathrm{GH}^2(\mathcal{L})$ increases by a factor of $x^{2/d}$. Thus, the new lattice dimension d' needs to satisfy

$$d' \geq \frac{2l + 2 + 2\log q + \log 3 - 2\log x}{2l + 3 - \log(\pi e)}.$$

The reduction of lattice dimension is given by

$$\frac{2\log x}{2l + 3 - \log(\pi e)} \approx \frac{\log x}{l}.$$

□

Figure 1 illustrates the reduction of lattice dimension as x increases. Four lines are plotted in Figure 1, representing $\mathrm{HNP}(160, 1)$, $\mathrm{HNP}(192, 1)$, $\mathrm{HNP}(224, 1)$, and $\mathrm{HNP}(384, 2)$, respectively. Solving these instances is believed to be impractical by previous lattice-based approaches [3,5,6,34,38]. However, our experimental results in Section 5 demonstrate the feasibility of solving $\mathrm{HNP}(160, 1)$ by using a large x. More computational resources and samples are required for solving more difficult instances such as $\mathrm{HNP}(192, 1)$.

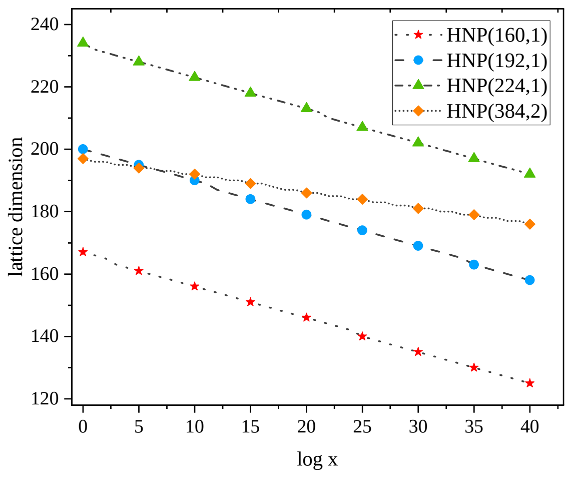


Fig. 1: Lattice dimension and x.

Success Probability. Under Assumption 1, the sieving algorithm outputs all vectors satisfying $\|\boldsymbol{v}\|^2 \leq 4\,\mathrm{GH}^2(\mathcal{L})/3$. Consequently, the probability that the sieving algorithm can output the target vector is represented as $\Pr(\|\boldsymbol{v}\|^2 \leq \mathbb{E}\left[\|\boldsymbol{v}\|^2\right])$, which is the success probability of our algorithm.

Theorem 3. *Let $\boldsymbol{v}$ be the target vector of our lattice described in Section 3.1. For all $d \geq 3$, there exists a constant $c > 0$ such that* $\Pr(\|\boldsymbol{v}\|^2 \leq \mathbb{E}\left[\|\boldsymbol{v}\|^2\right]) \geq c$.

Proof. Recall the vector representation

$$\begin{aligned}\boldsymbol{v} &= (k'_1 - \alpha_1 t'_1, \ldots, k'_{m-1} - \alpha_1 t'_{m-1}, x\alpha_0, -w/\sqrt{3}) \\ &= (v_0, v_{d-3}, v_{d-2}, v_{d-1}).\end{aligned}$$

For k'_i, t'_i, α_0, and α_1, they are all uniformly distributed: k'_i is over $[-w, w)$, t'_i is over $[-w/(8x), w/(8x))$, α_0 is over $[-w/x, w/x)$, and α_1 is over $[-x/2, x/2)$. It follows that $v_0, \ldots, v_{d-3}$ are independent and identically distributed.

Let $\mathbb{P}_d = \Pr\left(\frac{1}{d-2}\sum_{i=0}^{d-3} v_i^2 \leq \mathbb{E}\left[v_0^2\right]\right)$. Then we have

$$\begin{aligned}\Pr\left(\|\boldsymbol{v}\|^2 \leq \mathbb{E}\left[\|\boldsymbol{v}\|^2\right]\right) &= \Pr\left(\sum_{i=0}^{d-3} v_i^2 + x^2\alpha_0^2 \leq (d-2)\mathbb{E}\left[v_0^2\right] + \frac{w^2}{3}\right) \\ &\geq \Pr\left(\sum_{i=0}^{d-3} v_i^2 \leq (d-2)\mathbb{E}\left[v_0^2\right]\right) \cdot \Pr\left(x^2\alpha_0^2 \leq \frac{w^2}{3}\right) \\ &= \frac{\sqrt{3}}{3}\Pr\left(\frac{1}{d-2}\sum_{i=0}^{d-3} v_i^2 \leq \mathbb{E}\left[v_0^2\right]\right) = \frac{\sqrt{3}}{3}\mathbb{P}_d.\end{aligned}$$

Since $\mathrm{Var}[v_0^2] < +\infty$, by the Central Limit Theorem, it holds that

$$\lim_{d\to+\infty} \Pr\left(\frac{1}{d-2}\sum_{i=0}^{d-3} v_i^2 \le \mathbb{E}\left[v_0^2\right]\right) = \frac{1}{2}.$$

Hence, there exists a positive constant $c_1 > 0$ such that $\mathbb{P}_d \ge c_1$ for any $d \ge 3$. Let $c = c_1/\sqrt{3}$. Finally, we get

$$\Pr\left(\|\boldsymbol{v}\|^2 \le \mathbb{E}\left[\|\boldsymbol{v}\|^2\right]\right) \ge \frac{\sqrt{3}}{3}\mathbb{P}_d \ge c.$$

□

3.2 Improved Linear Predicate

In Albrecht and Heninger's approach, they employ non-linear constraints as a predicate to determine the unique hidden number [3,4]. Their predicate initially checks whether the absolute value of the last element of the candidate vector is τ. Subsequently, it determines the target vector by checking whether r is equal to $([k]G)_x$, where G is the generator point on the curve, r is a specific signature, and k is the corresponding nonce that can be computed from the candidate vector. The predicate involves time-consuming scalar multiplication on the curve.

In this section, we propose an improved linear predicate that utilizes linear constraints from $2\log q$ HNP samples and only involves two vector inner products, providing a significant efficiency advantage. Moreover, we present a modified version of the Sieve-Pred algorithm to integrate our predicate, and achieve higher efficiency.

Linear Predicate. Our linear predicate is described in Algorithm 2. It operates on a 2-dimensional vector $\boldsymbol{v} = (v_0, v_1)$, representing the last two elements of a candidate vector in the database. This predicate determines whether the candidate vector satisfies a set of linear conditions. If these conditions are met, the predicate reveals the hidden number; otherwise, it returns $\perp$. The algorithm follows these steps:

(1) Check if $0 < |v_0| \le w$ and whether $|v_1|$ equals $\pm\tau$. If this condition is met, proceed to the next step; otherwise, return $\perp$.
(2) Calculate the candidate α' from $\boldsymbol{v}$ and the HNP sample (t_0, a_0).
(3) For N HNP samples (t_i, a_i), check whether $|t_i \cdot \alpha' - a_i|_q < q/2^l$ for $i = 0, \ldots, N-1$. If this condition holds, return the candidate α' as the correct hidden number; otherwise, return $\perp$.

Note that the HNP samples used in step (3) are distinct from those used in our lattice construction.

Theoretical Analysis of Linear Predicate. According to the definition of HNP, the correct candidate α' should satisfy all the constraints from these HNP samples. However, for an incorrect candidate, there still exists a small chance

Algorithm 2: Improved Linear Predicate

```
  Input: A 2-dimensional vector $\boldsymbol{v} = (v_0, v_1)$, modulus $q$, number of nonce
         leakage $l$, embedding number $\tau$, $N = 2\log q$ HNP samples $(t_i, a_i)$
  Output: The hidden number $\alpha$ or $\perp$
1 if $v_0 = 0$ or $|v_0| > q/2^{l+1}$ or $|v_1| \neq \tau$ then
2     return $\perp$;
3 $k_0 \leftarrow -\operatorname{sign}(v_1) \cdot v_0 + q/2^{l+1}$ ;
4 $\alpha' \leftarrow t_0^{-1} \cdot (a_0 + k_0) \bmod q$ ;
5 for $i = 0$ to $N - 1$ do
6     if $|t_i \cdot \alpha' - a_i|_q \geq q/2^l$ then
7         return $\perp$;
8 return $\alpha'$;
```

of meeting all the constraints. Consider the case where Algorithm 2 receives candidate values for the hidden number α' from the range $[0, q-1]$, rather than computing alpha from the vectors obtained by the sieving algorithm. In this case, α' is uniformly distributed over $[0, q-1]$. When t_i and a_i are fixed, $|t_i\alpha' - a_i|_q$ is also uniformly distributed over $[0, q-1]$. Therefore, for $i = 0, 1, \ldots, N-1$, we have

$$\Pr\left(|t_i \cdot \alpha' - a_i|_q < q/2^l\right) = 2^{-l}.$$

Let $N = 2\log q$. Then an incorrect candidate α' satisfies all the constraints with probability $2^{-2l\log q} = q^{-2l}$. Since there are $q-1$ incorrect candidate α', the probability that the algorithm can find the candidate in the interval $[0, q-1]$ is given by

$$(1 - q^{-2l})^{q-1} \geq 1 - (q-1)q^{-2l} \geq 1 - \frac{q-1}{q^2} = 1 - \operatorname{negl}(\log q).$$

Here, we prove that the probability of the algorithm correctly identifying all q inputs $\{0, ..., q-1\}$ is very close to 1. Consequently, it can also correctly identify whether the hidden number candidates obtained from the lattice are correct.

The expected number of verifying operations for each candidate can be calculated as follows. As mentioned earlier, the samples used in lattice construction and linear predicate are distinct from each other. This allows us to assume that the values of $|t_i\alpha - a_i|_q$ are uniformly distributed over the range $[0, q-1]$. Consider a scenario with M candidates, under this assumption, the expected number of candidates meeting the first constraint is $M/2^l$, as only a fraction of $1/2^l$ candidates will satisfy the condition. Similarly, the expected number of candidates meeting the second constraint is $M/2^{2l}$, and so on. Therefore, the total expected number of verifying operations needed for M candidates is $M + M/2^l + M/2^{2l} + \cdots = M/(1-2^{-l})$. Consequently, only $1/(1-2^{-l})$ verifying operations on average are needed for each candidate, and the parameter $2\log q$ does not affect the efficiency of our predicate.

Comparison with the predicate in [38]. In [38], Xu et al. introduce a linear predicate that requires a d-dimensional vector as input. In the sieving implementation G6K [2], vectors in the output database are represented as coordinates under the lattice basis. To obtain all positions of the candidate vector, one needs to perform d vector inner products, i.e., multiply the d-dimensional coordinate vector by the $d \times d$ lattice basis matrix. While our predicate only involves two vector inner products, which provides a notable efficiency advantage in practice.

In addition, our predicate uses linear constraints from new HNP samples, rather than those used in the lattice construction[2]. The reason is that the vector $\boldsymbol{v}$ in the sieving database is inherently shorter. If HNP samples (t_i, a_i) from the lattice construction are used, the probability $\Pr(|t_i \cdot \alpha' - a_i| < q/2^{l+1})$ (equivalent to $\Pr(|v_i| < q/2^{l+1})$) will be higher. Consequently, more linear constraints need to be verified to identify a false candidate.

Furthermore, we would like to point out an issue with the predicate in [38]. Their predicate may return true for some incorrect candidates, which is not as expected. The underlying reason can be explained as follows: in Albrecht and Heninger's lattice, for any $\theta \in \mathbb{N}^+$, consider the following lattice vector:

$$v_\theta = (|\theta t'_1 - a'_1|_q, \ldots, |\theta t'_{d-2} - a'_{d-2}|_q, \theta, -\tau)$$

The predicate in [38] first checks whether v_{d-1} is equal to $\pm\tau$, and then computes the candidate α as $t_0^{-1}(\theta + q/2^{l+1} + a_0)$. Then it continues to check whether $a_i + v_{i-1} + q/2^{l+1}$ is equal to $t_i\alpha$ (Lines 6 and 13 in Algorithm 3 [38]). However, we always have $a_i + v_{i-1} + q/2^{l+1} = t_i\alpha \bmod q$. Specifically, we compute:

$$a_i + v_{i-1} + q/2^{l+1} = a_i + q/2^{l+1} + \theta t'_i - a'_i$$

Substituting $a'_i = a_i + q/2^{l+1} - (a_0 + q/2^{l+1})t_0^{-1}t_i$ and $t'_i = t_0^{-1}t_i \bmod q$ into the above equation, we get:

$$a_i + v_{i-1} + q/2^{l+1} = t_i t_0^{-1}(\theta + q/2^{l+1} + a_0) = t_i\alpha \bmod q$$

This indicates that the predicate in [38] will return true for any θ and vector v_θ. To enhance performance, the conditions in Lines 6 and 13 of Algorithm 3 [38] could be modified to $|v_{i-1}|_q > q/2^{l+1}$.

Modified Sieving with Predicate. As we have made modifications to both the input and output of the predicate, a modified version of the Sieve-Pred algorithm is proposed to ensure compatibility with the new predicate. The algorithm is outlined in Algorithm 3. It no longer takes the entire lattice vector as input but only the last two elements. Moreover, the algorithm immediately outputs the solution when our linear predicate returns true, instead of searching the entire database.

[2] In fact, the samples used in the lattice construction, the linear predicate, the interval reduction algorithm, and the prescreening technique are all distinct from each other.

Algorithm 3: Modified Sieving with Predicate

```
   Input: Lattice $\mathcal{L}(\boldsymbol{B})$ with dimension $d$, predicate $f(\cdot)$
   Output: The hidden number $\alpha$ or $\perp$
1  Run the sieving algorithm on $\mathcal{L}(\boldsymbol{B})$ and output list $L$;
2  for $\boldsymbol{v} \in L$ do
3      $\alpha \leftarrow f(v_{d-2}, v_{d-1})$;
4      if $\alpha \neq \perp$ then
5          return $\alpha$;
6  return $\perp$;
```

3.3 Predicate for Decomposition Technique

In Section 3.1, the transformed hidden number k'_0 is decomposed as $\alpha_0 x + \alpha_1$ where $|\alpha_1| \leq x/2$, and the target vector is $\boldsymbol{v} = \pm(k'_1 - \alpha_1 t'_1, \ldots, k'_{m-1} - \alpha_1 t'_{m-1}, x\alpha_0, -\tau)$. This vector contains information about α_0, which is a part of k'_0. However, due to the absence of information about α_1, it is impossible to directly compute the entire k'_0. Consequently, previous predicates are ineffective in this scenario [3,38].

The straightforward approach to recover k'_0 is to perform an exhaustive search over all possible values of α_1, which has time complexity $O(x)$. For each candidate value of α_0, we need to check the predicate for x candidate values of the hidden number. An exhaustive search will result in a substantial time overhead and becomes impractical when x is large. To address this issue, we introduce an interval reduction algorithm that reduces the complexity from $O(x)$ to $O(\log^2 x)$. Based on this algorithm, a predicate for the decomposition technique is also proposed.

There are two necessary notations. Assume that R is the union of a set of intervals. Let $|R|$ be the number of intervals in R, and $||R||$ be the number of integers within the intervals in R. For example, if $R = \{[1, 4]\}$, then $|R| = 1$ which means that there is one interval in R, and $||R|| = 4$ which means that there are four integers within the interval $[1, 4]$.

Interval Reduction Algorithm. The interval reduction algorithm takes an interval of length M and $\log M$ transformed HNP samples as input, and produces a set of smaller intervals as output. We denote the input interval as $[\text{low}, \text{high}]$, where $M = \text{high} - \text{low} + 1$. Let R be the set of intervals output by this algorithm. The interval reduction algorithm guarantee that if the hidden number is in the input interval $[\text{low}, \text{high}]$, then the hidden number must be in one interval of R. This algorithm is described in Algorithm 4, and it contains two steps as follows.

Firstly, we generate the intervals based on $\log M$ transformed samples (t'_i, a'_i). These intervals are computed using the HNP equations and the interval $[\text{low}, \text{high}]$. It holds that

$$t'_i \cdot \text{low} \leq t'_i k'_0 = a'_i + k'_i + nq \leq t'_i \cdot \text{high},$$

Algorithm 4: Interval Reduction Algorithm

```
   Input: interval [low, high] that may contain the hidden number $k'_0$, modulus $q$,
          parameter $l$, $N = \log M$ transformed samples $(t'_i, a'_i)$ satisfying
          $t'_i = O(q/M)$
   Output: set $R$ of intervals that may contain $k'_0$
1  $R \leftarrow \{[\text{low}, \text{high}]\}$;
2  for $i = 0$ to $N - 1$ do
3      Generate a new interval set $R_{\text{new}}$ based on $i$-th sample $(t'_i, a'_i)$;
4      $R \leftarrow \text{IntervalSetIntersection}(R, R_{\text{new}})$;
5  return R;
```

for $i = 1, \ldots, \log M$. Since $-w \leq k'_i \leq w - 1$, we get a set $\mathcal{S}$ that contains all possible values of n. For a specific n_1 in $\mathcal{S}$, we have

$$t'_i k'_0 = a'_i + k'_i + n_1 q \in [a'_i + n_1 q - w, a'_i + n_1 q + w - 1].$$

Therefore,

$$t'_i k'_0 \in \bigcup_{n \in \mathcal{S}} [a'_i + nq - w, a'_i + nq + w - 1].$$

This process yields a set of intervals that are sorted in ascending order, and one of these intervals may contain k'_0. To limit the number of intervals in this set, we require that $t'_i = O(q/M)$. For the rationale of this requirement, the reader is referred to the proof of Theorem 4.

Secondly, we intersect all the sets of intervals generated from the $\log M$ samples. For this operation, we present an interval set intersection algorithm in Algorithm 5. It takes two sets of intervals that are in ascending order as input and outputs their intersection. The time complexity of Algorithm 5 is $O(m+n)$, where m and n are the numbers of intervals in two sets respectively.

Predicate for decomposition technique. Our predicate for the decomposition technique is described in Algorithm 6. Firstly, this algorithm verifies whether $|v_0| < q/2^{l+1}$ and whether $|v_1|$ equals $\pm\tau$. Secondly, it computes the interval [low, high] that may contain k'_0. Thirdly, the interval reduction algorithm is employed to generate a set R. Finally, an exhaustive search is carried out to check the linear predicate for each integer in the intervals of R. The expected time complexity of this algorithm is $O(\log^2 x)$, as shown in Theorem 4.

Theorem 4. *The expected time complexity of Algorithm 6 is* $O(\log^2 x)$.

Proof. There are two parts in Algorithm 6. The first part is the interval reduction algorithm, and the second part is an exhaustive search.

(1) Let us study the time complexity of the first part. We need to bound $|R_{\text{new}}|$, where R_{new} is given in Algorithm 4. For any transformed sample (t'_i, a'_i), we have both $nq + a'_i + w - 1 \geq t'_i \cdot \text{low}$ and $nq + a'_i - w \leq t'_i \cdot \text{high}$. Hence, the

Algorithm 5: Interval Set Intersection Algorithm

```
   Input: Two sets of intervals in ascending order: $A$, $B$
   Output: Intersection of $A$ and $B$
1  $i, j \leftarrow 0$;
2  $R \leftarrow \{\}$ ;
3  while $i < |A|$ and $j < |B|$ do
4      Let $[a_0, a_1]$ and $[b_0, b_1]$ be the $i$-th and $j$-th intervals in A and B,
       respectively;
5      if $a_1 < b_0$ then
6          $i \leftarrow i + 1$;
7          Continue;
8      if $b_1 < a_0$ then
9          $j \leftarrow j + 1$;
10         Continue;
11     if $a_1 \geq b_1$ then
12         $j \leftarrow j + 1$;
13         Add the interval $[\max(a_0, b_0), b_1]$ to $R$;
14         Continue;
15     $i \leftarrow i + 1$;
16     Add the interval $[\max(a_0, b_0), a_1]$ to $R$;
17 return $R$;
```

Algorithm 6: Predicate for Decomposition Technique

```
   Input: A 2-dimensional vector $\boldsymbol{v} = (v_0, v_1)$, modulus $q$, number of nonce
          leakage $l$, embedding number $\tau$, predicate $f(\cdot)$
   Output: hidden number $\alpha$ or $\perp$
1  if $v_0 = 0$ or $|v_0| > q/2^{l+1}$ or $|v_1| \neq \tau$ then
2      return $\perp$;
3  low $\leftarrow -\mathrm{sign}(v_1) \cdot v_0 - x/2$ , high $\leftarrow -\mathrm{sign}(v_1) \cdot v_0 + x/2$;
4  $R \leftarrow$ IntervalReduction([low, high]);
5  for $[a, b] \in R$ do
6      for $h = a$ to $b$ do
7          $\alpha \leftarrow f(h, -\tau)$;
8          if $\alpha \neq \perp$ then
9              return $\alpha$;
10 return $\perp$ ;
```

number of possible values of n is not bigger than

$$\frac{t'_i \cdot (\text{high} - \text{low}) + 2w - 1}{q} + 1 = \frac{t'_i \cdot x - t'_i + 2w - 1}{q} + 1.$$

Since $t'_i = O(q/x)$, there exists a constant C such that $|R_{\text{new}}| \leq C$.

For $i = 0, 1, \ldots, N - 1$, let R_i be the set returned by the interval set intersection algorithm in Algorithm 4 just after i-th sample. Then we have $|R_i| \leq (i + 1)C$. Thus, for any $0 \leq i \leq N - 1$, the time complexity of i-th step in

Algorithm 4 is at most $O((i+2)C)$. It follows that the time complexity of the first part is $O(N^2) = O(\log^2 x)$.

(2) For the time complexity of the second part, let $R_{-1} = [\text{low}, \text{high}]$. Then $\|R_{-1}\| = x$. For an incorrect hidden number candidate, the probability that this candidate satisfies the constraint given by each HNP sample is only $1/2^l$. Thus, for $0 \leq i \leq N-1$, we have

$$\frac{\mathbb{E}(\|R_i\|)}{\mathbb{E}(\|R_{i-1}\|)} = \frac{1}{2^l}.$$

Finally, we get

$$\mathbb{E}(\|R_{N-1}\|) = \mathbb{E}(\|R_{-1}\|) \cdot \left(\frac{1}{2^l}\right)^N \leq \frac{x}{2^N} = 1.$$

Therefore, the expected time complexity of the second step is $O(1)$.

By (1) and (2), the expected time complexity of Algorithm 6 is $O(\log^2 x)$. □

Pre-screening Technique. Before running the interval reduction algorithm in Algorithm 6, a pre-screening technique can be used to eliminate the majority of incorrect candidates. This technique involves only a few linear operations and can significantly enhance the efficiency of Algorithm 6.

The pre-screening technique makes use of a small number of transformed HNP samples (t_i', a_i'), where $|t_i'| \leq q/(2^{l+3}x)$. For an incorrect candidate α_0', it will be rejected if the following condition is satisfied:

$$\left| |x \cdot t_i' \cdot \alpha_0' - a_i' + \frac{q}{2}|_q - \frac{q}{2} \right| > w + \frac{q}{2^{l+4}}.$$

Let us explain the reason. For any correct candidate α_0, we have $x \cdot t_i'\alpha_0 - a_i' \equiv k_i' - \alpha_1 t_i' \mod q$. Therefore, we get

$$|k_i' - \alpha_1 t_i'| \leq |k_i'| + |\alpha_1| \cdot |t_i'| \leq w + \frac{x}{2} \cdot \frac{q}{2^{l+3}x} = w + \frac{q}{2^{l+4}}.$$

Note that this technique does not increase the sampling cost, as we can use the samples that satisfy $q/(2^{l+4}x) \leq |t_i'| \leq q/(2^{l+3}x)$ for pre-screening. These samples are already available during the pre-selection of the samples used to construct the lattice in Section 3.1.

To illustrate the efficiency improvements brought by the interval reduction algorithm and pre-screening technique, we conduct an experiment on HNP$(256, 2)$ with $x = 2^{15}$. We record the time taken to search the database using different methods on a single thread. The experimental data demonstrates that, compared with the exhaustive search, the interval reduction algorithm provides a 2590-fold speedup. Furthermore, when combined with the pre-screening technique, we achieve a 3895-fold speedup.

4 Hidden Number Problem with Erroneous Input

In practical side-channel attacks, errors often appear in the data. This means that attackers may obtain incorrect nonces, resulting in erroneous HNP samples. Lattice-based attacks are believed to perform poorly when dealing with erroneous input [6,31]. A common strategy to tackle this issue is to run the HN-P solver on subsets of the samples until a correct solution is found [22]. However, this method does not fundamentally improve the lattice's ability to handle errors. Consequently, some works assume that the input is error-free [22,34,38]. In [3], Albrecht and Heninger discussed the solution for handling errors, but did not provide a detailed analysis. With the increase of error rate, the dimension of their lattice would increase rapidly. The restricted efficiency of their non-linear predicate would result in a high cost for searching the sieving database, thereby constraining the ability to handle erroreous HNP instances. On the other hand, Fourier analysis-based attacks demonstrated stronger robustness to errors [5,6,14], highlighting a gap between these two approaches.

In this section, we define HNP with erroneous input based on the ECDSA nonce leakage model. The effectiveness of our new lattice construction in solving this problem is demonstrated through theoretical analysis. An estimation for the minimum lattice dimension is also provided. Furthermore, we extend our algorithms in Section 3, and significantly enhance the lattice's ability to handle errors. This narrows the gap between lattice-based attacks and Fourier analysis-based attacks.

4.1 Theoretical Analysis

Definition 4 (Hidden Number Problem with Erroneous Input). *Given a modulus q, an error rate $0 < p < 1$, and both the hidden number α and the coefficients t_i being random numbers in $\mathbb{Z}_q$, the elements a_i satisfy the condition that with probability $1-p$, $|t_i\alpha - a_i|_q < q/2^l$, while with probability p, $|t_i\alpha - a_i|_q$ is a random number in $\mathbb{Z}_q$. The problem is to recover the hidden number α when m samples (t_i, a_i) are given.*

Let us show the rationality of this definition. In the ECDSA signature, let $k = 2^l k_{msb} + k_{lsb}$, where $0 \leq k_{msb} < q/2^l$, and $0 \leq k_{lsb} < 2^l$. Then we have

$$2^{-l}(k_{lsb} - s^{-1} \cdot h) + k_{msb} \equiv 2^{-l}s^{-1}r \cdot sk \mod q.$$

Assume that we have probability $1-p$ to obtain the correct value of k_{lsb}, and probability p to obtain a random integer k'_{lsb} in $[0, 2^l - 1]$.

Let $\alpha = sk, a_i = |2^{-l}(k_{lsb} - s^{-1} \cdot h(m))|_q$, $k_i = k_{msb}$, and $t_i = |2^{-l}s^{-1}r|_q$. If we obtain a random integer k'_{lsb}, then $a_i = |2^{-l}(k'_{lsb} - s^{-1} \cdot h)|_q$, and we have

$$|t_i\alpha - a_i|_q = |k_{msb} + 2^{-l}(k_{lsb} - k'_{lsb})|_q = |2^{-l}(k - k'_{lsb})|_q.$$

Since k is randomly chosen from $\mathbb{Z}_q$ and q is a prime number, $|2^{-l}(k - k'_{lsb})|_q$ is also a random number in $\mathbb{Z}_q$. Thus, we obtain a HNP instance with erroneous input.

New Minimum Lattice Dimension. In this section, the lattice constructed by us is the same as that in Section 3.1:

$$\begin{bmatrix} q & 0 & \cdots & 0 & 0 & 0 \\ 0 & q & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & q & 0 & 0 \\ x \cdot t'_1 & x \cdot t'_2 & \cdots & x \cdot t'_{m-1} & x & 0 \\ a'_1 & a'_2 & \cdots & a'_{m-1} & 0 & \tau \end{bmatrix}.$$

The target vector is $\boldsymbol{v} = \pm(k'_1 - \alpha_1 t'_1, \dots, k'_{m-1} - \alpha_1 t'_{m-1}, x\alpha_0, \tau)$. From the definition above, for $i = 0, \dots, m-1$, k_i is randomly distributed in $[0, q)$ with probability p. Thus, $x\alpha_0 \equiv k'_0 - \alpha_1 \equiv k_0 - w - \alpha_1$ and $k'_i - \alpha_1 t'_i \equiv k_i - w - \alpha_1 t'_i$ also have probability p of being randomly distributed in $[-q/2, q/2)$. So we have

$$\begin{aligned} \mathbb{E}\left[\|\boldsymbol{v}\|^2\right] &= (d-1)(1-p)\frac{w^2}{3} + (d-1)\frac{pq^2}{12} + \tau^2 \\ &= (d-1)\frac{w^2}{3}\left(1 + p(2^{2l}-1)\right) + \tau^2. \end{aligned}$$

Similar to the analysis in Section 3.1, when $\tau^2 = w^2(1 + p(2^{2l} - 1))/3$, the ratio $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] / \mathrm{GH}^2(\mathcal{L})$ obtains its minimum. Substituting this into $\mathbb{E}\left[\|\boldsymbol{v}\|^2\right] \leq 4\,\mathrm{GH}^2(\mathcal{L})/3$, we get

$$d \geq \frac{2l + 2 + 2\log q + \log 3 - 2\log x - \log\left(1 + p \cdot (2^{2l} - 1)\right)}{2l + 3 - \log(\pi e) - \log\left(1 + p \cdot (2^{2l} - 1)\right)}.$$

The reduction of dimension is $2\log x/(2l+3-\log(\pi e)-\log(1+p\cdot(2^{2l}-1)))$, which is more than $\log x/l$. This discovery reveals that our new lattice construction achieves a greater reduction (compared to Albrecht and Heningerąŕs lattice [3]) in lattice dimension in the presence of erroneous samples compared to error-free samples.

From a different perspective, the parameter x can be viewed as a balance to the error rate p. Given the lattice dimension d, p amplifies the target vector's squared magnitude by $1 + p(2^{2l} - 1)$, while x amplifies $\mathrm{GH}^2(\mathcal{L})$ by $x^{2/d}$. To maintain this ratio, we can set $1 + p(2^{2l} - 1) = x^{2/d}$, which leads to $x = (1 + p(2^{2l} - 1))^{d/2}$. Thus, for a higher error rate, we can increase x to keep the lattice dimension unchanged.

4.2 Modified Algorithms

Linear Predicate for Erroneous Input. For the hidden number candidate α', we compute $|t_i\alpha' - a_i|_q$ for each HNP sample (t_i, a_i). If this value falls within the range $[0, q/2^l)$, we say α' passes the test; otherwise, it fails. For error-free samples, if the candidate fails a single test, it can be regarded as incorrect. However, if there are errors in the samples, this judgment is not necessarily true.

Table 2 lists the passing probabilities for error-free and erroneous samples. We can see that regardless of whether the candidate is correct or not, there exists a possibility that it fails a test. We denote the probability of passing a single sample as p_1 when $\alpha' = \alpha$, and as p_2 when $\alpha' \neq \alpha$. Then we get $p_1 = 1-p+p\cdot 2^{-l}$, $p_2 = 2^{-l}$, and $p_1 > p_2$.

To extend the linear predicate in Section 3.2 to handle erroneous input, we count the number of samples that pass the test. Specifically, we collect $N = 2\log q$ samples and count the number of samples that α' passes, denoted as M. If M exceeds $N(p_1+p_2)/2$, we conclude that α' is the correct hidden number α. Otherwise, we discard it. This procedure is detailed in Algorithm 7.

Table 2: Passing probability for error-free and erroneous samples

	Error-free sample	Erroneous sample
$\alpha' = \alpha$	1	2^{-l}
$\alpha' \neq \alpha$	2^{-l}	2^{-l}

Algorithm 7: Predicate for erroneous input

```
Input: A 2-dimensional vector $\boldsymbol{v} = (v_0, v_1)$, modulus $q$, number of nonce
       leakage $l$, embedding number $\tau$, $N = 2\log q$ erroneous samples $(t_i, a_i)$
Output: The hidden number $\alpha$ or $\perp$
1  if $v_0 = 0$ or $v_0 > q/2^{l+1}$ or $|v_1| \neq \tau$ then
2      return 0;
3  $k_0 \leftarrow -\operatorname{sign}(v_1)v_0 + q/2^{l+1}$ ;
4  $\alpha' \leftarrow t_0^{-1}(a_0 + k_0) \bmod q$ ;
5  $M \leftarrow 0$;
6  for $i = 0$ to $N-1$ do
7      if $|t_i\alpha' - a_i|_q < q/2^l$ then
8          $M \leftarrow M + 1$;
9  if $M > N(1-p+(1+p)2^{-l})/2$ then
10     return $\alpha'$;
11 else
12     return $\perp$;
```

Theorem 5. *Algorithm 7 has an overwhelming success probability* $1-\operatorname{negl}(\log q)$.

Proof. Let P_1 be the probability that Algorithm 7 rejects a correct hidden number, and P_2 be the probability that Algorithm 7 accepts an incorrect candidate.

We prove both P_1 and P_2 are negligible. Define

$$X_i = \begin{cases} 1, & \text{if } \alpha' \text{ passes the } i\text{-th sample;} \\ 0, & \text{if } \alpha' \text{ fails the } i\text{-th sample.} \end{cases}$$

When $\alpha' = \alpha$, X_i follows the Bernoulli distribution with probability p_1, and when $\alpha' \neq \alpha$, X_i follows the Bernoulli distribution with probability p_2. Let $S_N = \sum_{i=1}^{N} X_i$.

For the case of P_1, let μ_1 be the expected value of S_N. Then $\mu_1 = p_1 N$. Let $\delta_1 = (p_1 - p_2)/(2p_1) \geq 1/(8p_1)$. By the Chernoff inequality, we have

$$P_1 = \Pr\left(S_N \leq \frac{N(p_1+p_2)}{2}\right) < e^{-\mu_1 \frac{\delta_1^2}{2}} \leq e^{-\frac{p_1 N}{128 p_1^2}} \leq e^{-\frac{\log q}{64}}.$$

For the case of P_2, let μ_2 be the expected value of S_N. Then $\mu_2 = p_2 N$. Let $\delta_2 = (p_1 - p_2)/(2p_2) = (1-p)(2^l - 1)/2 \geq 2^{l-3}$. By the Chernoff inequality, we have

$$P_2 = \Pr\left(S_N > \frac{N(p_1+p_2)}{2}\right) < e^{-\mu_2 \frac{\delta_2^2}{2}} \leq e^{-p_2 N \frac{2^{2l-6}}{2}} \leq e^{-2^{l-6}\log q} \leq e^{-\frac{\log q}{32}}.$$

□

Pre-screening Technique for Erroneous Input. The idea above can also be applied to the pre-screening technique. To achieve efficient pre-screening, we use $\log q$ samples that satisfy $|t_i'| < q/(2^{l+3}x)$. For each sample (t_i', a_i'), we compute $\left| |xt_i'\alpha_0 - a_i' + q/2|_q - q/2 \right|$. A sample (t_i', a_i') is called non-compliant if $\left| |xt_i'\alpha_0 - a_i' + q/2|_q - q/2 \right| > w + q/2^{l+4}$.

During the pre-screening, we cannot make any decision based only on a single non-compliant sample. Our strategy is to collect a set of samples, and make the decision when the number of non-compliant samples reaches a certain threshold. Different from Algorithm 7, the goal of pre-screening is to retain the correct hidden numbers, rather than to eliminate all incorrect candidates. Since the number of erroneous samples is at most $3p\log q$ with overwhelming probability, we discard the hidden number candidate if more than $3p\log q$ samples are non-compliant.

Sub-sampling Technique. In Section 3.3, we introduce an interval reduction algorithm for lattices constructed from the decomposition technique. This algorithm can efficiently perform an exhaustive search over an interval. However, it requires error-free samples. Otherwise, the algorithm may exclude the correct candidate. To overcome this limitation, we propose a sub-sampling technique. The specific steps are as follows.

(1) Select $3\log x/2$ samples to form a pool.
(2) Draw $\log x$ samples from this pool, and apply Algorithm 6 to the candidate α_0', but replace the linear predicate in Algorithm 6 with the predicate for erroneous input in Algorithm 7. Return upon success.

(3) Repeat the second step for γ times. If the hidden number is not found, the candidate is rejected.

Let P_1 be the probability that this algorithm rejects a correct hidden number candidate, and P_2 be the probability that this algorithm accepts an incorrect hidden number candidate. For any correct hidden number α, it has probability $p_1 = 1-p+p\cdot 2^{-l}$ to pass a single sample. Hence, the second step has probability $p_1^{\log x}$ to return α. Therefore, $P_1 = (1-p_1^{\log x})^{\gamma}$. For any incorrect hidden number α, if α is returned, α must be returned by Algorithm 7. By Theorem 5, we have $P_2 = \mathrm{negl}(\log q)$.

In practice, for given p, x and l, we can adjust γ to minimize P_1. For example, for the case of $p = 0.02, x = 2^{25}, l = 1$, let $\gamma = 10$, then we get $P_1 < 3 \times 10^{-7}$.

5 Key Recovery of ECDSA with Nonce Leakage

In this section, experimental results are provided to demonstrate the performance of our algorithms. Assume that the least significant bits of the nonce used for each signature are leaked. Our goal is to recover the secret signing key by solving the corresponding HNP instance.

Our implementation integrates the progressive sieving technique [25,15], which starts at a low sieving dimension and increases the sieving dimension by 1 with each iteration. The process of searching the database in Algorithm 3 is parallelized for improving the efficiency. Moreover, before applying Algorithm 3 to the lattice basis constructed from HNP samples, we preprocess the basis with BKZ-20, which can randomize the lattice basis and increase the success rate. In [3], Albrecht et al. used BKZ-(d-20) in their preprocessing step. However, the experimental results show that our preprocessing step is much more efficient and achieves a success rate close to [3].

To introduce the strategy of our attack on instances of varying difficulty, we categorize ECDSA instances into three classes. Specifically, this classification relies on the minimum lattice dimension d estimated via Albrecht and Heninger's lattice. These three classes are denoted as Easy ($d \leq 100$), Medium ($100 < d \leq 140$), and Hard ($d > 140$). In practical applications, we may adjust the parameter x. This allows us to obtain an optimal balance between time consumption and the number of available samples.

5.1 Compared with Other Lattice-based Attacks

Solving Easy Instances. Table 3a lists our attack results for several easy instances. These experiments are conducted using a single thread on an Intel Xeon Gold 6154 CPU with the G6K library [2]. For each instance, the average CPU-seconds and the success rate (s/r) are calculated from 100 experiments. Figure 2 illustrates the efficiency comparison between our algorithm and recent works [3,34,38]. It is evident that our algorithm exhibits a significant efficiency advantage. For example, when dealing with ECDSA$(384, 4)$, Albrecht and Heninger [3] successfully conducted an attack in 49200 seconds, and Xu et al. [38] improved the time to 11153 seconds, whereas we only require 5583 seconds. When targeting ECDSA$(192, 2)$, our attack also demonstrates a 31-fold speedup compared to the attack performed by Albrecht and Heninger [3].

When $x = 1$, the key difference between our work and other works [3,38] lies in the predicate used in the attack. We substitute our predicate with those in [3] and [38] separately, and record the time of searching the database on the same machine. We denote the time taken by our predicate as t_1, and the time taken by the predicates in [3] and [38] as t_2 and t_3, respectively. The speed-up ratio brought by our predicate is shown in Figure 3.

Table 3: Performance of our lattice-based attacks

Curve	Leakage	d	x	CPU-seconds	s/r	Previous records
secp160r1	2	82	1	206s	52%	259s in [38]
		77	2^{10}	71s	58%	
secp192r1	2	99	1	10360s	60%	87500s in [3]
		94	2^{10}	2829s	69%	
secp256r1	4	66	1	7s	65%	15s in [38]
		64	2^{10}	5s	79%	
	3	87	1	634s	53%	924s in [38]
		84	2^{10}	359s	57%	
secp384r1	4	98	1	8154s	62%	11153s in [38]
		96	2^{10}	5583s	56%	

(a) Easy instances

Curve	Leakage	d	x	Wall time	Mem GiB	Previous records
secp112r1	1	116	1	6min	35	260min in [38]
secp256r1	2	129	1	95min	219	466min in [38]
		124	2^{10}	31min	114	
secp384r1	3	130	1	128min	252	156min in [38]
		125	2^{15}	39min	132	

(b) Medium instances

The success rate of our attack can be further improved by increasing the lattice dimension. Taking ECDSA$(160, 2)$ as an example, Figure 4 illustrates how the success rate notably increases as the lattice dimension grows. We conduct experiments on 500 randomly generated instances. When the lattice dimension reaches 92, the success rate approaches 100%. This observation can be attributed

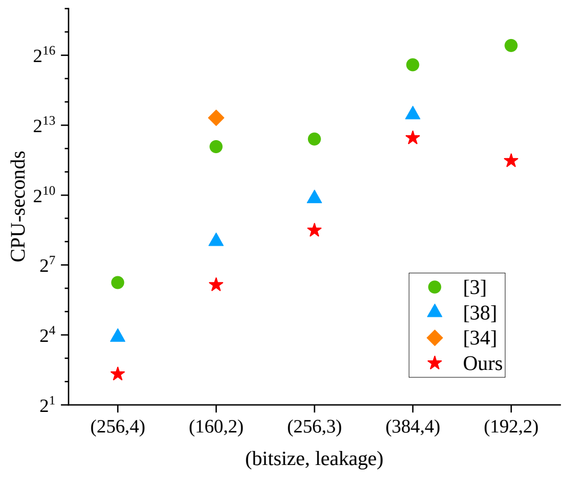


Fig. 2: Comparison of CPU-seconds with previous works.

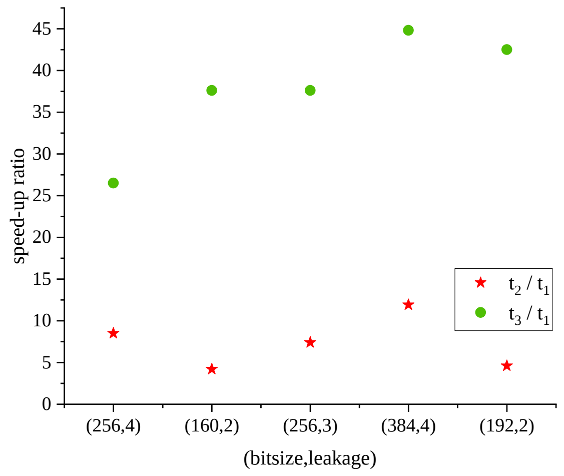


Fig. 3: Searching the sieving database using different predicates.

to the fact that the majority of the $\|\boldsymbol{v}\|/\,\mathrm{GH}$ ratios are below $\sqrt{4/3}$, which enables us to find the target vectors through sieving algorithms with high success rates. The experimental results also indicate that our algorithm can still handle

Table 4: New records of lattice-based attacks against ECDSA

Curve	Leakage	d	x	Samples	Wall time	Mem GiB
brainpoolp512r1	4	130	1	2^{10}	96min	254

(a) 4-bit leakage

Curve	Leakage	d	x	Samples	Wall time	Mem GiB
secp128r1	1	131	1	2^{8}	72min	294
		118	2^{15}	2^{26}	8min	53
secp160r1	1	144	2^{14}	2^{25}	824min	1939
		138	2^{25}	2^{36}	279min	850

(b) 1-bit leakage

Curve	Error rate	d	x	Samples	Wall time	Mem GiB
secp128r1	0.1	140	2^{20}	2^{31}	370min	1090
secp160r1	0.02	144	2^{14}	2^{25}	1009 min	1960

(c) Less than 1-bit leakage

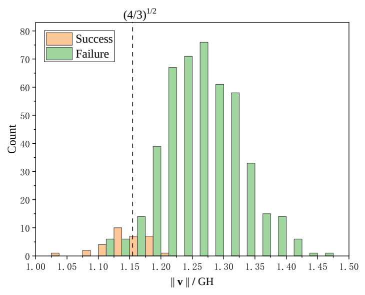
instances with $\|\boldsymbol{v}\|/\,\mathrm{GH}$ ratios greater than $\sqrt{4/3}$ with a lower success rate, rather than failing 100%.

Solving Medium Instances. Our attack results for medium instances are presented in Table 3b. These experiments are conducted on an Intel Xeon Platinum 8480+ CPU and two GeForce RTX 4090 GPUs. We employ G6K-GPU [16] to achieve high-performance sieving algorithms. Our attacks demonstrate high efficiency. Taking ECDSA(256,2) as an example, when x is set to 1, the attack is completed in 95 minutes by constructing a 129-dimensional lattice. The time can be significantly reduced by using a larger x. In our experiment, we set $x = 2^{10}$ and complete the attack in just 31 minutes. On the other hand, the attack in [38] required 466 minutes with four GeForce RTX 3090 GPUs.

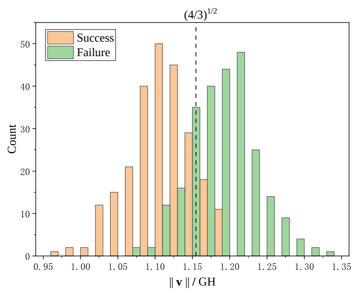
5.2 New Records of Lattice-based Attacks against ECDSA

We achieve several new records of lattice-based attacks against ECDSA. These attacks are conducted using an Intel Xeon Platinum 8480+ CPU and four GeForce RTX 4090 GPUs. Specific details about time and memory consumption can be found in Table 4.

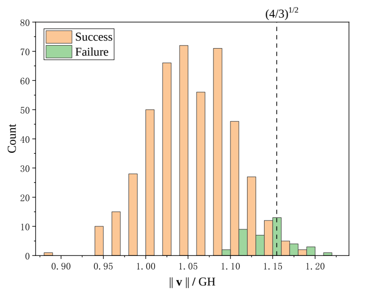
4-bit Leakage. The previous record for 4-bit leakage is only achieved on a 384-bit curve. As depicted in Table 4a, our algorithm is able to break ECDSA$(512, 4)$



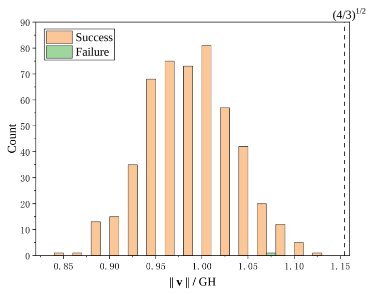
(a) $d = 77$



(b) $d = 82$



(c) $d = 87$



(d) $d = 92$

Fig. 4: Breaking ECDSA(160, 2) via lattices with different dimensions.

by a 130-dimensional lattice, which takes 96 minutes and consumes 254 GiB of memory.

1-bit Leakage. The first implementation of Fourier-based attack against ECDSA with 1-bit leakage was proposed by Aranha et al. at ASIACRYPT 2014 [5]. They achieved a full key recovery on a 160-bit elliptic curve using 2^{33} ECDSA signatures. However, breaking ECDSA with 1-bit leakage using lattice algorithms has been considered to be very difficult. In 2023, Xu et al. reported a successful key recovery for 1-bit leakage on a 112-bit curve [38]. Nonetheless, breaking ECDSA(160, 1) was regarded as exceptionally challenging by previous lattice approaches [5,6,3,34,38]. Xu et al. required a lattice dimension of approximately 165, which would make the time and space complexities of sieving algorithms unacceptable [38]. Besides, Sun et al. estimated the time complexity of their guessing bits algorithm is 2^{110} BKZ-30 operations [34].

We present the first implementation of lattice attacks against ECDSA with 1-bit leakage on both 160-bit and 128-bit curves. The experimental results are presented in Table 4b. When targeting ECDSA(160, 1), it is essential to reduce the sieving dimension to a manageable size for accepted time and memory use. We implement the attack with the parameter x set to 2^{25} and 2^{14}, corresponding to total sample sizes of 2^{36} and 2^{25}, respectively. When x is set to 2^{25}, we construct a 138-dimensional lattice to perform the attack, which takes approximately 279 minutes and consumes 850 GiB of memory.

Less than 1-bit Leakage. Before this work, only Fourier analysis-based attacks [6] could break ECDSA with less than 1-bit nonce leakage. We provide the first lattice-based attack results in Table 4c. Our attacks successfully handle an ECDSA instance with an error rate of 0.1 on a 128-bit curve and an ECDSA instance with an error rate of 0.02 on a 160-bit curve.

Acknowledgements. We would like to thank the anonymous reviewers of ASIACRYPT 2024, EUROCRYPT 2024 and CRYPTO 2024 for their insightful suggestions. We also thank Fan Huang, Xiaolin Duan, Yaqi Wang, and Changhong Xu for their valuable support to this work. This work was supported by National Natural Science Foundation of China (Grant No. 62472397) and Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0302902).

References

1. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: 33rd ACM STOC. pp. 601-610. (2021). https://doi.org/10.1145/380752.380857
2. Albrecht, M.R., Ducas, L., Herold, G., Kirshanova, E., Postlethwaite, E.W., Stevens, M.: The general sieve kernel and new records in lattice reduction. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11477, pp. 717-746. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17656-3_25
3. Albrecht, M.R., Heninger, N.: On bounded distance decoding with predicate: Breaking the "lattice barrier" for the hidden number problem. In: Canteaut, A., Standaert, FX. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 528-558. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_19
4. Albrecht, M.R., Heninger, N.: Bounded distance decoding with predicate source code (2020). https://github.com/malb/bdd-predicate
5. Aranha, D.F., Fouque, PA., Gérard, B., Kammerer, JG., Tibouchi, M., Zapalowicz, JC.: GLV/GLS decomposition, power analysis, and attacks on ECDSA signatures with single-bit nonce bias. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 262-281. Springer, Berlin, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_14
6. Aranha, D.F., Novaes, F.R., Takahashi, A., Tibouchi, M., Yarom, Y.: LadderLeak: Breaking ECDSA with less than one bit of nonce leakage. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020, pp. 225-242. ACM Press (2020). https://doi.org/10.1145/3372297.3417268
7. Babai, L.: On Lovász lattice reduction and the nearest lattice point problem. Combinatorica 6, 1-13 (1986). https://doi.org/10.1007/BF02579403
8. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Krauthgamer, R. (ed.) 27th SODA, pp. 10-24. ACM-SIAM (2016). https://doi.org/10.1137/1.9781611974331.ch2

9. Becker, A., Gama, N., Joux, A.: Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest neighbor search. Cryptology ePrint Archive, Report 2015/522 (2015). http://eprint.iacr.org/2015/522
10. Bleichenbacher, D.: On the generation of one-time keys in DL signature schemes. Presentation at IEEE P1363 Working Group Meeting (2000).
11. Bleichenbacher, D.: Experiments with DSA. Rump session at CRYPTO (2005). https://www.iacr.org/conferences/crypto2005/r/3.pdf.
12. Boneh, D., Venkatesan, R.: Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In: Koblitz, N. (ed.) CRYPTO 96. LNCS, vol. 1109, pp. 129-142. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_11
13. Breitner, J., Heninger, N.: Biased nonce sense: Lattice attacks against weak ECDSA signatures in cryptocurrencies. In: Goldberg, I., Moore, T. (eds.) FC 2019. LNCS, vol. 11598, pp. 3-20. Springer, Heidelberg (2019). https://doi.org/10.1007/978-3-030-32101-7_1
14. De Mulder, E., Hutter, M., Marson, M.E., Pearson, P.: Using Bleichenbacher's solution to the hidden number problem to attack nonce leaks in 384-bit ECDSA. In: Bertoni, G., Coron, J.S. (eds.) CHES 2013. LNCS, vol. 8086, pp. 435-452. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40349-1_25
15. Ducas, L.: Shortest vector from lattice sieving: A few dimensions for free. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10820, pp. 125-145. Springer, Heidelberg (2018). https://doi.org/10.1007/978-3-319-78381-9_5
16. Ducas, L., Stevens, M., van Woerden, W.: Advanced lattice sieving on GPUs, with tensor cores. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 249-279. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_9
17. Fincke, U., Pohst, M.: Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Mathematics of Computation 44(170), 463-471 (1985).
18. Fitzpatrick, R., Bischof, C., Buchmann, J., Dagdelen, Ö., Göpfert, F., Mariano, A., Yang, B.-Y.: Tuning GaussSieve for speed. In: LATINCRYPT 2014. LCNS, vol. 8895, pp. 288-305. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-16295-9_16
19. Gama, N., Nguyen, P.Q., Regev, O.: Lattice enumeration using extreme pruning. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 257-278. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_13
20. Heninger, N.: Using Lattices for Cryptanalysis. (2020). https://simons.berkeley.edu/-talks/using-lattices-cryptanalysis
21. Herold, G., Kirshanova, E., Laarhoven, T. : Speed-ups and time-memory tradeoffs for tuple lattice sieving. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol 10769, pp.407-436. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_14
22. Jancar, J., Sedlacek, V., Svenda, P., Sys, M.: Minerva: The curse of ECDSA nonces. IACR TCHES 2020(4), 281-308 (2020). https://doi.org/10.13154/tches.v2020.i4.281-308
23. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. 12(3), 415-440 (1987). https://doi.org/10.1287/moor.12.3.415
24. Laarhoven, T.: Sieving for shortest vectors in lattices using angular locality-sensitive hashing. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 3-22. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_1

25. Laarhoven, T., Mariano, A.: Progressive lattice sieving. In: Lange, T., Steinwandt, R. (eds.) PQCrypto 2018. LNCS, vol. 10786, pp. 292-311. Springer, Heidelberg (2018). https://doi.org/10.1007/978-3-319-79063-3_14
26. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261, 366-389 (1982). https://infoscience.epfl.ch/record/164484/files/nscan4.PDF
27. Micciancio, D., Voulgaris, P.: Faster exponential time algorithms for the shortest vector problem. In: Charika, M. (ed.) 21st SODA. pp. 1468-1480. ACM-SIAM (2010). https://doi.org/10.1137/1.9781611973075.119
28. Moghimi, D., Sunar, B., Eisenbarth, T., Heninger, N.: TPM-FAIL: TPM meets timing and lattice attacks. In: Capkun, S., Roesner, F. (eds.): USENIX Security 2020. pp. 2057-2073. (2020). https://www.usenix.org/system/files/sec20-moghimi-tpm.pdf
29. Nguyen, P.Q., Shparlinski, I.: The insecurity of the digital signature algorithm with partially known nonces. Journal of Cryptology 15(3), 151-176 (2002). https://doi.org/10.1007/s00145-002-0021-3
30. Nguyen, P.Q., Vidick, T.: Sieve algorithms for the shortest vector problem are practical. J. of Mathematical Cryptology 2(2), 181-207 (2008). https://doi.org/10.1515/JMC.2008.009
31. Ryan, K.: Return of the hidden number problem. IACR TCHES 2019(1), 146-168 (2018). https://tches.iacr.org/index.php/TCHES/article/view/7337
32. Schnorr, C.P. : Lattice reduction by random sampling and birthday methods. In: Alt, H., Habib, M. (eds.) STACS 2003. LNCS, vol. 2607, pp. 145-156. Springer, Berlin, Heidelberg (2003). https://doi.org/10.1007/3-540-36494-3_14
33. Schnorr, C., Euchner, M.: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. Math. Program. 66, 181-199 (1994). https://doi.org/10.1007/BF01581144
34. Sun, C., Espitau, T., Tibouchi, M., Abe, M.: Guessing bits: Improved lattice attacks on (EC)DSA with nonce leakage. IACR TCHES 2022(1), 391-413 (2022). https://tches.iacr.org/index.php/TCHES/article/view/9302
35. Takahashi, A., Tibouchi, M., Abe, M.: New Bleichenbacher records: Fault attacks on qDSA signatures. IACR TCHES 2018(3), 331-371 (2018). https://tches.iacr.org/index.php/TCHES/article/view/7278
36. The G6K development team: G6K (2020). https://github.com/fplll/g6k
37. The G6k-GPU-Tensor development team: G6k-GPU-Tensor (2021). https://github.com/WvanWoerden/G6K-GPU-Tensor
38. Xu, L., Dai, Z., Wu, B., Lin, D.: Improved attacks on (EC)DSA with nonce leakage by lattice sieving with predicate. IACR TCHES 2023(2), 568-586 (2023). https://doi.org/10.46586/tches.v2023.i2.568-586