Efficient Error-tolerant Side-channel Attacks on GPV Signatures Based on Ordinary Least Squares Regression

Jaesang Noh, Hyunseo Choi, Dongwoo Han, and Dong-Joon Shin

Department of Electronic Engineering, Hanyang University, Seoul, Korea
{darkelzm, hyun123456a, hdw0131, djshin}@hanyang.ac.kr

Abstract. The Gentry-Peikert-Vaikuntanathan (GPV) framework is utilized for constructing practical digital signatures, which is proven to be secure in both the classical/quantum random-oracle models. Falcon is such a signature scheme, recognized as a compact and efficient signature among NIST-standardized signature schemes. Recently, Guerreau et al. (CHES 2022) and Zhang et al. (Eurocrypt 2023) proposed the secret key recovery attack on Falcon utilizing signatures filtered by simple power analysis (SPA) attacks. However, these attacks, which exploit the conditional signature distributions, require a large number of SPA attacks to obtain the filtered signatures. Furthermore, no existing attack considers general GPV signatures despite the importance of the GPV framework in modern digital signatures. Therefore, we address these problems as follows.

First, we introduce, for the first time, a concept of vulnerable partial information of GPV signatures and propose a non-filtering secret key recovery attack, called OLS attack, which effectively utilizes partial information without filtering. The proposed OLS attack is a linear estimator with computational complexity that scales linearly with the number of samples, making OLS attack highly practical. Furthermore, we prove that the secret key recovered by the OLS attack converges to the real secret key in probability as the number of samples increases.

Second, we leverage SPA to extract Gaussian leakage, which is used as partial information for the OLS attack on Falcon. As a result, the OLS attack shows a significantly higher success rate with the fewest samples than the state-of-the-art attacks. Furthermore, by incorporating the DDGR attack, the OLS attack can recover the secret key using much fewer samples with a success rate close to 100%. Moreover, we propose an OLS attack specialized for Falcon, which can even more reduce the number of required side-channel attacks.

Third, we propose an error-tolerant power analysis attack using MAP decoding, which effectively corrects the errors in samples to properly estimate Gaussian leakage. For concrete experimental validation, an ELMO simulator generates noisy power traces and ChipWhisperer measures power traces from the STM32F415 model. The proposed MAP decoding achieves high effectiveness for estimating Gaussian leakage, particularly when applied to power traces collected using low-resolution ChipWhisperer. In conclusion, it is important for future work to study countermeasures for OLS attacks.

1 Introduction

A rapid advancement of quantum computing has posed a significant threat to traditional cryptosystems. In particular, Shor's algorithm can break RSA cryptosystems in polynomial time [32], and Grover's search algorithm reduces the security bit size of symmetric key cryptosystems by half [14]. To address these quantum threats, the national institute of standards and technology (NIST) initiated a competition to identify post-quantum cryptographic (PQC) algorithms that are resilient against quantum attacks. After three rounds of evaluation, NIST finally selected Falcon [30], CRYSTALS-Dilithium [20], and SPHINCS+ [2] as quantum-resistant signature schemes.

Falcon and Dilithium are constructed based on different cryptographic frameworks: Falcon is built on the GPV framework [12] and Dilithium is based on Fiat-Shamir with abort. Notably, the GPV framework and Fiat-Shamir with abort are widely regarded as the most powerful frameworks for constructing secure and practical PQC signatures. The security of the GPV framework is based on the hardness of the short integer solution (SIS) problem, and its security is proven even in both classical and quantum random-oracle models [4]. By appropriately selecting lattices and trapdoor samplers, the GPV framework can be used to construct practical signatures with strong PQC security. Therefore, several signature schemes, including Falcon, Mitaka [8] and HuFu [36], have been developed using the GPV framework. However, despite theoretical security guarantees, GPV signatures may have vulnerabilities in physical implementations, and for this reason, research on the resistance of GPV signatures against side-channel attacks becomes increasingly critical.

In the PQC standardization process, NIST has been evaluating algorithms in terms of not only their resistance to quantum attack but also their security against side-channel attacks in real situations. Moreover, in the status report on the third round of NIST PQC standardization, NIST emphasized to future engineers and researchers the importance of side-channel analysis of PQC and expressed hope that such research would continue [1]. Therefore, side-channel attack is a very important issue in PQC algorithms and the NIST standard signature schemes have made significant progress in addressing side-channel attacks, gradually strengthening their defenses. Although additional computations are needed in Dilithium, its signatures become resilient to the t-probing model by employing a masking scheme [24]. Falcon adopts a constant-time Gaussian sampler [31] in its reference implementation to resist timing attacks [10].

Despite such protections, a side-channel attack through floating-point operations can fully recover the secret key of Falcon-512 using 10,000 electromagnetic traces measured by PicoScope 3206D oscilloscope [18]. Although the subsequent work [15] through floating-point operations can recover the secret key using 5,000 power traces measured by ChipWhisperer, these correlation-based attacks must compute the correlation across all measured tracepoints for at least 2^{20} search space. Consequently, these attack schemes require a considerable amount of computation time. More recently, variants of the learning parallelepiped scheme [25], known as the hidden parallelepiped [15] and learning slice [37] schemes, can recover the secret key of Falcon by leveraging the conditional signature distribution. Specifically, these attack schemes recover the secret key by utilizing samples with signatures conditioned on specific Gaussian leakage, which is estimated by simple power analysis (SPA) attack. Some initial research results of the hidden parallelepiped scheme showed that the secret key of Falcon-512 could be recovered using 1 million samples, while the learning slice scheme demonstrated that a direct recovery of the secret key is possible using just 70,000 samples with a 25% success rate. If the learning slice scheme is combined with the exhaustive search, it can recover the secret key using 45,000 samples with a 25% success rate.

However, these secret key attack schemes, which exploit the conditional signature distribution, inevitably require filtered (or processed) signatures to recover the secret basis of GPV signature including Falcon. This requirement arises because the distribution of GPV signatures does not reveal any information about the secret basis [12]. Thus, obtaining filtered signatures requires a substantial number of samples. For example, in the learning slice scheme, 161,000 signatures are typically required to obtain 45,000 filtered signatures. To our knowledge, no attack has been proposed, which can leverage all the available signatures to maximize information entropy for recovering the secret basis. Moreover, no general attack scheme on GPV signatures has been proposed, despite the importance of GPV framework in PQC. Therefore, in this paper, we identify specific vulnerable partial information in the trapdoor sampler of the GPV framework and propose an efficient attack scheme on GPV signatures, which utilizes all the available signatures without filtering. To demonstrate the generality and efficiency of the proposed attack, Falcon is taken as a concrete example. The main contributions of this work are summarized as follows.

1.1 Our Contributions

Non-filitering Secret Key Recovery Attack on GPV Signatures. We establish, for the first time, a new concept of vulnerable partial information of GPV signatures and propose a general secret key recovery attack, called ordinary least squares (OLS) attack, based on simple OLS regression. Unlike the existing attack schemes exploiting selected samples following conditional signature distribution, the OLS attack efficiently recovers the secret key of GPV signatures by exploiting all the available signatures and partial information without filtering. Since the OLS attack is a linear estimator, it is straightforward to implement and has a computational complexity of $O(mn^2 + n^3)$, meaning that its complexity linearly increases with the number of samples m for a given dimension of the signature n. We also theoretically prove its consistency. i.e., the secret key recovered by the OLS attack converges to the real secret key in probability.

Efficient and Fast OLS Attack on Falcon. The OLS attack recovers the secret key of high-security variant of Falcon (Falcon-256, -512, -1024) using Gaussian leakage as partial information, which is the most effective for recovering the secret key of Falcon compared to the state-of-the-art attacks. For Falcon-512, the OLS attack directly recovers the secret key using only 35,000 samples with a 60% success rate. By combining it with the DDGR attack in [22], the secret key is recovered using 25,000 samples with a high success rate close to 100%. Since the OLS attack is based on simple linear regression, the secret key of Falcon-512 is directly recovered in about 8 seconds without using equipment such as high-performance CPUs. Furthermore, we develop an efficient OLS attack [1] on Falcon by leveraging the orthogonal basis properties of NTRU lattice, thereby reducing the required number of side-channel attacks by a factor of $1/2n$ for $n = 256, 512, 1024$.

Error-tolerant Power Analysis Attacks on Half-Gaussian Sampler. The SPA attack on the BaseSampler (half-Gaussian sampler) of Falcon is highly inefficient if the power trace is contaminated by the ever-present noise. Therefore, we propose an error-tolerant power analysis attack using maximum a posteriori (MAP) decoding, which corrects the errors in samples due to noise. To evaluate the performance of MAP decoding under various conditions, noisy power traces are generated using the ELMO simulator, and actual hardware power traces from STM32F415 are measured using ChipWhisperer. The proposed MAP decoding is demonstrated to be effective for estimating Gaussian leakage from noisy power traces collected using low-resolution ChipWhisperer. The OLS attack combined with MAP decoding further reduces the errors of the recovered secret key. Specifically, for ChipWhisperer, the combined scheme can recover the secret key of Falcon-512 using 47,000 samples. In contrast, without MAP decoding, 87,000 samples are required. Furthermore, compared to the OLS attack without MAP decoding, the combined scheme reduces about 50 errors in 30,000 samples at SNR 30 dB for Falcon-512. Therefore, the combined scheme can efficiently recover the secret key with fewer samples and faster than the OLS attack without MAP decoding.

1.2 Technical Overview

Proposing OLS Attack on General GPV Signatures. In the GPV signature, the trapdoor sampler takes the secret basis $\mathbf{B}$ as input and generates the signature $\mathbf{s}$ by modifying the coefficient vector $\mathbf{w}$. Specifically, the signature is computed as $\mathbf{s} = \mathbf{wA} = (\mathbf{r} - \mathbf{u})\mathbf{A}$, where $\mathbf{A}$ represents either $\mathbf{B}$ or its Gram-Schmidt orthogonal (GSO) matrix $\tilde{\mathbf{B}}$, determined by the trapdoor sampler. The vector $\mathbf{r}$ is typically derived from the message and $\mathbf{u}$ is a random vector generated by the trapdoor sampler. The distribution of the GPV signature has zero mean and a variance that does not reveal information about the secret basis, ensuring that the secret basis remains secure. To satisfy the zero mean property, the conditional expectation of $\mathbf{u}$ given $\mathbf{r}$ is set to $\mathbf{r}$. This conditional expectation property makes $\mathbf{u}$ a vulnerable partial information of the GPV signature.

[1] The OLS attack implementation can be found in the following GitHub repository: `https://github.com/JaesangNoh-crypto/Falcon_recovery_OLSattack`.

Suppose the partial information $\mathbf{u}$ is estimated by side-channel attack. As in Eq. (5), the dependent output vector $\mathbf{y}$ and the independent input vector $\mathbf{x}$ are constructed from $\mathbf{u}$ and $\mathbf{s}$, respectively. Since $\mathbf{r}$ is an unknown random vector due to the random message, $\mathbf{r}$ can be regarded as an error vector $\mathbf{e}$. Consequently, $\mathbf{s} = (\mathbf{r} - \mathbf{u})\mathbf{A}$ can be transformed into a multiple linear regression (MLR) model $\mathbf{y} = \mathbf{x}\mathbf{A}^{-1} + \mathbf{e}$. A key idea of this paper, proven in Lemma 1, is that $\mathbf{x}$ and $\mathbf{e}$ are uncorrelated if $\mathbf{u}$ satisfies the conditional expectation property. Exploiting this uncorrelated property, we demonstrate that the proposed OLS estimator for $\mathbf{A}^{-1}$ in Eq. (7) converges in probability. Furthermore, by applying the continuous mapping theorem, Theorem 4 demonstrates that the inverse transformation of the OLS estimator in Eq. (8) also converges to $\mathbf{A}$ in probability. The inverse transformation of the OLS estimator, referred to as the OLS attack, has a computational complexity of $O(mn^2+n^3)$ for the number of samples m and signature dimension n. Since $\mathbf{A}$ serves as the secret key of the GPV signature, the OLS attack can successfully recover the secret key given sufficient samples, and its computational time scales linearly with m. Therefore, unlike conventional secret key recovery schemes exploiting conditional signature distribution, the OLS attack utilizes all collected partial information and signatures without filtering and can be applied to any GPV signature.

Implementing OLS Attack on Falcon. The FFO sampler, a trapdoor sampler of Falcon, is functionally equivalent to the Klein-GPV algorithm. Consequently, Lemma 4.4 in [12], which introduced the Klein-GPV algorithm, is used in Corollary 1 to demonstrate that Gaussian leakage can serve as partial information of Falcon. By leveraging Gaussian leakage as partial information, the OLS attack can recover the GSO matrix $\tilde{\mathbf{B}}$ of Falcon secret basis $\mathbf{B}$. Since the first row of $\tilde{\mathbf{B}}$ contains all the coefficients of the NTRU lattice secret polynomials, recovering $\tilde{\mathbf{B}}$ can fully reconstruct $\mathbf{B}$ of Falcon. However, the coefficients of the estimated first row are real numbers requiring an additional step to approximate the coefficients as integers. Two main approaches for approximation are rounding and lattice reduction. When using the rounding with 35,000 samples, the secret key is recovered with 60 % success rate. Using the DDGR scheme [5] employing lattice reduction, the first row can be recovered with nearly 100% using 25,000 samples. Notably, 25,000 samples required for the OLS attack are significantly fewer than 161,000 samples required for the state-of-the-art attack.

Improving the Efficient of OLS Attack on Falcon. To recover the first row of $\mathbf{B}$, it is unnecessary to estimate the Gaussian leakage for every coefficient of the signature. The Gaussian leakage corresponding to the first coefficient is sufficient to recover the first rows. A simple OLS estimator in Eq. (11) is constructed to estimate the first column of $\tilde{\mathbf{B}}^{-1}$. Notably, the transpose of the first column of $\tilde{\mathbf{B}}^{-1}$ and the first row of $\tilde{\mathbf{B}}$ are scalar multiples of each other and their inner product equals 1. Leveraging this property, the estimator in Eq. (12) recovers the first row of $\tilde{\mathbf{B}}$ using the estimated first column of $\tilde{\mathbf{B}}^{-1}$. As a result, the required number of side-channel attacks per signature is reduced from $2n$ to 1.

Moreover, Theorem 5 proves that the estimator in Eq. (12) also converges to the first row of $\mathbf{B}$ in probability, and the computational complexity of the estimator is $O(mn + n^3)$.

Proposing Error-tolerant Power Analysis Attacks Using MAP Decoding. To estimate Gaussian leakage, the output of BaseSampler and the corresponding sign value must be estimated by SPA. The BaseSampler operates 18 for-loop iterations to generate an output ranging from 0 to 18. Each iteration exhibits an underflow feature in the power trace, which indicates whether 1 is added to the output or not. By detecting underflow features, SPA can identify which iteration involves adding 1 or 0, thereby allowing the recovery of the BaseSampler output by summing across all iterations. However, noise in the power trace may cause wrong binary decisions in SPA, allowing even a single misclassification in any of the 18 iterations to result in an incorrect estimation of the output.

A key observation is that for any given output of the BaseSampler, 1 is added during the iterations up to the output value, and 0 is added in the remaining iterations. For example, if the output is 3, then 1 is added in the first three for-loop iterations, followed by 0 in the remaining iterations. This sequence of 18 additions can be interpreted as a length-18 binary codeword, allowing an error-correcting scheme through MAP decoding based on the prior probability of SPA errors. This MAP decoding effectively corrects errors in SPA, enabling accurate estimation of Gaussian leakage even when certain for-loop iterations are incorrectly identified.

2 Preliminaries

2.1 Notations

This section introduces the conventional notations used in the paper *except when specified otherwise.*

Linear Algebra. Vectors and matrices are denoted in bold lowercase and uppercase letters, respectively. Vectors are considered row vectors. $\langle \mathbf{u}, \mathbf{v} \rangle$ denotes the inner product of vectors $\mathbf{u}$ and $\mathbf{v}$, and if $\langle \mathbf{u}, \mathbf{v} \rangle = 0$, $\mathbf{u}$ and $\mathbf{v}$ are called orthogonal. $\|\mathbf{v}\|$ denotes the Euclidean norm (L_2 norm) of $\mathbf{v}$, and $\|\mathbf{v}\|_1$ denotes the Manhattan norm (L_1 norm) of $\mathbf{v}$. $\mathbf{B}^T$ represents the transpose of $\mathbf{B}$, and $\mathbf{B}^*$ represents a Hermitian matrix of $\mathbf{B}$, i.e., $\mathbf{B}^*$ is the conjugate transpose of $\mathbf{B}$.

Distribution. For a distribution $Q_{\sigma,\lambda}$, σ is the standard deviation and λ is the expectation. The notation $x \sim Q_{\sigma,\lambda}$ denotes that a random variable x is distributed according to $Q_{\sigma,\lambda}$, and the probability density function (PDF) of x is denoted by $Q_{\sigma,\lambda}(x)$. $x \leftarrow Q_{\sigma,\lambda}$ implies that x is independently sampled from the distribution $Q_{\sigma,\lambda}$. For any set S, $x \leftarrow S$ represents that x is independently and randomly sampled from S. Given an algorithm or a real-valued function $\mathcal{A}$, $x \leftarrow \mathcal{A}(y)$ represents that x is an output of $\mathcal{A}$ on the input y.

Statistical Analysis. For a random variable x, $\mathbb{E}_x[x]$ and $Var[x]$ denote the expected value and variance of x, respectively. A random vector $\mathbf{x} = (x_1, x_2, \dots, x_n)$ is a multivariate random variable where x_i's are random variables. A random matrix $\mathbf{X}$ is also a multivariate random variable whose elements are random variables. In the paper, all random variables in a multivariate random variable are considered to be identically distributed. Given real random vectors $\mathbf{x}$ and $\mathbf{y}$ having finite variance and expected values, the covariance matrix $\mathbf{K_{xx}}$ is defined as

$$\mathbf{K_{xx}} := Cov[\mathbf{x}, \mathbf{x}] = \mathbb{E}_{\mathbf{x}}[(\mathbf{x} - \mathbb{E}_{\mathbf{x}}[\mathbf{x}])^T (\mathbf{x} - \mathbb{E}_{\mathbf{x}}[\mathbf{x}])],$$

and the cross-covariance matrix $\mathbf{K_{xy}}$ is defined as

$$\mathbf{K_{xy}} := Cov[\mathbf{x}, \mathbf{y}] = \mathbb{E}_{\mathbf{x},\mathbf{y}}[(\mathbf{x} - \mathbb{E}_{\mathbf{x}}[\mathbf{x}])^T (\mathbf{y} - \mathbb{E}_{\mathbf{y}}[\mathbf{y}])].$$

2.2 Convergence in Probability and Related Results

Convergence in Probability [11]. Let (X, d_X) be the metric space where X is a set and d_X is a metric on X. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be the probability space where Ω is a sample space, $\mathcal{F}$ is a σ-algebra on Ω, and $\mathbb{P}$ is a probability measure. Given a sequence of random variables $\{x_n\}$ on $(\Omega, \mathcal{F}, \mathbb{P})$ such that $x_n : \Omega \to X$ for all $n \in \mathbb{Z}^+ = \{1, 2, \dots\}$, then the sequence $\{x_n\}$ is said to converge to a random variable $x : \Omega \to X$ in probability if for every $\varepsilon > 0$,

$$\lim_{n \to \infty} \mathbb{P}(d_X(x, x_n) > \varepsilon) = 0.$$

The convergence in probability is denoted by using the probability limit operator 'plim' such that $\operatorname{plim}_{n \to \infty} x_n = x$.

Let $(V, \|\cdot\|_V)$ be a normed space where V is a vector space and $\|\cdot\|_V$ is a norm on V. Let $\{\mathbf{x}_n = (x_{n,1}, x_{n,2}, \dots, x_{n,m})\}$ be the sequence of multivariate random variables such that $\mathbf{x}_n : \Omega^m \to V$ for $n \in \mathbb{Z}^+$ where all coordinates $x_{n,i}$ of $\mathbf{x}_n$ are random variables on the same probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The convergence in probability of the sequence $\{\mathbf{x}_n\}$ to a multivariate random variable $\mathbf{x} : \Omega^m \to V$ is defined as follows. For every $\varepsilon > 0$,

$$\lim_{n \to \infty} \mathbb{P}(\|\mathbf{x} - \mathbf{x}_n\|_V > \varepsilon) = 0,$$

which can be written by $\operatorname{plim}_{n \to \infty} \mathbf{x}_n = \mathbf{x}$.

Convergence Theorems. Three theorems related to the convergence in probability are introduced, which are used in the proofs of Theorems 4 and 5. The first theorem is the weak law of large numbers (WLLN).

Theorem 1 (Weak Law of Large Numbers [9]). *Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n$ be independent and identically distributed (IID) multivariate random variables with $\mathbb{E}_{\boldsymbol{x}_i}[\boldsymbol{x}_i] = \boldsymbol{c} < \infty$ for all $i \in \{1, 2, \dots, n\}$. If $\boldsymbol{s}_n = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i$ then*

$$\operatorname*{plim}_{n \to \infty} \boldsymbol{s}_n = \boldsymbol{c}.$$

The following Slutsky's theorem shows the convergence of some simple operational results of two random matrices.

Theorem 2 (Slutsky's Theorem [9]). *Let $\{\boldsymbol{X}_n\}$ and $\{\boldsymbol{Y}_n\}$ be sequences of random matrices. If $\operatorname{plim}_{n\to\infty} \boldsymbol{X}_n = \boldsymbol{X}$ and $\operatorname{plim}_{n\to\infty} \boldsymbol{Y}_n = \boldsymbol{C}$ for a random matrix $\boldsymbol{X}$ and a constant matrix $\boldsymbol{C}$, then*

$$(a)\ \operatorname*{plim}_{n\to\infty}(\boldsymbol{X}_n + \boldsymbol{Y}_n) = \boldsymbol{X} + \boldsymbol{C}.$$

$$(b)\ \operatorname*{plim}_{n\to\infty}(\boldsymbol{X}_n \boldsymbol{Y}_n) = \boldsymbol{X}\boldsymbol{C},\ \textit{if } \boldsymbol{X}_n \boldsymbol{Y}_n \textit{ is defined.}$$

$$(c)\ \operatorname*{plim}_{n\to\infty}(\boldsymbol{X}_n \boldsymbol{Y}_n^{-1}) = \boldsymbol{X}\boldsymbol{C}^{-1},\ \textit{if } \boldsymbol{X}_n \boldsymbol{Y}_n^{-1} \textit{ is defined and } \boldsymbol{C} \textit{ is an invertible matrix.}$$

The final theorem is the continuous mapping theorem, which concerns the convergence of functions at continuous points.

Theorem 3 (Continuous Mapping Theorem [3]). *Let $(X, \|\cdot\|_X)$ and $(Y, \|\cdot\|_Y)$ be normed spaces, and let f be a measurable function from X to Y. Let D_f denote the discontinuity set of f. Given a sequence of multivariate random variables $\{\boldsymbol{x}_n\}$ defined on the metric space X, if $\operatorname{plim}_{n\to\infty} \boldsymbol{x}_n = \boldsymbol{x}$ for a multivariate random variable $\boldsymbol{x}$ and $\mathbb{P}(\boldsymbol{x} \in D_f) = 0$, then*

$$\operatorname*{plim}_{n\to\infty} f(\boldsymbol{x}_n) = f\left(\operatorname*{plim}_{n\to\infty} \boldsymbol{x}_n\right) = f(\boldsymbol{x}).$$

2.3 The Gentry-Peikert-Vaikuntanathan (GPV) Framework

Lattice. A basis $\mathbf{B}$ of a lattice $\mathcal{L}$ is a set $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\}$ of linearly independent vectors $\mathbf{b}_i \in \mathbb{R}^m$ for $i \in \{1, 2, \ldots, n\}$, which also generates the lattice $\mathcal{L}$ as

$$\mathcal{L} = \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i \mid x_i \in \mathbb{Z} \right\}.$$

Also, $\mathbf{B}$ can be expressed as an $n \times m$ full-rank matrix where $\mathbf{b}_i$ is located as the ith row vector. A lattice generated by a basis $\mathbf{B}$ is a discrete additive subgroup of $\mathbb{R}^m$, which is denoted by $\mathcal{L}(\mathbf{B})$, and n is the dimension of $\mathcal{L}(\mathbf{B})$.

A q-ary lattice, which satisfies $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$ for some integer $q \in \mathbb{Z}^+$, is primarily used in the construction of lattice-based cryptography. For any full-rank matrix $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$, the q-ary lattice generated by $\mathbf{B}$ is defined as

$$\Lambda_q(\mathbf{B}) := \{\mathbf{y} \in \mathbb{Z}^m \mid \mathbf{y} = \mathbf{z}\mathbf{B} \bmod q \text{ for } \mathbf{z} \in \mathbb{Z}^n\}.$$

The orthogonal lattice of $\Lambda_q(\mathbf{B})$ modulo q, denoted as $\Lambda_q^{\perp}(\mathbf{B})$, is defined as

$$\Lambda_q^{\perp}(\mathbf{B}) := \{\mathbf{y} \in \mathbb{Z}^m \mid \mathbf{y}\mathbf{B}^T = \mathbf{0} \bmod q\}.$$

The orthogonal lattice $\Lambda_q^{\perp}(\mathbf{B})$ will be used for defining the GPV framework.

GPV framework. The GPV framework is a hash-and-sign paradigm to construct a lattice-based signature [12]. Its security is guaranteed based on the hardness of the SIS problem, which has been proven under the classical/quantum random oracle model [4]. The GPV framework can be briefly described as follows.

- Public key: A full-rank matrix $\mathbf{P} \in \mathbb{Z}_q^{n \times m} (n < m)$ denotes a basis of an n-dimensional q-ary lattice $\Lambda_q(\mathbf{P})$. The public key contains the matrix $\mathbf{P}$, called a public basis.
- Secret key: A matrix $\mathbf{B} \in \mathbb{Z}_q^{m \times m}$ is a short basis of the q-ary lattice $\Lambda_q^{\perp}(\mathbf{P})$, satisfying $\mathbf{B} \times \mathbf{P}^T = \mathbf{0} \mod q$. The secret key contains the matrix $\mathbf{B}$, called a secret basis.
- Signing: Given a message $\mathbf{m}$ and a salt $\mathbf{r} \in \{0,1\}^k$, the signature $\mathbf{s} \in \mathbb{Z}_q^m$ of $\mathbf{m}$ is a short vector satisfying $\mathbf{s}\mathbf{P}^T = H(\mathbf{m}||\mathbf{r})$, where $H : \{0,1\}^* \to \mathbb{Z}_q^n$ is a hash function and $\mathbf{m}||\mathbf{r}$ denotes a concatenation of $\mathbf{m}$ and $\mathbf{r}$. The computation of signature $\mathbf{s}$ is performed as follows.
  1. Choose a salt $\mathbf{r} \leftarrow \{0,1\}^k$ and find a preimage $\mathbf{c} \in \mathbb{Z}_q^m$ such that $\mathbf{c}\mathbf{P}^T = H(\mathbf{m}||\mathbf{r})$.
  2. Compute $\mathbf{v} \leftarrow T_{\mathbf{B}}(\mathbf{c})$, where $T_{\mathbf{B}} : \mathbb{Z}^m \to \Lambda_q^{\perp}(\mathbf{P})$ is a trapdoor sampler that samples a lattice point $\mathbf{v} \in \Lambda_q^{\perp}(\mathbf{P})$ close to the input $\mathbf{c}$.
  3. Compute $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$. Since $\mathbf{v}$ satisfies $\mathbf{v}\mathbf{P}^T = \mathbf{0}$. Thus $\mathbf{s}\mathbf{P}^T = (\mathbf{c} - \mathbf{v})\mathbf{P}^T = \mathbf{c}\mathbf{P}^T = H(\mathbf{m}||\mathbf{r})$ and $\mathbf{s}$ is short.
- Verifying: If the signature $\mathbf{s}$ is short and satisfies $\mathbf{s}\mathbf{P}^T = H(\mathbf{m}||\mathbf{r})$, the signature is accepted. Otherwise, the signature is rejected.

A signature scheme based on the GPV framework, called GPV signature, is constructed by properly choosing the lattices $\Lambda_q(\mathbf{P})$ and $\Lambda_q^{\perp}(\mathbf{P})$, and a trapdoor sampler $T_{\mathbf{B}}$ which leaks no information about the secret basis $\mathbf{B}$ from the signature distribution. The Klein-GPV algorithm [12] is the first trapdoor sampler family, which is proven to leak no information about the secret basis $\mathbf{B}$ from the signature distribution.

2.4 Trapdoor Sampler

Gram-Schmidt Orthogonalization. The Gram-Schmidt orthogonalization (GSO) is a linear transformation that uniquely generates a set of mutually orthogonal vectors from a given set of linearly independent vectors. Let $\mathbf{B} \in \mathbb{R}^{n \times m}$ be a basis of a lattice. Then the Gram matrix $\mathbf{B}\mathbf{B}^*$ is uniquely decomposed by LDL^* decomposition as follows.

$$\mathbf{B}\mathbf{B}^* = \mathbf{L}\mathbf{D}\mathbf{L}^* = \mathbf{L}(\tilde{\mathbf{B}}\tilde{\mathbf{B}}^*)\mathbf{L}^*,$$

where $\mathbf{L} \in \mathbb{R}^{n \times n}$ is a lower triangular matrix, $\mathbf{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix, and $\tilde{\mathbf{B}}$ is the orthogonal basis of $\mathbf{B}$ such that $\mathbf{B} = \mathbf{L}\tilde{\mathbf{B}}$. The row vectors $\tilde{\mathbf{b}}_i$ of $\tilde{\mathbf{B}}$ are pairwise orthogonal.

Gaussians. Let $\rho_{\sigma,\mathbf{c}} : \mathbb{R}^n \to \mathbb{R}$ be the n-dimensional Gaussian PDF with the standard deviation σ and the center $\mathbf{c} \in \mathbb{R}^n$, defined as

$$\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \exp\left(-\frac{\|\mathbf{x}-\mathbf{c}\|^2}{2\sigma^2}\right).$$

For a lattice $\Lambda \subset \mathbb{R}^n$, the discrete Gaussian distribution $D_{\Lambda,\sigma,\mathbf{c}}$ over Λ with the standard deviation σ and the center $\mathbf{c} \in \mathbb{R}^n$ is defined as follows. For all $\mathbf{z} \in \Lambda$,

$$D_{\Lambda,\sigma,\mathbf{c}}(\mathbf{z}) := \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{z})}{\sum_{\mathbf{x}\in\Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})}.$$

If the center $\mathbf{c}$ is omitted from the discrete Gaussian distribution such as $D_{\Lambda,\sigma}$ then $\mathbf{c}$ is assumed to be $\mathbf{0}$.

Trapdoor Sampler. GGH [13] and NTRUSign [16] are signature schemes that generate short signatures by using Babai's rounding-off algorithm. The signature distributions of them can be exploited to recover the secret basis $\mathbf{B}$ by learning a parallelepiped scheme [25]. This attack scheme can recover the secret basis $\mathbf{B}$ even if Babai's nearest plane algorithm is used in the signature scheme.

In the GPV framework, a trapdoor sampler is constructed to leak no information about the secret basis $\mathbf{B}$ from the signature distribution. The trapdoor sampler makes the signature distribution statistically close to a discrete Gaussian distribution over the lattice $D_{\mathbf{c}+\mathcal{L}(\mathbf{B}),\sigma}$ which is isotropic and centered at zero ($\mathbb{E}_{\mathbf{s}}[\mathbf{s}] = \mathbf{0}$). Therefore, the learning a parallelepiped scheme cannot extract the information about $\mathbf{B}$ from the signature distribution. The Klein-GPV algorithm [12] is proven to prevent any leakage of the secret basis $\mathbf{B}$ because the signature distribution is statistically close to $D_{\mathbf{c}+\mathcal{L}(\mathbf{B}),\sigma}$. The hybrid sampler [6] and Peikert sampler [26] are similar to the Klein-GPV algorithm such that the signature distribution is statistically close to $D_{\mathbf{c}+\mathcal{L}(\mathbf{B}),\sigma}$. These trapdoor samplers are a type of randomized rounding-off algorithm or nearest-plane algorithm, which modifies the coefficients of the secret basis $\mathbf{B}$ or the orthogonal basis $\tilde{\mathbf{B}}$ to generate the signature $\mathbf{s}$ in the following form.

$$\mathbf{s} = \mathbf{c} - \mathbf{v} = (\mathbf{r} - \mathbf{u})\mathbf{A} = \mathbf{w}\mathbf{A}, \tag{1}$$

where $\mathbf{r}$ is a random vector which is generated by the trapdoor sampler using $\mathbf{c}$ and $\mathbf{A}$, $\mathbf{w}$ is the coefficient vector of $\mathbf{A}$, $\mathbf{u}$ is a partial information which is adjusted by the trapdoor sampler to construct the signature distribution close to $D_{\mathbf{c}+\mathcal{L}(\mathbf{B}),\sigma}$, and $\mathbf{A}$ is either $\mathbf{B}$ or $\tilde{\mathbf{B}}$. Each scheme employs a different matrix to generate the signature (Peikert sampler: $\mathbf{B}$, Klein-GPV algorithm: $\tilde{\mathbf{B}}$), which results in different variance of the signature distribution [29]. The trapdoor sampler in Falcon is a variant of the Klein-GPV algorithm, known as Ducas-Prest's fast Fourier orthogonal (FFO) sampler. Note that this FFO sampler applies Ducas FFO scheme [7] to the Klein-GPV algorithm to accelerate the sampling process.

2.5 An Overview of FALCON

NTRU. The NTRU lattice was first introduced in 1996 by Hoffstein, Pipher, and Silverman to construct a lattice over a ring [17]. Let $n, q \in \mathbb{Z}^+$, $\mathcal{R} := \mathbb{Z}[x]/\langle x^n+1\rangle$ and $\mathcal{R}_q := \mathbb{Z}_q[x]/\langle x^n+1\rangle$ be the quotient ring of the polynomial ring $\mathbb{Z}[x]$ and $\mathbb{Z}_q[x]$, respectively. The NTRU lattice employs a ring structure, which reduces the public key size to a single polynomial $h \in \mathcal{R}_q$. Let $f, g, F, G \in \mathcal{R}$ be the small secret polynomials of the NTRU lattice, which satisfy the following NTRU equation:

$$fG - gF = q \mod (x^n + 1).$$

The secret key f should be invertible modulo q and the NTRU public key h is generated as $h = gf^{-1} \mod q$. If the average norms of f and g are slightly larger than $\sqrt{q}$ and n is a power of two, it was proven that h is statistically indistinguishable from a uniformly sampled element in $\mathcal{R}_q$ [33]. However, in practice, the NTRU assumption still states that it is hard to find small secret polynomials $f, g \in \mathcal{R}$ from the public key $h \in \mathcal{R}_q$ [27].

When instantiating the GPV framework over the NTRU lattice, the public basis $\mathbf{P} \in \mathcal{R}_q^2$ and the secret basis $\mathbf{B} \in \mathcal{R}^{2\times 2}$ are determined as follows:

$$\mathbf{P} = \begin{pmatrix} 1 & h^* \end{pmatrix}, \ \mathbf{B} = \begin{pmatrix} g & -f \\ G & -F \end{pmatrix}.$$

Here, h^* is the Hermitian adjoint which is a unique polynomial in $\mathbb{Q}[x]/\langle x^n+1\rangle$ satisfying $h^*(\zeta) = \overline{h(\zeta)}$ for any root ζ of $x^n + 1$ where $\bar{\cdot}$ denotes the complex conjugate over $\mathbb{C}$. Since polynomial multiplication over $\mathcal{R}$ or $\mathcal{R}_q$ can be performed by negacyclic matrix multiplication, all polynomials in $\mathcal{R}$ or $\mathcal{R}_q$ can be expressed as negacyclic matrices. A negacyclic matrix has a property that each row is a cyclic shift of the previous row, with the last element negated. Since each component polynomial of $\mathbf{P}$ and $\mathbf{B}$ can be replaced by a negacyclic matrix, $\mathbf{P}$ and $\mathbf{B}$ can be represented as the matrices in $\mathbb{Z}_q^{n\times 2n}$ and $\mathbb{Z}^{2n\times 2n}$, respectively.

Note that converting h^* into a negacyclic matrix is the same as converting h into a negacyclic matrix and then taking the conjugate transpose. Hence, the operator $*$ for the Hermitian adjoint polynomial h^* is the same as the Hermitian (conjugate and transpose) operator $*$ for making Hermitian matrix. As mentioned in Section 2.3, $\mathbf{B}$ and $\mathbf{P}$ are orthogonal to each other such that $\mathbf{BP}^* = \mathbf{0} \mod q$.

Falcon Signature. Falcon is the GPV signature whose trapdoor sampler is a FFO sampler over the NTRU lattice [30]. Let $q = 12289$ and n be a power of two. The FFO sampler efficiently implements the Klein-GPV algorithm on the ring structure such as $\mathcal{R}$. Since the public key is a polynomial $h \in \mathcal{R}_q$, the NTRU lattice reduces the size of public key. The combination of FFO sampler and NTRU lattice makes Falcon one of the most compact and efficient schemes in the NIST standard. In the NIST standard, Falcon-512 and Falcon-1024 are adopted where 512 and 1024 denote the dimension n of the polynomial $x^n + 1$.

Algorithm 1. FalconSign($\mathbf{m}, \mathbf{sk}$)

```
Input: A message $\mathbf{m}$ and a secret key $\mathbf{sk}$
Output: A signature $sig$ of $\mathbf{m}$
1: $\mathbf{r} \leftarrow \{0,1\}^{320}$
2: $c \leftarrow HashToPoint(\mathbf{r}||\mathbf{m}, q, n)$    ▷ $c \in \mathcal{R}_q$
3: $\mathbf{t} \leftarrow (c, 0)\mathbf{B}^{-1}$
4: while $\|\mathbf{s}\|^2 > \lfloor 2.42 \cdot n \cdot \sigma^2 \rfloor$ do    ▷ $\sigma = \frac{1.17}{\pi\sqrt{2}} \cdot \sqrt{q \cdot \log\left(4n(1 + 2^{32} \cdot \sqrt{n/4})\right)}$
5:     $\mathbf{z} \leftarrow ffsampling_n(\mathbf{t}, T)$    ▷ $\mathbf{zB} \sim D_{\mathcal{L}(\mathbf{B}),\sigma,(c,0)}$
6:     $\mathbf{s} \leftarrow (\mathbf{t} - \mathbf{z})\mathbf{B}$    ▷ $\mathbf{s} \sim D_{\mathcal{L}(\mathbf{B})+(c,0),\sigma}$
7: end while
8: return $sig = (\mathbf{r}, \mathbf{s})$
```

Algorithm 1 shows a pseudo-code of Falcon signing. As described in [30], Falcon utilizes FFT to efficiently perform polynomial multiplication in $\mathcal{R}$. However, Algorithm 1 in this paper omits the FFT part for the simplicity of explanation. The inputs $\mathbf{m}$ and $\mathbf{sk} = (\mathbf{B}, T)$ represent the message string and the secret key of Falcon, respectively, where $\mathbf{B}$ is the secret basis and T is the LDL^* tree of $\mathbf{B}$. $HashToPoint$ is a hash function mapping a string $(\mathbf{r}||\mathbf{m}, q, n)$ to a polynomial in $\mathcal{R}_q$. Falcon generates the signature $\mathbf{s}$ by finding a lattice point $\mathbf{zB} \sim D_{\mathcal{L}(\mathbf{B}),\sigma,(c,0)}$ using the FFO sampler $ffsampling_n$. The FFO sampler ensures that the signature distribution is (statistically) close to $D_{\mathcal{L}(\mathbf{B})+(c,0),\sigma}$ which leaks no information about the secret basis $\mathbf{B}$.

Algorithm 2. SamplerZ(c, σ')

```
Input: The center of the discrete Gaussian distribution $c \in \mathbb{R}$ and the standard deviation $\sigma' \in [\sigma_{min}, \sigma_{max}]$
Output: An integer $z \in \mathbb{Z}$ such that $z \sim D_{\mathbb{Z},\sigma',c}$
1: $\bar{c} \leftarrow c - \lfloor c \rfloor$
2: $z_0 \leftarrow BaseSampler()$    ▷ See Algorithm 3
3: $b \leftarrow \{0,1\}$
4: $\bar{z} \leftarrow b + (2 \cdot b - 1) \cdot z_0$
5: $x \leftarrow \frac{(\bar{z}-\bar{c})^2}{2\sigma'^2} - \frac{z_0^2}{2\sigma_{max}^2}$
6: $z \leftarrow \bar{z} + \lfloor c \rfloor$
7: return $z$ passing with a probability of $\frac{\sigma_{min}}{\sigma'} \exp(-x)$, otherwise go to Line 2 and restart
```

An FFO sampler requires an oracle that exactly samples from the discrete Gaussian distribution over integers $D_{\mathbb{Z},\sigma,\lambda}$ for any desired $\sigma > 0$ and $\lambda \in \mathbb{R}$. The FFO sampler in Falcon uses the SamplerZ in Algorithm 2 as a discrete Gaussian sampler over integers. SamplerZ performs rejection sampling to sample an integer z from the distribution close to $D_{\mathbb{Z},\sigma',c}$ in a constant time [31]. If the operation time is not constant, the SamplerZ is vulnerable to timing attacks that estimate the standard deviation σ' to recover the secret basis $\mathbf{B}$ [10]. BaseSampler is a half-discrete Gaussian sampler which samples an integer z_0 from $D_{\mathbb{Z}^+,\sigma_{max}}$ and the value $b \in \{0,1\}$ determines the sign of integer $\bar{z}$. In the reference code of Falcon, BerExp function is used to pass z with probability $\frac{\sigma_{\max}}{\sigma'}\exp(-x)$ [30]. However, the BerExp function is omitted for the simplicity of explanation.

Algorithm 3. BaseSampler()

```
Input: -
Output: An integer $z_0 \in \{0,1,\ldots,18\}$ such that $z_0 \sim D_{\mathbb{Z}^+,\sigma_{max}}$
1: $u \leftarrow \{0,1\}^{72}$
2: $z_0 \leftarrow 0$                              ▷ Set the initial value of $z_0$ as 0
3: for $i = 0,1,\ldots,17$ do
4:     $z^+ \leftarrow [[u < \mathbf{RCDT}[i]]]$
5:     $z_0 \leftarrow z_0 + z^+$
6: end for
7: return $z_0$
```

Algorithm 3 shows a pseudo-code of BaseSampler as a half-discrete Gaussian sampler from $D_{\mathbb{Z}^+,\sigma_{max}}$. The array $\mathbf{RCDT}$ denotes the (scaled) reverse cumulative distribution table, where $\mathbf{RCDT}[i]$ is equal to $2^{72} - 2^{72} \cdot \sum_{k=0}^{i} D_{\mathbb{Z}^+,\sigma_{max}}(k)$ for all $i \in \{0,1,\ldots,18\}$. A random 72-bit u is compared with $\mathbf{RCDT}[i]$ for $i \in \{0,1,\ldots,18\}$ to determine the value $z^+ \in \{0,1\}$. If $u < \mathbf{RCDT}[i]$, then $z^+ = 1$, and otherwise, $z^+ = 0$. The output z_0 of BaseSampler is computed by summing all z^+.

3 Secret Key Recovery Attack on GPV Signatures Using Partial Information about Signature

In this section, we propose a secret key recovery attack, called OLS attack, on GPV signatures based on ordinary least squares (OLS) regression, and demonstrate the consistency of the proposed scheme.

3.1 Problem Setup for OLS Attack on GPV Signatures

There have been a few research results on the side-channel attack using the vector $\bar{\mathbf{z}} = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_{2n})$ to recover the secret basis $\mathbf{B}$ of Falcon, where $\bar{z}_i$ is used to compute $z_i = \bar{z}_i + \lfloor c_i \rfloor$ in Algorithm 2 for the output $\mathbf{z} = (z_1, z_2, \ldots, z_{2n})$ of FFO sampler. The timing attack in [10] aims to estimate all the variance σ_i of $\bar{z}_i$ distributed according to D_{σ_i,r_i} and to recover the secret basis $\mathbf{B}$ of Falcon using them. Recently, power analysis attack schemes are proposed to classify the signatures $\mathbf{s}$ according to the first coordinate $\bar{z}_1$ as 0 or 1, and then to recover the secret basis $\mathbf{B}$ of Falcon by using the signatures with $\bar{z}_1 = 0$ or 1 through the hidden parallelepiped or the learning slice schemes [15], [37]. As will be derived in Section 4.1, $\bar{\mathbf{z}}$ in Falcon can be used as a partial information $\mathbf{u}$ of signature $\mathbf{s}$ in Eq. (1) such that

$$\mathbf{s} = (\mathbf{r} - \bar{\mathbf{z}})\tilde{\mathbf{B}} \text{ for } r_i \sim \chi_{\zeta_i,\lambda_i} \text{ and } \bar{z}_i \sim D_{\mathbb{Z},\sigma_i,r_i}, \tag{2}$$

where $\mathbf{r} = (r_1, r_2, \ldots, r_{2n})$ is an unknown random vector, χ_{ζ_i,λ_i} is a distribution of r_i whose support is $[0, 1)$ for $i \in \{1, 2, \ldots, 2n\}$, and $\tilde{\mathbf{B}}$ is the orthogonal basis of $\mathbf{B}$. If the signature $\mathbf{s}$ is expressed as in Eq. (2), the proposed OLS attack can be effectively applied as presented in Section 3.3.

Next, we will generalize Eq. (2) from the Falcon signature to the GPV signature, in order to apply the OLS attack more broadly. Consider a GPV signature described in Section 2.3. Let $\mathbf{B} = \left(\mathbf{b}_1^T\ \mathbf{b}_2^T\ \cdots\ \mathbf{b}_n^T\right)^T \in \mathbb{Z}_q^{n\times n}$ be the secret basis of GPV signature and $\mathbf{s}$ be the signature. The coefficient vector $\mathbf{w}$ in Eq. (1) must satisfy $\mathbb{E}_{\mathbf{w}}[\mathbf{w}] = \mathbf{0}$ because $\mathbb{E}_{\mathbf{s}}[\mathbf{s}] = \mathbf{0}$ as described in Section 2.4. To generate $\mathbf{w}$ with $\mathbb{E}_{\mathbf{w}}[\mathbf{w}] = \mathbf{0}$, a partial information $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ used in the trapdoor sampler must satisfy the following relation:

$$\mathbf{s} = \mathbf{w}\mathbf{A} = (\mathbf{r} - \mathbf{u})\mathbf{A} \text{ for } r_i \sim \Upsilon_{\zeta_i,\lambda_i} \text{ and } u_i \sim Q_{\sigma_i,r_i}, \tag{3}$$

where $\mathbf{A}$ is either $\mathbf{B}$ or $\tilde{\mathbf{B}}$, $\mathbf{r} = (r_1, r_2, \ldots, r_n)$ is an unknown random vector, and ζ_i and σ_i are also unknown. Similar to $\bar{\mathbf{z}}$ in Eq. (2) for Falcon, $\mathbf{u}$ used in the trapdoor sampler of GPV signature satisfy $\mathbb{E}_{\mathbf{u}|\mathbf{r}}[\mathbf{u} \mid \mathbf{r}] = \mathbf{r}$ to ensure that $\mathbb{E}_{\mathbf{w}}[\mathbf{w}] = \mathbb{E}_{\mathbf{r},\mathbf{u}}[\mathbf{r} - \mathbf{u}] = \mathbb{E}_{\mathbf{r}}[\mathbf{r} - \mathbb{E}_{\mathbf{u}|\mathbf{r}}[\mathbf{u}\,|\,\mathbf{r}]] = \mathbb{E}_{\mathbf{r}}[\mathbf{r} - \mathbf{r}] = \mathbf{0}$ for $\mathbf{r}$.

If $\mathbf{u}$ and $\mathbb{E}_{\mathbf{r}}[\mathbf{r}] = \boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ of Eq. (3) are estimated through side-channel attack or other method, an attacker attempting to recover the secret key faces the following problem.

Problem 1. Let $\mathbf{B} \in GL_n(\mathbb{Z})$ be the secret basis of GPV signature and $\mathbf{s} \in \mathbb{Z}^n$ be a signature. Define the probabilistic algorithm $W_{\Upsilon,Q}(\mathbf{A})$ which outputs $(\mathbf{s}, \mathbf{u})$ such that $\mathbf{s} = (\mathbf{r} - \mathbf{u})\mathbf{A}$ for $r_i \sim \Upsilon_{\zeta_i,\lambda_i}$ and $u_i \sim Q_{\sigma_i,r_i}$. Here, $\mathbf{A}$ is either $\mathbf{B}$ or $\tilde{\mathbf{B}}$, and $\mathbf{r}$, ζ_i, σ_i are unknown. Given N statistically independent samples $(\mathbf{s}, \mathbf{u}) \leftarrow W_{\Upsilon,Q}(\mathbf{A})$ and $\mathbb{E}_{\mathbf{r}}[\mathbf{r}] = \boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n) \in \mathbb{R}^n$, find a good approximation of $\mathbf{A}$. Note that, in this case, $\mathbf{u}$ is the partial information about $\mathbf{s}$.

Definitely, the solution of Problem 1 is an approximation of $\mathbf{B}$ or $\tilde{\mathbf{B}}$ which serves as the secret key of GPV signatures.

3.2 Transforming Problem 1 into Multiple Linear Regression Problem

Consider m statistically independent samples $(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{A})$ in Problem 1, where $\mathbf{s}_i$ is the ith GPV signature and $\mathbf{u}_i$ is the partial information about $\mathbf{s}_i$ for $i \in \{1, 2, \ldots, m\}$. Let $S_m = \{(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{A}) \mid i \in \{1, 2, \ldots, m\}\}$. For any $i \in \{1, 2, \ldots, m\}$, the vector $\mathbf{r}_i \in \mathbb{R}^n$, used for sampling $(\mathbf{s}_i, \mathbf{u}_i)$ from $W_{\Upsilon,Q}(\mathbf{A})$, satisfies $\mathbf{s}_i = (\mathbf{r}_i - \mathbf{u}_i)\mathbf{A}$, which can be transformed as follows:

$$\boldsymbol{\lambda} - \mathbf{u}_i = \mathbf{s}_i\mathbf{A}^{-1} + (\boldsymbol{\lambda} - \mathbf{r}_i), \tag{4}$$

where $\boldsymbol{\lambda}$ denotes the expectation of $\mathbf{r}_i$. Eq. (4) can be regarded as a multiple linear regression (MLR) model as explained below, which is used for recovering $\mathbf{A}$ in Section 3.3. The MLR model for the samples $(\mathbf{s}_i, \mathbf{u}_i) \in S_m$, $i \in \{1, 2, \ldots, m\}$, is as follows.

$$\mathbf{y}_i = \mathbf{x}_i \mathbf{A}^{-1} + \mathbf{e}_i, \tag{5}$$

where $\mathbf{y}_i := \boldsymbol{\lambda} - \mathbf{u}_i$ is the dependent output vector, $\mathbf{x}_i := \mathbf{s}_i$ is the independent input vector, and $\mathbf{e}_i := \boldsymbol{\lambda}_i - \mathbf{r}_i$ is the noise vector. Then, $\mathbf{x}_i \in \mathbb{Z}^n$ and $\mathbf{e}_i \in \mathbb{R}^n$ of the proposed MLR model in Eq. (5) satisfy the following Lemma 1.

Lemma 1 (Uncorrelatedness). *Let $\mathbf{x} = \mathbf{s} \in \mathbb{Z}^n$ and $\mathbf{e} = \boldsymbol{\lambda} - \mathbf{r} \in \mathbb{R}^n$ be the independent input and noise vectors, respectively, in Eq. (5). Then, $\mathbf{x}$ and $\mathbf{e}$ are uncorrelated, i.e., $\mathbf{K}_{\mathbf{xe}} = Cov[\mathbf{x}, \mathbf{e}] = \mathbf{0}$.*

The complete proof of Lemma 1 is provided in Supplementary Material A. The MLR model in Eq. (5) can be represented in matrix form for the given set S_m as follows.

$$\mathbf{Y} = \mathbf{X}\mathbf{A}^{-1} + \mathbf{E},\ \mathbf{Y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_m \end{pmatrix},\ \mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_m \end{pmatrix},\ \mathbf{E} = \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \vdots \\ \mathbf{e}_m \end{pmatrix}, \tag{6}$$

where $\mathbf{Y} \in \mathbb{R}^{m \times n}$ is the dependent output matrix, $\mathbf{X} \in \mathbb{Z}^{m \times n}$ is the independent input matrix, and $\mathbf{E} \in \mathbb{R}^{m \times n}$ is the noise matrix. Then, the proposed MLR model in Eq. (6) satisfies the following Lemma 2.

Lemma 2. *Let $S_m = \{(\mathbf{s}_i, \mathbf{u}_i) \mid i \in \{1, 2, \ldots, m\}\}$ be a set of m independent samples $(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{A})$. Let $\mathbf{X} \in \mathbb{Z}^{m \times n}$ and $\mathbf{E} \in \mathbb{R}^{m \times n}$ be the input and noise matrices, respectively, in Eq. (6). As m goes to infinity, $\frac{1}{m}\mathbf{X}^T\mathbf{E}$ converges to $\mathbf{0}$ in probability, and $\frac{1}{m}\mathbf{X}^T\mathbf{X}$ converges to $Cov[\mathbf{s}, \mathbf{s}]$ in probability. In other words,*

$$\operatorname*{plim}_{m \to \infty} \left(\frac{1}{m} \mathbf{X}^T \mathbf{E} \right) = \mathbf{0} \ \textit{and} \ \operatorname*{plim}_{m \to \infty} \left(\frac{1}{m} \mathbf{X}^T \mathbf{X} \right) = Cov[\mathbf{s}, \mathbf{s}].$$

The complete proof of Lemma 2 is provided in Supplementary Material A. Lemma 2 is used in Section 3.3 for proving the consistency of the proposed OLS attack.

3.3 OLS Attack on GPV Signatures Using Partial Information about Signature

In this section, an ordinary least squares (OLS) attack on GPV signatures, which estimates $\mathbf{A}$, is proposed and it is shown that this estimator of $\mathbf{A}$ is consistent. Algorithm 4 is a pseudo-code of the OLS attack for recovering $\mathbf{A}$ from the collected samples $(\mathbf{s}, \mathbf{u})$ using the proposed MLR model in Eq. (6). The inputs of the OLS attack in Algorithm 4 are a sample set S_m consisting of m independent

Algorithm 4. OLS attack($S_m, \boldsymbol{\lambda}$)

```
Input: A sample set $S_m = \{(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{A}) \mid i \in \{1, 2, \ldots, m\}\}$ and the expected
    vector $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n)$
Output: $\widehat{\mathbf{A}}$: an approximations of secret key $\mathbf{A}$
1: for $i = 1, 2, \ldots, m$ do
2:     $\mathbf{x}_i \leftarrow \mathbf{s}_i$                          ▷ $\mathbf{s}_i$ is the $i$th GPV signature
3:     $\mathbf{y}_i \leftarrow \boldsymbol{\lambda} - \mathbf{u}_i$      ▷ $\boldsymbol{\lambda} = \mathbb{E}_{\mathbf{r}}[\mathbf{r}]$ and $\mathbf{u}_i$ is partial information of $\mathbf{s}_i$ in Eq. (3)
4: end for
5: $\mathbf{X} \leftarrow \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_m \end{pmatrix}$ and $\mathbf{Y} \leftarrow \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_m \end{pmatrix}$      ▷ Vertical concatenation of each $\mathbf{x}_i$ and $\mathbf{y}_i$
6: $\widehat{\mathbf{A}} \leftarrow (\mathbf{X}^T\mathbf{Y})^{-1}\mathbf{X}^T\mathbf{X}$
7: return $\widehat{\mathbf{A}}$
```

samples $(\mathbf{s}, \mathbf{u}) \leftarrow W_{\Upsilon,Q}(\mathbf{A})$, and an expected vector $\boldsymbol{\lambda} = \mathbb{E}_{\mathbf{r}}[\mathbf{r}]$ (see Eq. (4)). The output $\widehat{\mathbf{A}}$ in Algorithm 4 is the estimation of the secret key $\mathbf{A}$.

By applying an ordinary least squares (OLS) estimator to the proposed MLR model in Eq. (6), we can obtain an estimation of $\mathbf{A}^{-1}$ as follows:

$$\widehat{\mathbf{A}^{-1}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}. \tag{7}$$

It is clear that the estimator $\widehat{\mathbf{A}^{-1}}$ minimizes the squared error $\sum_{i=1}^{m}(\mathbf{y}_i - \mathbf{x}_i\widehat{\mathbf{A}^{-1}})(\mathbf{y}_i - \mathbf{x}_i\widehat{\mathbf{A}^{-1}})^T$ over the sample set S_m. Note that $\mathbf{x}_i$ and $\mathbf{y}_i$ are the ith row vectors of $\mathbf{X}$ and $\mathbf{Y}$, respectively. The estimator $\widehat{\mathbf{A}}$ is the inverse of $\widehat{\mathbf{A}^{-1}}$ in Eq. (7) as follows:

$$\widehat{\mathbf{A}} = \left(\widehat{\mathbf{A}^{-1}}\right)^{-1} = (\mathbf{X}^T\mathbf{Y})^{-1}\mathbf{X}^T\mathbf{X}. \tag{8}$$

Since $\widehat{\mathbf{A}}$ in Eq. (8) is a simple linear estimator, the proposed OLS attack is very efficient and its consistency is proven in the following Theorem 4. Note that the consistency implies that the linear estimator $\widehat{\mathbf{A}}$ converges to $\mathbf{A}$ in probability as the number of samples $(\mathbf{s}, \mathbf{u})$ increases.

Theorem 4 (Consistency of the Proposed OLS Estimator). *Let* $S_m = \{(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{B}) \mid i \in \{1, 2, \ldots, m\}\}$ *be a set of* m *independent GPV signature samples. Let* $\mathbf{X} \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{Y} \in \mathbb{R}^{m\times n}$ *be the input and output matrices, respectively, in Eq. (6). The estimator* $\widehat{\mathbf{A}} = (\mathbf{X}^T\mathbf{Y})^{-1}\mathbf{X}^T\mathbf{X}$ *of the OLS attack in Algorithm 4 is consistent, i.e.,* $\operatorname{plim}_{m\to\infty} \widehat{\mathbf{A}} = \mathbf{A}$.

The complete proof of Theorem 4 is provided in Supplementary Material A. Therefore, the more samples $(\mathbf{s}, \mathbf{u})$ are collected, the more accurately $\mathbf{A}$ is estimated. If $\mathbf{A}$ is the secret basis $\mathbf{B}$, the secret key of any GPV signature is directly recovered by the proposed OLS attack in Algorithm 4. Otherwise, one more step is required to recover the secret basis $\mathbf{B}$ from the orthogonal basis $\tilde{\mathbf{B}}$. In [10], a scheme is proposed to recover the NTRU secret basis $\mathbf{B}$ using $\|\tilde{\mathbf{b}}_i\|$, $i \in \{1, 2, \ldots, n\}$, where $\tilde{\mathbf{b}}_i$ is the ith row of $\tilde{\mathbf{B}}$. In addition, the NTRU secret basis $\mathbf{B}$ can be more easily recovered from $\tilde{\mathbf{B}}$ by the scheme proposed in Section 4.1. However, it is still an interesting open problem to recover the secret basis $\mathbf{B}$ from the orthogonal basis $\tilde{\mathbf{B}}$ for arbitrary lattice.

4 An Efficient Practical Implementation of OLS Attack on Falcon

In this section, we introduce an efficient implementation scheme for recovering the secret basis $\mathbf{B}$ of Falcon based on the OLS attack in Algorithm 4. The proposed implementation scheme reduces the number of samples necessary for attack on SamplerZ by $\frac{1}{2n}$ times compared to the general OLS attack in Algorithm 4. In addition, an error-tolerant power analysis attack on BaseSampler is proposed to further improve the attack efficiency by correcting the erroneous samples z_0 obtained from BaseSampler.

4.1 Partial Information of Falcon and OLS Attack on Falcon

To perform the OLS attack on Falcon, it is necessary to determine $\mathbf{r}$, $\mathbf{u}$, and $\mathbf{A}$ in Eq. (3) corresponding to Falcon. Let $\mathbf{B} \in GL_{2n}(\mathbb{Z})$ be the secret basis of Falcon. We have set that the partial information $\mathbf{u}$ and the secret key $\mathbf{A}$ in Eq. (3) for Falcon are $\bar{\mathbf{z}} = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_{2n})$ and the orthogonal basis $\tilde{\mathbf{B}}$, respectively, where $\bar{z}_i$ is used to compute $z_i = \bar{z}_i + \lfloor c_i \rfloor$ in Algorithm 2 for the output $\mathbf{z} = (z_1, z_2, \ldots, z_{2n})$ of FFO sampler. The vector $\bar{\mathbf{z}}$ satisfies the following relation, which will be shown in Corollary 1.

$$\mathbf{s} = (\mathbf{r} - \bar{\mathbf{z}})\tilde{\mathbf{B}} \text{ for } r_i \sim \chi_{\zeta_i,\lambda_i} \text{ and } \bar{z}_i \sim D_{\mathbb{Z},\sigma_i,r_i}, \tag{9}$$

where $\mathbf{r} \in [0, 1)^{2n}$ is an unknown random vector and χ_{ζ_i,λ_i} is the distribution of r_i for $i \in \{1, 2, \ldots, 2n\}$. It should be noted that $\bar{\mathbf{z}}$ can be estimated by using a power analysis attack as described in Section 5.1. Before proving Corollary 1, we will first examine the following Lemma 3 for the output $\mathbf{z}$ of the FFO sampler with the signature $\mathbf{s}$.

Lemma 3 (Informal, see Lemma 4.4 in [12] for a formal statement). *Let T be the LDL^* tree of Falcon, $c \in \mathcal{R}$ be the output of HashToPoint, and $\mathbf{t} \in \mathbb{R}^{2n}$ be $(c, 0)\mathbf{B}^{-1}$ in Algorithm 1. Then, for any input $(\mathbf{t}, T)$ and any output $\mathbf{z}$ of FFO sampler $ffsampling_n(\mathbf{t}, T)$, we have*

$$\mathbf{s} = (\tilde{\mathbf{t}} - \mathbf{z})\tilde{\mathbf{B}},$$

where $\tilde{\mathbf{t}} = \mathbf{t}\mathbf{L} - \mathbf{z}\mathbf{L} + \mathbf{z}$ and $\mathbf{L} \in \mathbb{R}^{2n\times 2n}$ is a lower triangular matrix in LDL^* decomposition of Gram matrix $\mathbf{B}\mathbf{B}^*$. The ith coordinate z_i of $\mathbf{z}$ is distributed according to $D_{\mathbb{Z},\sigma_i,\tilde{t}_i}$ and $\tilde{t}_i$ is the ith coordinate of $\tilde{\mathbf{t}}$ for $i \in \{1, 2, \ldots, 2n\}$.

Lemma 4.4 in [12] addresses the relationship between the output of the Klein-GPV algorithm and the signature. Since the FFO sampler is a variant of the Klein-GPV algorithm, which employs a quotient ring to efficiently run the nearest plane algorithm [7], the relationship between the signature $\mathbf{s}$ and the output $\mathbf{z}$ of the FFO sampler in Falcon is the same as the relationship in Lemma 4.4 in [12]. Lemma 3 shows the relationship between $\mathbf{s}$ and $\mathbf{z}$ in Falcon, which leads to Corollary 1 implying that $\bar{z}$ of SamplerZ can be used as partial information for recovering the orthogonal basis $\tilde{\mathbf{B}}$ of Falcon.

Corollary 1. *Let $\mathbf{z} = (z_1, z_2, \ldots, z_{2n}) \in \mathbb{Z}^{2n}$ be the output of FFO sampler $ffsampling_n$, and let $\bar{\mathbf{z}} = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_{2n}) \in \mathbb{Z}^{2n}$ be the vector such that $\bar{z}_i$ is calculated from $z_i = \bar{z}_i + \lfloor c_i \rfloor$ in Algorithm 2. Here, c_i, serving as an input to SamplerZ for producing the output z_i, is equal to $\tilde{t}_i$ of $\tilde{\boldsymbol{t}} = (\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_{2n})$ in Lemma 3. Then for any signature $\mathbf{s} \in \mathbb{Z}^{2n}$ of Falcon and $\bar{\mathbf{z}}$, there exists $\mathbf{r} = (r_1, r_2, \ldots, r_{2n}) \in [0,1)^{2n}$ such that*

$$\mathbf{s} = (\mathbf{r} - \bar{\mathbf{z}})\tilde{\mathbf{B}},$$

where $\bar{z}_i$ is distributed according to $D_{\mathbb{Z},\sigma_i,r_i}$ for $i \in \{1, 2, \ldots, 2n\}$.

The complete proof of Corollary 1 is provided in Supplementary Material A. If you look at Problem 1 from the viewpoint of Falcon, Corollary 1 implies that the outputs $(\mathbf{s}, \bar{\mathbf{z}})$ of the probabilistic algorithm $W_{\chi,D}(\tilde{\mathbf{B}})$ satisfy the relation in Eq. (9). Since various attack schemes assume that $\mathbf{r}$ is sampled from $[0,1)^{2n}$ uniformly at random [15], [25], [37], we assume that the expected vector $\boldsymbol{\lambda}$ in Algorithm 4 is $\left(\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}\right)$. Let m independent samples $(\mathbf{s}, \bar{\mathbf{z}}) \leftarrow W_{\chi,D}(\tilde{\mathbf{B}})$ be collected, where the support of χ_{ζ_i,λ_i} is $[0,1)$. In practice, these m vectors $\bar{\mathbf{z}}$ are independently collected by applying power analysis attack to m Falcon signatures made by the same secret basis $\mathbf{B}$. As in Theorem 4, the estimator $\widehat{\mathbf{A}}$ in Eq. (8) converges to the orthogonal basis $\tilde{\mathbf{B}}$ of Falcon in probability.

In Falcon, the secret basis $\mathbf{B}$ can be easily recovered from the orthogonal basis $\tilde{\mathbf{B}}$ by using the properties of NTRU lattice and GSO. The secret basis $\mathbf{B}$ is constructed by the secret polynomials $f, g, F, G \in \mathbb{Z}[x]/\langle x^n + 1 \rangle$ as follows:

$$\mathbf{B} = \begin{pmatrix} g & -f \\ G & -F \end{pmatrix}.$$

Note that these secret polynomials satisfy the NTRU equation, $f \cdot G - F \cdot g = q \mod (x^n + 1)$. In Falcon, F and G are obtained by solving this NTRU equation given f and g [28], [30]. Thus, the secret basis $\mathbf{B}$ of Falcon can be recovered by finding f and g. The first row $\mathbf{b}_1$ of $\mathbf{B}$ contains all the coefficient information of secret polynomials $f(x) = \sum_{i=0}^{n-1} f_i x^i$ and $g(x) = \sum_{i=0}^{n-1} g_i x^i$ such that

$$\mathbf{b}_1 = (g_0, g_1, \ldots, g_{n-1}, -f_0, -f_1, \ldots, -f_{n-1}).$$

Since the first row $\mathbf{b}_1$ of $\mathbf{B}$ is the same as the first row $\tilde{\mathbf{b}}_1$ of $\tilde{\mathbf{B}}$ due to the property of GSO, f and g can be recovered only from $\tilde{\mathbf{b}}_1$, which is obtained by estimating $\tilde{\mathbf{B}}$ using the OLS attack in Algorithm 4 with samples and $\boldsymbol{\lambda}$. Therefore, we only need the first row $\tilde{\mathbf{b}}_1$ of $\tilde{\mathbf{B}}$. Given Falcon-n (in general, $n = 512$, 1024) and m samples, the time complexity for computing $\widehat{\mathbf{A}} = \widehat{\tilde{\mathbf{B}}}$ in Eq. (8) is given as

$$\mathcal{O}(mn^2 + n^3).$$

It is because computing $\mathbf{X}^T\mathbf{Y}$, $\mathbf{X}^T\mathbf{X}$, and $(\mathbf{X}^T\mathbf{Y})^{-1}$ take $\mathcal{O}(mn^2)$, $\mathcal{O}(mn)$, and $\mathcal{O}(n^3)$ operations, respectively.

4.2 Efficient OLS Attack on Falcon

Based on the analysis in Section 4.1, an efficient OLS attack on Falcon to recover the secret basis $\mathbf{B}$ is performed only using $\bar{z}_1$ of the partial information $\bar{\mathbf{z}}$ because $\mathbf{B}$ (equivalently, the secret polynomials f and g) can be recovered by estimating the first row $\tilde{\mathbf{b}}_1$ of the orthogonal basis $\tilde{\mathbf{B}}$. Note that $\tilde{\mathbf{b}}_1$ is a scaled transpose of the first column vector $\mathbf{v}_1 \in \mathbb{R}^{2n}$ of $\tilde{\mathbf{B}}^{-1}$ such that

$$\tilde{\mathbf{b}}_1 = \frac{\mathbf{v}_1^T}{\|\mathbf{v}_1\|^2}. \tag{10}$$

It is clear that $\tilde{\mathbf{b}}_1 = \mathbf{v}_1^T/\|\mathbf{v}_1\|^2$ is the first row of $\tilde{\mathbf{B}}$ because the rows of $\tilde{\mathbf{B}}$ are mutually orthogonal. In practice, since $\mathbf{v}_1$ is a real coefficient vector, we need to round each coordinate of $\mathbf{v}_1^T/\|\mathbf{v}_1\|^2$ in Eq. (10) to obtain an integer vector $\tilde{\mathbf{b}}_1$.

Since the estimator $\widehat{\mathbf{A}^{-1}}$ in Eq. (7) estimates $\tilde{\mathbf{B}}^{-1}$ in Falcon, the first column $\mathbf{v}_1$ of $\tilde{\mathbf{B}}^{-1}$ is estimated only using the first column $\mathbf{y}'_1$ of $\mathbf{Y}$ in Eq. (7) as follows.

$$\widehat{\mathbf{v}_1} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}'_1, \tag{11}$$

where $\widehat{\mathbf{v}_1}$ is the estimation of $\mathbf{v}_1$. Note that the first column $\mathbf{y}'_1$ of $\mathbf{Y}$ consists of m elements $\lambda_1 - u_{i,1} = \frac{1}{2} - \bar{z}_{i,1}$ where $u_{i,1}$ and $\bar{z}_{i,1}$ are the first coordinate of $\mathbf{u}_i$ and $\bar{\mathbf{z}}_i$, respectively, for $i \in \{1, 2, \ldots, m\}$, and $\bar{z}_{i,1}$'s are obtained by power analysis attack to the SamplerZ. The first row $\mathbf{b}_1$ of $\mathbf{B}$ is recovered by scaling $\widehat{\mathbf{v}_1}$ as in Eq. (10) such that

$$\widehat{\mathbf{b}_1} = \frac{\widehat{\mathbf{v}_1}^T}{\|\widehat{\mathbf{v}_1}\|^2}, \tag{12}$$

where $\widehat{\mathbf{b}_1}$ represents the estimation of the first row $\mathbf{b}_1$ of $\mathbf{B}$. Note that $\mathbf{Y}$ in Algorithm 4 is constructed by $2nm$ traces obtained by power analysis attack, but the proposed OLS attack only needs $\mathbf{y}'_1$ to recover $\mathbf{b}_1$ and hence the number of power analysis attacks (or power traces) is reduced from $2nm$ to m. Theorem 5 proves that $\widehat{\mathbf{b}_1}$ in Eq. (12) is a consistent estimator.

Theorem 5 (Consistent Estimator $\widehat{\mathbf{b}_1}$ in Eq. (12)). *Let* $\mathbf{B}$ *be the secret basis of Falcon, and* $W_{\chi,D}(\tilde{\mathbf{B}})$ *be the probabilistic algorithm whose outputs* $(\mathbf{s}, \bar{\mathbf{z}})$ *satisfy Eq.* (9)*. Here,* χ *and* D *are distribution of* r_i *and discrete Gaussian distribution, respectively. From* m *independently measured samples* $(\mathbf{s}, \bar{\mathbf{z}}) \leftarrow W_{\chi,D}(\tilde{\mathbf{B}})$*,* *the MLR model in Eq. (6) for Falcon is constructed using* $\mathbf{X}$ *and* $\mathbf{Y}$ *consisting of* m *row vectors* $\mathbf{x}_\mathrm{i} = \mathbf{s}_\mathrm{i}$ *and* $\mathbf{y}_\mathrm{i} = \boldsymbol{\lambda}_\mathrm{i} - \bar{\mathbf{z}}_\mathrm{i}$*, respectively. As* m *goes to infinity, the estimator* $\widehat{\mathbf{b}_1} = \frac{1}{k}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}'_1)^T$ *converges to the first row* $\mathbf{b}_1$ *of the secret basis* $\mathbf{B}$ *in probability, where* $\mathbf{y}'_{1,m} \in \mathbb{Q}^m$ *is the first column of* $\mathbf{Y}$ *consisting of* m *elements* $\lambda_1 - \bar{z}_{i,1}$ *and* $k = \|(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}'_1\|^2$*, i.e.,* $\mathrm{plim}_{m\to\infty}\widehat{\mathbf{b}_1} = \mathbf{b}_1$.

The complete proof of Theorem 5 is provided in Supplementary Material A. Theorem 5 implies that a sufficient number of samples $(\mathbf{s}, \bar{z}_1)$ accurately recover $\mathbf{b}_1 = (g, -f)$. Therefore, the secret basis $\mathbf{B}$ of Falcon can be efficiently and accurately recovered using $\bar{z}_1$. Given Falcon-n and m samples, the time complexity for computing $\widehat{\mathbf{b}_1}$ in Eq. (12) is

$$\mathcal{O}(mn + n^3).$$

It is because computing $\mathbf{X}^T\mathbf{y}'_1$, $\mathbf{X}^T\mathbf{X}$, and $(\mathbf{X}^T\mathbf{X})^{-1}$ take $\mathcal{O}(mn)$, $\mathcal{O}(mn)$, and $\mathcal{O}(n^3)$ operations, respectively. Also, scaling by L2 norm takes $\mathcal{O}(n)$ and hence the total time complexity for computing $\widehat{\mathbf{b}_1}$ is $\mathcal{O}(mn + n^3)$. Note that, in Eq. (11), $\mathbf{X}$ consists of m row vectors $\mathbf{x} = \mathbf{s}$ and $\mathbf{y}'_1$ consists of m elements $\frac{1}{2} - \bar{z}_1$.

4.3 Error-tolerant Power Analysis Attack on BaseSampler

This section proposes an error-tolerant power analysis attack on BaseSampler when z_0 is estimated by using a binary classifier for $z^+ \in \{0, 1\}$. The BaseSampler in Algorithm 3 performs 18 iterations of calculating $z_0 \leftarrow z_0 + z^+$ using sample $z_0 \sim D_{\mathbb{Z}^+,\sigma_{max}}$. Let $\mathbf{z}^+ = (z^+_1, z^+_2, \ldots, z^+_{18})$ be the vector where $z^+_i \in \{0, 1\}$ is obtained from the ith loop. Note that the output z_0 of BaseSampler is equal to $\|\mathbf{z}^+\|_1$. The set C of all possible 19 vectors $\mathbf{z}^+$ is as follows:

$$C = \{\mathbf{z}^+ = (z^+_1, z^+_2, \ldots, z^+_{18}) \in \{0, 1\}^{18} \mid z^+_i \in \{0, 1\} \text{ and } z^+_i \geq z^+_j \text{ for } i < j\}.$$

An underflow, which results in at least 8 bits to flip in the 32-bit register, occurs in each loop when $z^+_i = 1$ in the BaseSampler. Such underflow allows binary classification for $z^+_i \in \{0, 1\}$ by observing power trace [15]. Let $\mathcal{A}^{(i)}_{binary}$ be a binary classifier that takes the power trace of the ith loop as input and classifies z^+_i. Then, $\mathcal{A}^{(i)}_{binary}$ estimates $\mathbf{z}^+ = (z^+_1, z^+_2, \ldots, z^+_{18})$ as follows:

$$\hat{\mathbf{c}} = (\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_{18}),\ \hat{c}_i \leftarrow \mathcal{A}^{(i)}_{binary}(\mathbf{Tr}_i) \in \{0, 1\},$$

where $\hat{\mathbf{c}} \in \{0, 1\}^{18}$ is the estimated $\mathbf{z}^+$ and $\mathbf{Tr}_i$ is the power trace of the ith loop. If $\mathcal{A}_{binary}$ fails to correctly classify the value of z^+_i, then $\hat{\mathbf{c}}$ may not belong to C such that

$$\hat{\mathbf{c}} = \mathbf{z}^+ + \mathbf{e}^+ \mod 2,$$

where $\mathbf{z}^+ \in C$ and $\mathbf{e}^+ = (e^+_1, e^+_2, \ldots, e^+_{18}) \in \{0, 1\}^{18}$ is an error vector. Assume that the errors e^+_i, $i \in \{1, 2, \ldots, 18\}$, are statistically independent. For given $\hat{\mathbf{c}} \in \{0, 1\}^{18}$, for each $\mathbf{c} \in C$, define $I_{\mathbf{c},\hat{\mathbf{c}}} := \{i \in \{1, 2, \ldots, 18\} \mid e^+_i = 1 \text{ for } \mathbf{e}^+ =$

$\mathbf{c} + \hat{\mathbf{c}} \mod 2\}$ and $J_{\mathbf{c},\hat{\mathbf{c}}} := \{1, 2, \ldots, 18\} \setminus I_{\mathbf{c},\hat{\mathbf{c}}}$. Then, the following maximum a posteriori (MAP) decoding is proposed to correct the errors in $\hat{\mathbf{c}}$ as follows:

$$\begin{aligned}
\mathbf{c}^* &= \arg\max_{\mathbf{c}\in C} \mathbb{P}\left(\mathbf{c} \mid \hat{\mathbf{c}}\right) \\
&= \arg\max_{\mathbf{c}\in C} \mathbb{P}(\mathbf{c})\mathbb{P}\left(\hat{\mathbf{c}} \mid \mathbf{c}\right) \\
&= \arg\max_{\mathbf{c}\in C} \mathbb{P}(\mathbf{c})\mathbb{P}(\mathbf{e}^+) \\
&= \arg\max_{\mathbf{c}\in C} \mathbb{P}(\mathbf{c}) \prod_{i\in I_{\mathbf{c},\hat{\mathbf{c}}}} \mathbb{P}(e_i^+ = 1) \prod_{j\in J_{\mathbf{c},\hat{\mathbf{c}}}} \mathbb{P}(e_j^+ = 0), \qquad \text{--------} \ (13)
\end{aligned}$$

where $\mathbf{c}^* \in C$ is the corrected $\hat{\mathbf{c}}$ by using MAP decoding. Since the probability of selecting z_0 in BaseSampler is predetermined by the RCDT table [30] and z_0 is uniquely determined by the codewords $\mathbf{c}$ of the set C such that $z_0 = \|\mathbf{c}\|_1$, the probability of selecting $\mathbf{c}$ from C is the same as the probability of selecting $z_0 \in \{0, 1, \ldots, 18\}$, i.e., $\mathbb{P}(\mathbf{c}) = \mathbb{P}(z_0)$. Therefore, after performing MAP decoding in Eq. (15) the output z_0 of BaseSampler is estimated uniquely as follows:

$$\widehat{z_0} = \|\mathbf{c}^*\|_1.$$

Even for the case that the errors are not statistically independent, the MAP decoding approach in Eq. (15) can be taken to correct the errors in $\hat{\mathbf{c}}$.

5 Experimental Evaluation

This section presents the experimental evaluation of deep-learning-based SPA attack to collect samples and the proposed OLS attack using these samples, as discussed in Section 4. All evaluations were conducted on Ubuntu 20.04.6 LTS with an Intel i7-7700k (4.2GHz), 16GB DDR4 RAM, and an NVIDIA GeForce GTX 970. The operation for secret key recovery in the experiment was not optimized. Hence, the processing time could be further reduced through enhanced CPUs, single instruction multiple data (SIMD), and so on.

5.1 Simple Power Analysis Attacks on SamplerZ with Noise

Note that $\bar{z}$ in Algorithm 2 is $\bar{z} = b + (2b - 1) \cdot z_0$ for the BaseSampler output z_0 and the sign value b. The value of $\bar{z}$ is estimated by z_0 and b, obtained through power analysis attack. Since discrete Gaussian samplers play a crucial role in PQC including Falcon, numerous studies have already explored SPA targeting discrete Gaussian samplers using CDT and RCDT to estimate the leakage of z_0 and b [15,19,21,35,37]. Although most studies do not consider noisy environments when estimating z_0 and b, estimating z_0 and b under various noise conditions is necessary for practical reasons. Therefore, we focus on the realistic power analysis attack for estimating z_0 and b from noisy power traces. Moreover, we evaluate the attack capability when the errors in z_0 are corrected by the MAP decoding proposed in Section 4.3.

The power traces were collected using ChipWhisperer and the ELMO simulator, targeting the ARM Cortex-M4 processor. The source code to be implemented on the processor is the recent Falcon reference code [2] in NIST. For ChipWhisperer, we mounted an STM32F415 on a CW308 UFO target board and measrued power traces using ChipWhisperer Lite. Detailed setup for ChipWhisperer can be found in [35]. The ELMO simulator emulates noise-free power traces targeting the ARM Cortex-M0/M4 model [23]. To simulate various noise conditions, the noisy power traces were generated by adding white Gaussian noise to the noise-free power traces produced by the ELMO simulator. A measure of signal quality is determined by the signal-to-noise ratio (SNR) defined as

$$SNR_{dB} := 10 \log \left(\frac{P_{signal}}{P_{noise}} \right),$$

where P_{signal} is the signal power and P_{noise} is the noise power.

A multi-layer perceptron (MLP) is used as a binary classifier to estimate z_0 and b using noisy power traces as input. The MAP decoding scheme in Sec. 4.3 can operate with any binary classifier, and a SPA-based MLP was selected since it is good enough to demonstrate the performance of MAP decoding. The MAP decoding scheme is still effective even when employing the SPA proposed in [15] and [21]. Additionally, the MLP can distinguish the underflow feature in the power traces and detect power differences at the assembly instruction level [35], allowing efficient classification of z_0 and b. The structure and hyperparameters of the MLP used in the experiment are presented in Supplementary Material B.

Half Gaussian Leakage with Noise. The MLP serves as a binary classifier $\mathcal{A}^{(i)}_{binary}$ described in Section 4.3, which takes the power trace $\mathbf{Tr}_i$ as input and classifies z_i^+ as 0 or 1 for $i \in \{1, 2, \ldots, 18\}$, where i is the iteration number of the for-loop in the BaseSampler. Note that the training data consists of noisy power traces $\mathbf{Tr}_i$ each labeled with z_i^+. For ChipWhisperer, the performance of $\mathcal{A}^{(i)}_{binary}$ was evaluated using 10,000 training data and a separate test data of 10,000 samples for each iteration i. Similarly, for ELMO, the performance was evaluated using 10,000 training data and 10,000 test data for each iteration i and each SNR levels [45dB, 40dB, 35dB, 30dB, 25dB]. When training an MLP with noise-free (SNR ∞ dB) training data, around 1,000 training samples per iteration are sufficient to achieve 100% classification accuracy. However, in the experiment, we trained $\mathcal{A}^{(i)}_{binary}$ with 10,000 data per iteration to guarantee the MLP's performance with a sufficient amount of training data. Table 1 shows the accuracy of the MLP $\mathcal{A}^{(i)}_{binary}$ for the ith loop on each platforms under various SNR.

As shown in Table 1, for ChipWhisperer, the third iteration show the worst accuracy of $\mathcal{A}^{(i)}_{binary}$, while the remaining iterations consistently achieve high accuracy. For ELMO, the first iteration is most affected by the noise, leading to

[2] `https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022`

Table 1. Accuracy of the MLP $\mathcal{A}^{(i)}_{binary}$ for estimating z_i^+ for various platforms and SNR.

Platform	**SNR (dB)**	**Accuracy of $\mathcal{A}^{(i)}_{binary}$**					
		$i = \mathbf{1}$	$i = \mathbf{2}$	$i = \mathbf{3}$	$i = \mathbf{4}$	$i = \mathbf{5}$	$i \geq \mathbf{6}$
ChipWhisperer	-	0.9932	0.9824	0.9698	0.9900	0.9883	0.9999
ELMO	45	0.9988	0.9988	0.9932	0.9944	0.9932	0.9999
ELMO	40	0.9916	0.9956	0.9796	0.9964	0.9912	0.9998
ELMO	35	0.9716	0.9936	0.9428	0.9876	0.9912	0.9998
ELMO	30	0.8952	0.9676	0.8964	0.982	0.9912	0.9998
ELMO	25	0.7632	0.9000	0.8432	0.9632	0.9892	0.9998

a sharp decline in accuracy as SNR decreases, followed by the third iteration. The remaining iterations are progressively less affected by the noise as the iteration continues. These experimental results of both ELMO and ChipWhisperer show that the underflow feature varies for each loop due to the noise.

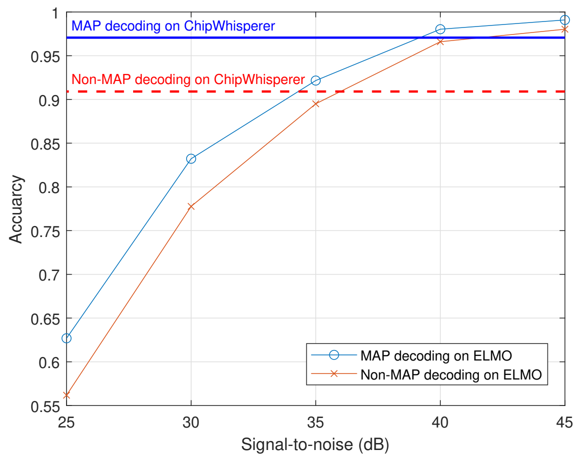

Fig. 1. Accuracy of estimating z_0 by $\mathcal{A}^{(i)}_{binary}$ for various SNR and platforms.

Fig. 1 shows the accuracy of estimating z_0 by the MLP $\mathcal{A}^{(i)}_{binary}$. It shows that MAP decoding consistently provides better accuracy compared to Non-MAP decoding. Specifically, the proposed MAP decoding achieves approximately 12.5% improvement in accuracy over Non-MAP decoding on ELMO at SNR 25dB. When MAP decoding is applied to power traces measured on actual hardware using ChipWhisperer, it further enhances z_0 estimation accuracy compared to ELMO. The average probability over the first six iterations in Table 1 is 0.9873 for ChipWhisperer, and the closest corresponding probability for ELMO is 0.9811 at 35dB. As shown in Fig. 1, the MAP decoding improves accuracy by about 3% for ELMO at 35 dB, whereas ChipWhisperer achieves more than twice that improvement, with a 7% increase in accuracy. Although the average probabilities are similar, ChipWhisperer shows better improvement due to higher probability of correct classification in the first loop. Since BaseSampler samples $z_0 \in \{0, 1, 2, 3\}$ with 95% probability, the accuracy of estimating z_i^+ during the first four iterations significantly influences the performance of MAP decoding. In particular, as the probability of $z_0 = 0$ is 36 %, achieving correct classification of z_i^+ in the first iteration is crucial for MAP decoding performance.

The reason for such relatively modest performance improvement is that the set C of 19 vectors $\mathbf{z}^+$ is not a good error-correcting code, which has a minimum Hamming distance of 1, and the accuracy of the MLP model deteriorates mostly due to the noise in the first loop as in Table 1 for ELMO. However, such improvement with MAP decoding can substantially improve the performance of the OLS attack, as discussed in Section 5.2.

Sign Leakage with Noise. As demonstrated in [37], the sign b was estimated using the power traces generated during the execution of two instruction codes in SamplerZ. The first instruction code generates $b \in \{0, 1\}$ uniformly at random and the second instruction code performs $b + (2 \cdot b - 1) \cdot z_0$. The power traces generated during the execution of $b + (2 \cdot b - 1) \cdot z_0$ show significant difference depending on b. Because the register for b changes from 0x00000000 to 0xFFFFFFFF by an underflow occurred when $b = 0$, this significant power difference makes it easy to distinguish even with the noise. Specifically, even at SNR 0 dB in ELMO, the MLP trained with only 300 samples can perfectly classify the sign value b. Similarly, for ChipWhisperer, training with 300 samples or even fewer also achieves 100% accuracy in classifying b. Therefore, estimating the sign b from noisy power traces does not require noise reduction scheme.

5.2 Experimental Results of the OLS Attack on Falcon

Performance of the OLS Attack on Falcon using Noise-free Samples. The experiment focuses on the performance of the proposed OLS attack on Falcon using samples $\bar{\mathbf{z}}$ estimated without noise. The setup is conducted in a noise-free environment to maintain consistency with pervious works [15] and [37], allowing for fair performance comparison. The impact of actual hardware errors and various noise conditions on OLS performance is separately discussed later in this section.

Twenty instances of secret key $\mathbf{b}_1$ for each of Falcon-256, 512, and 1024 are used for the experiment. We measured the error size and computation CPU time of the OLS attack using up to 60,000 $(\mathbf{s}, \bar{\mathbf{z}})$ samples per instance. Note that the secret basis $\mathbf{B}$ of Falcon can be recovered by estimating the first row $\mathbf{b}_1$ of $\mathbf{B}$, as described in Section 4.1. In the experiment, the similarities between each of two secret key estimators $\widehat{\mathbf{b}_1}$ and $\widehat{\mathbf{a}_1}$, and the real $\mathbf{b}_1$ are evaluated for various number of samples, where $\widehat{\mathbf{b}_1}$ is the estimator in Eq. (12) and $\widehat{\mathbf{a}_1}$ is the first row vector of $\widehat{\mathbf{A}}$ in Eq. (8). The error size of an estimator $\widehat{\mathbf{x}}$ is defined as $\|\mathbf{b}_1 - \lfloor\widehat{\mathbf{x}}\rceil\|_1$, where $\lfloor\widehat{\mathbf{x}}\rceil$ represents the rounding of all coefficients of $\widehat{\mathbf{x}}$.

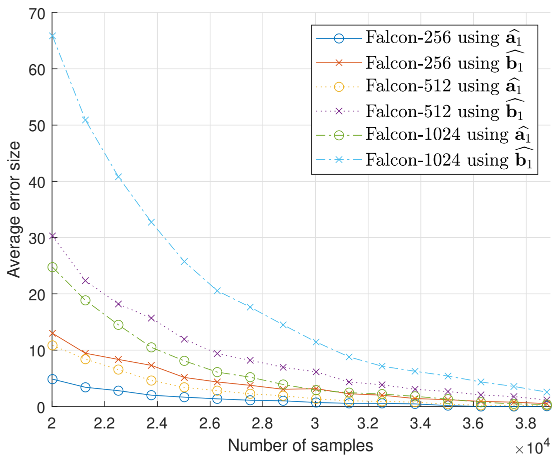

Fig. 2. The average error size of 20 instances for the estimators $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$ on Falcon-256, 512, 1024.

Fig. 2 shows the average error size of 20 instances for $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$ with various number of samples. As the number of samples increases, both $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}}_1$ converge to $\mathbf{b}_1$ as proven in Theorems 4 and 5. The estimators $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}}_1$ require 33,700 and 46,200 samples, respectively, to directly recover $\mathbf{b}_1$ of Falcon-512 with 25% success rate. For the same number of samples, $\widehat{\mathbf{b}_1}$ shows larger estimation error compared to $\widehat{\mathbf{a}_1}$. It is because the estimator $\widehat{\mathbf{A}}$ is the inverse of the OLS estimator $\widehat{\mathbf{A}^{-1}}$, which can be precisely computed in a single step using $(\mathbf{X}^T\mathbf{Y})^{-1}\mathbf{X}^T\mathbf{X}$ but the computation of $\widehat{\mathbf{b}_1}$ requires two steps of OLS regression and scaling by the L2 norm, which leads to bigger errors. The number of samples required to directly recover the secret basis $\mathbf{B}$ for various n is presented in Table 2. As shown in Table 2, $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$ exhibit a trade-off between the required number of samples (or execution time) and the success rate.

Fig. 3 shows the average CPU time of computing $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$ for estimating 20 instances with various number of samples. Since the time complexities of two estimators $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$ are $\mathcal{O}(mn^2 + n^3)$ and $\mathcal{O}(mn + n^3)$, we can see that $\widehat{\mathbf{b}_1}$ is more efficient. However, as shown in Fig. 3, for Falcon-256 and 512, the average CPU time for computing $\widehat{\mathbf{b}_1}$ is similar to that for $\widehat{\mathbf{a}_1}$. For Falcon-1024, the average computation time of $\widehat{\mathbf{b}_1}$ is smaller than that of $\widehat{\mathbf{a}_1}$ by about 1.5 seconds. Also the computation time of two proposed estimators increases linearly with the number

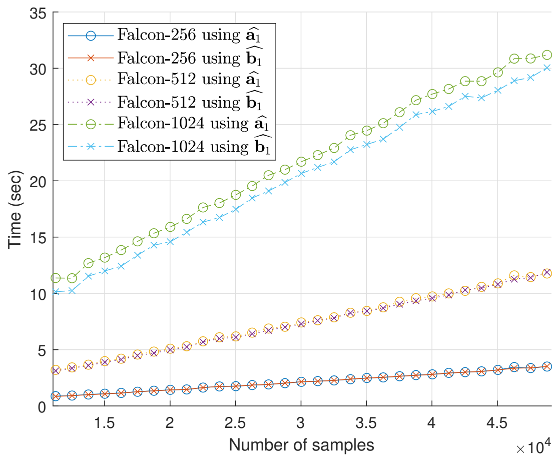

Fig. 3. The average CPU time of 20 instances over the number of samples for computing $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$.

of samples and exponentially with n. The time required to directly recover $\mathbf{b}_1$ from $\widehat{\mathbf{a}_1}$ and $\widehat{\mathbf{b}_1}$ for various n is presented in Table 2.

Table 2. The average number of samples and time required for achieving particular success rate (SR) to recover the secret key $\mathbf{b}_1$ of Falcon-256, 512, and 1024 by the OLS attack.

	SR (%)	$\widehat{\mathbf{a}_1}$		$\widehat{\mathbf{b}_1}$	
		#Samples	**Time (sec)**	**#Samples**	**Time (sec)**
Falcon-256	25	30,000	1.73	37,500	2.63
	60	31,200	1.75	43,700	3.06
	90	35,000	2.48	49,500	3.64
Falcon-512	25	33,700	8.27	46,200	11.25
	60	35,000	8.44	48,900	12.02
	90	40,000	9.72	51,800	12.81
Falcon-1024	25	36,000	25.12	46,800	29.03
	60	37,000	26.09	49,000	30.14
	90	42,500	28.85	53,600	32.45

Approximation Scheme. For a small number of samples, the OLS attack on Falcon can provide an estimator $\widehat{\mathbf{x}}$ for approximating $\mathbf{b}_1$, even if it is not very close to $\mathbf{b}_1$. Since $\mathbf{b}_1$ can be recovered from $\widehat{\mathbf{x}}$ using lattice reduction when a sufficiently close vector $\widehat{\mathbf{x}}$ is available, the OLS attack combined with the lattice

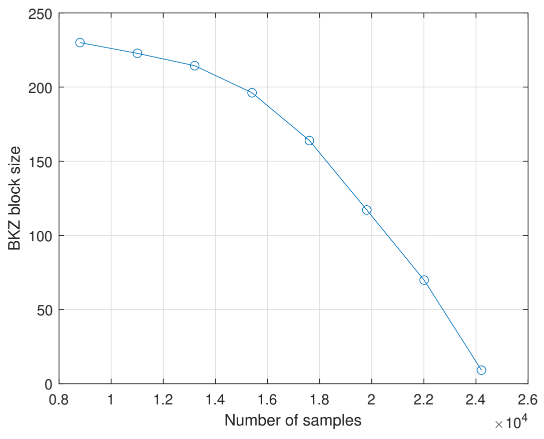

Fig. 4. The BKZ block size of Falcon-512 estimated by the leaky-LWE estimator given the coefficients of $\widehat{\mathbf{a}_1}$ as hints based on the number of samples.

reduction provides a trade-off between the number of samples for OLS attack and the computational time for lattice reduction. Fig. 4 shows the BKZ block size estimated by the leaky-LWE estimator [5] for recovering $\mathbf{b}_1$ of Falcon-512 using each coefficient of $\widehat{\mathbf{a}_1}$ as an approximate hint. The BKZ block size initially required to recover $\mathbf{b}_1$ in Falcon-512 is around 480 and drops below 200 when over 15,000 samples are used to estimate $\widehat{\mathbf{a}_1}$ by the OLS attack.

An attack scheme in [22] is based on the DDGR attack [5], which easily integrates perfect hints, modulus hints, and approximate hints. For 10 instances of Falcon-512 by using $\widehat{\mathbf{a}_1}$ as a hint, which is provided by the OLS attack using 25,000 samples, the lattice-reduction scheme in [22] can successfully recover the secret key with 100% success rate within 2.5 hours. Specifically, the coefficients of $\widehat{\mathbf{a}_1}$ with a decimal value between -0.3 and 0.3 are used as perfect hints, and other coefficients are used as approximate hints.

Comparison with the State-of-the-art Attacks. Since the OLS attack exploits Gaussian leakage to recover the secret basis of Falcon, this study compares the performance of OLS attack with the state-of-the-art attacks that also utilize Gaussian leakage [15,37]. For Falcon-512, the number of samples and the computational time required to recover the secret basis $\mathbf{B}$ are compared for the OLS attack and the state-of-the-art attacks in [15] and [37]. The scheme in [15] solves the hidden parallelepiped problem for recovering $\mathbf{b}_1$ using the samples $(\mathbf{s}, z_0 = 0)$, where $z_0 \in \{0, 1, \ldots, 18\}$ is the output of BaseSampler in Algorithm 3. It was shown that this scheme can recover $\mathbf{b}_1$ by combining DDGR attack with 1 million samples and 25 hours of CPU time, whereas 10 million samples are needed for direct recovery of $\mathbf{b}_1$ only by this scheme. The secret key recovery attack in [37] solves the learning slice problem to recover $\mathbf{b}_1$ using the samples $(\mathbf{s}, \bar{z}_1 = 0)$, where $\bar{z}_1 \in \{-17, -16, \ldots, 18\}$ is calculated in SamplerZ. This scheme with an exhaustive search takes 30 minutes of CPU time and 45,000 samples to recover the secret key with 25 % success rate. These two schemes recover $\mathbf{b}_1$ from a conditional signature distribution, implying that only a portion of measured samples are used to recover $\mathbf{b}_1$. For Falcon-512, the standard deviation of coefficient of signature $\mathbf{s}$ is given by $\sigma' = 1.2915$. Consequently, the probability of obtaining $\mathbf{s}$ conditioned on $z_0 = 0$ is $\mathbb{P}(\mathbf{s}|z_0 = 0) \approx \text{erf}\left(\frac{1}{\sqrt{2}\sigma'}\right) = 0.5612$, where erf is the Gaussian error function. In the hidden parallelepiped scheme, to obtain 1 million filtered signatures with $z_0 = 0$, an average of 1.78 million signature samples must be collected by the side-channel attack. In the learning slice scheme, the probability of sampling $\bar{z}_1 = 0$ is 0.2806. Therefore, in order to obtain 45,000 filtered signatures with $\bar{z}_1 = 0$, 161,000 samples should be collected by the side-channel attack. Unlike the state-of-the-art attacks, the proposed OLS attack recover $\mathbf{b}_1$ using all measured samples $(\mathbf{s}, \bar{\mathbf{z}})$ and $(\mathbf{s}, \bar{z}_1)$.

Table 3 shows the required sample size and time to recover $\mathbf{b}_1$ of Falcon-512 using the estimated $\bar{\mathbf{z}}$ without noise for various attacks. As shown in Table 3, the OLS attack requires fewest samples and a significantly short computation time compared to the other schemes. Furthermore, if the DDGR attack is combined, the OLS attack recovers $\mathbf{b}_1$ of Falcon-512 using 25,000 samples with 100 % success rate for ten instances. When performing side-channel attack on Falcon, about 25,000 signatures are required for SPA to successfully recover the secret key. This number is significantly smaller than that of the state-of-the-art attack, which requires 161,000 signatures for SPA, demonstrating the superior efficiency and strength of the proposed attack scheme.

Table 3. Performance comparison of various attack schemes on Falcon-512. 'SR' denotes success rates. 'HP' and 'LS' denote the hidden parallelepiped scheme and the learning slice scheme, respectively. 'Approximation' refers to the scheme to recover the secret key from the approximated secret key. 'Filtered' refers to the number of filtered signatures used in the secret key recovery attack. 'Total' refers the average number of signatures required for obtaining filtered signatures using SPA.

	Approximation	**#Samples** ($\times 10^4$) **(Filtered/Total)**	**Time**	**SR** (%)
HP [15]	Rounding	1,000 / 1,780	-	-
	DDGR [5]	100 / 178	25 hours	-
LS [37]	Rounding	7 / 25	-	25
	Exhaustive Search	4.5 / 16.1	30 min	25
OLS	Rounding	3.5 / 3.5	8.4 sec	60
	DDGR [22]	2.5 / 2.5	2.5 hours	100

Performance of the OLS Attack on Falcon using Noisy Samples. Experimental results are provided when noisy z_0's are used, where the noise is generated by reflecting the MLP accuracy in Table 1 (30dB and ChipWhisperer) as the error probability of each iteration in BaseSampler. Specifically, each error e_i of the ith loop in BaseSampler is independently generated, and the error probability is assumed to be $\mathbb{P}(e_i = 1) = \mathbb{P}(\hat{c}_i = 1 \mid c_i = 0) = \mathbb{P}(\hat{c}_i = 0 \mid c_i = 1)$. The accuracy of the MLP classifier of z^+ for the ith loop is $1 - \mathbb{P}(e_i = 0)$ and hence the error vector $\mathbf{e}^+ = (e_1, e_2, \ldots, e_{18})$ can be generated based on the MLP accuracy Table 1 (30dB and ChipWhisperer). As described in Section 4.3, if we measure $\hat{\mathbf{c}} = \mathbf{z}^+ + \mathbf{e}^+$ by power analysis attack for $\mathbf{z}^+ \in C$ and the MAP decoding is used, $\hat{\mathbf{c}}$ is decoded into $\mathbf{c}^* \in C$, and $\hat{z_0}$ is estimated as $\|\mathbf{c}^*\|_1$. For Non-MAP decoding, $\mathbf{c}^* = \hat{\mathbf{c}}$, which may not be in C, and $\hat{z_0}$ is estimated as $\|\mathbf{c}^*\|_1$. Fig. 5 shows the error size in $\widehat{\mathbf{a}_1}$ obtained by the OLS attack for a single instance of Falcon-512 using the estimated $\hat{z_0}$.

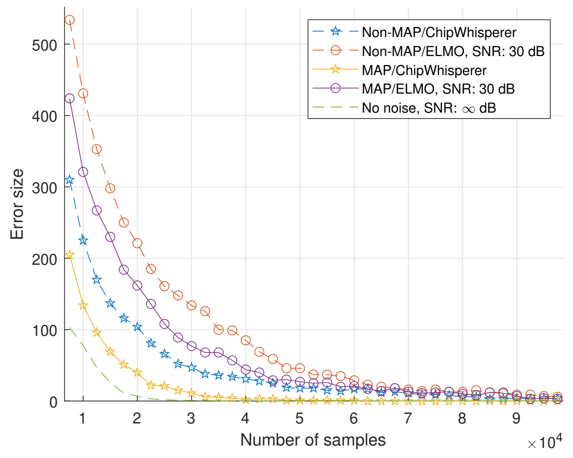

Fig. 5. The error size of the OLS estimator $\widehat{\mathbf{a}_1}$ for Falcon-512 using the noisy z_0. 'No noise' refers to the case when the power trace of z_0 does not contain noise.

As shown in Fig. 5, the number of samples required for the OLS attack to recover the secret key increases with the number of noisy samples. However, even when the accuracy of estimating z_0 is 77.5% at 30dB, the OLS attack can recover the secret key only within 100,000 samples. As demonstrated by Theorem 4, the experimental results in Fig. 5 suggests that the OLS attack using noisy samples still converges to successfully recover the secret key in probability. In other words, the OLS attack is an error-tolerant attack scheme against any errors occurring during side-channel attack. Also despite the presence of noisy samples, the OLS attack requires fewer samples than the state-of-the-art attacks. In particular, when MAP decoding is applied to the power traces from ChipWhisperer, the

OLS attack can recover the secret key only with 47,000 samples, whereas Non-MAP decoding requires at least 85,000 samples, demonstrating the effectiveness of MAP decoding in real hardware.

If MAP decoding is used, the error size is consistently smaller than that without MAP decoding. Notably, in both ChipWhisperer and ELMO at SNR 30dB, the error size gap between MAP decoding and Non-MAP decoding is approximately 50 at 30,000 samples. This suggests that combining the DDGR scheme and MAP decoding can further reduce the required number of samples and computational time for recovering the secret key.

6 Conclusion

We first derived vulnerable partial information in GPV framework as in Eq. (3) and proposed an efficient secret key recovery attack, called the OLS attack, which effectively leverages partial information to recover the secret key of GPV signatures without filtering. Interestingly, the OLS attack is shown to be a linear consistent estimator, which implies that it can efficiently recover the secret key given a sufficient number of samples.

If we apply the OLS attack to Falcon, it becomes the most powerful and practical such that it requires the fewest samples and significantly less time compared to the state-of-the-art attacks using Gaussian leakage [15], [37]. Unlike the previous works, which use only a part of the samples, the OLS attack utilizes all the samples. In addition, we proposed more efficient OLS attack by using the property of NTRU lattice, which further reduces the number of side-channel attacks (or power traces or samples). We also proposed an error-tolerant power analysis attack based on MAP decoding to improve the quality of samples obtained by side-channel attack.

Although this paper primarily focuses on Falcon, the OLS attack is not limited to Falcon and can be applied to any GPV signatures. For example, Mitaka signature [8] employs a hybrid sampler that can be viewed as a perturbed version of the Klein-GPV algorithm. If Gaussian leakage is used as partial information, the OLS attack can similarly recover the secret key, like Falcon. Moreover, the OLS attack recovers the secret key only by using the first row of the matrix $\widehat{\mathbf{A}}$ in Eq. (8). If the OLS attack is extended to leverage all information of $\widehat{\mathbf{A}}$, the secret key may be recovered using fewer samples. Given these possibilities for further development, the OLS paves a path for significant research potential.

References

1. Alagic, G., Alagic, G., Apon, D., Cooper, D., Dang, Q., Dang, T., Kelsey, J., Lichtinger, J., Liu, Y.K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Smith-Tone, D.: Status report on the third round of the nist post-quantum cryptography standardization process (2022)
2. Bernstein, D.J., Hopwood, D., Hülsing, A., Lange, T., Niederhagen, R., Papachristodoulou, L., Schneider, M., Schwabe, P., Wilcox-O'Hearn, Z.: Sphincs:

practical stateless hash-based signatures. In: EUROCRYPT 2015. pp. 368–397. Springer (2015). https://doi.org/10.1007/978-3-662-46800-5_15
3. Billingsley, P.: Convergence of probability measures. John Wiley & Sons (2013)
4. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: ASIACRYPT 2011. pp. 41–69. Springer (2011). https://doi.org/10.1007/978-3-642-25385-0_3
5. Dachman-Soled, D., Ducas, L., Gong, H., Rossi, M.: Lwe with side information: Attacks and concrete security estimation. In: CRYPTO 2020. pp. 329–358. Springer (2020). https://doi.org/10.1007/978-3-030-56880-1_12
6. Ducas, L., Prest, T.: A hybrid gaussian sampler for lattices over rings. IACR Cryptology ePrint Archive p. 660 (2015), https://ia.cr/2015/660
7. Ducas, L., Prest, T.: Fast fourier orthogonalization. In: Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation. pp. 191–198 (2016). https://doi.org/10.1145/2930889.2930923
8. Espitau, T., Fouque, P.A., Gérard, F., Rossi, M., Takahashi, A., Tibouchi, M., Wallet, A., Yu, Y.: Mitaka: a simpler, parallelizable, maskable variant of falcon. In: EUROCRYPT 2022. pp. 222–253. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_9
9. Ferguson, T.: A course in large sample theory. Routledge (2017)
10. Fouque, P.A., Kirchner, P., Tibouchi, M., Wallet, A., Yu, Y.: Key recovery from gram–schmidt norm leakage in hash-and-sign signatures over ntru lattices. In: EUROCRYPT 2020. pp. 34–63. Springer (2020). https://doi.org/10.1007/978-3-030-45727-3_2
11. Galambos, J.: Advanced probability theory, vol. 10. CRC Press (1995)
12. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC). pp. 197–206 (2008). https://doi.org/10.1145/1374376.1374407
13. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: CRYPTO '97. Springer (1997). https://doi.org/10.1007/BFb0052231
14. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996). https://doi.org/10.1145/237814.237866
15. Guerreau, M., Martinelli, A., Ricosset, T., Rossi, M.: The hidden parallelepiped is back again: Power analysis attacks on falcon. IACR Transactions on Cryptographic Hardware and Embedded Systems **2022**(3), 141–164 (2022). https://doi.org/10.46586/tches.v2022.i3.141-164
16. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J., Whyte, W.: Ntrusign: Digital signatures using the ntru lattice. In: Cryptographers' track at the RSA conference. pp. 122–140. Springer (2003). https://doi.org/10.1007/3-540-36563-x_9
17. Hoffstein, J., Pipher, J., Silverman, J.H.: Ntru: A ring-based public key cryptosystem. In: International Algorithmic Number Theory Symposium. pp. 267–288. Springer (1998). https://doi.org/10.1007/BFb0054868
18. Karabulut, E., Aysu, A.: Falcon down: Breaking falcon post-quantum signature scheme through side-channel attacks. In: 2021 58th ACM/IEEE Design Automation Conference (DAC). pp. 691–696. IEEE (2021). https://doi.org/10.1109/DAC18074.2021.9586131

19. Kim, S., Hong, S.: Single trace analysis on constant time cdt sampler and its countermeasure. Applied Sciences **8**(10), 1809 (2018). https://doi.org/10.3390/app8101809
20. Lyubashevsky, V., Ducas, L., Kiltz, E., Lepoint, T., Schwabe, P., Seiler, G., Stehlé, D., Bai, S.: Crystals-dilithium. Algorithm Specifications and Supporting Documentation (2020)
21. Marzougui, S., Wisiol, N., Gersch, P., Krämer, J., Seifert, J.P.: Machine-learning side-channel attacks on the galactics constant-time implementation of bliss. In: Proceedings of the 17th International Conference on Availability, Reliability and Security. pp. 1–11 (2022). https://doi.org/10.1145/3538969.3538980
22. May, A., Nowakowski, J.: Too many hints – when lll breaks lwe. In: ASIACRYPT 2023. pp. 106–137. Springer (2023). https://doi.org/10.1007/978-981-99-8730-6_4
23. McCann, D., Oswald, E., Whitnall, C.: Towards practical tools for side channel aware software engineering: 'grey box' modelling for instruction leakages. In: 26th USENIX Security Symposium (USENIX Security 17). pp. 199–216 (2017). https://doi.org/10.5555/3241189.3241207
24. Migliore, V., Gérard, B., Tibouchi, M., Fouque, P.A.: Masking dilithium: Efficient implementation and side-channel evaluation. In: Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17. pp. 344–362. Springer (2019). https://doi.org/10.1007/978-3-030-21568-2_17
25. Nguyen, P., Regev, O.: Learning a parallelepiped: Cryptanalysis of ggh and ntru signatures. In: EUROCRYPT 2006. pp. 271–288. Springer (2006). https://doi.org/10.1007/11761679_17
26. Peikert, C.: An efficient and parallel gaussian sampler for lattices. In: CRYPTO 2010. pp. 80–97. Springer (2010). https://doi.org/10.1007/978-3-642-14623-7_5
27. Pellet-Mary, A., Stehlé, D.: On the hardness of the ntru problem. In: ASIACRYPT 2021. pp. 3–35. Springer (2021). https://doi.org/10.1007/978-3-030-92062-3_1
28. Pornin, T., Prest, T.: More efficient algorithms for the ntru key generation using the field norm. In: PKC 2019. pp. 504–533. Springer (2019). https://doi.org/10.1007/978-3-030-17259-6_17
29. Prest, T.: Gaussian sampling in lattice-based cryptography. Ph.D. thesis, Ecole normale supérieure-ENS PARIS (2015)
30. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: Falcon. Post-Quantum Cryptography Project of NIST (2020)
31. Rossi, T., Prest, T., Ricosset, M.: Simple, fast and constant-time gaussian sampling over the integers for falcon. Post-Quantum Cryptography Project of NIST (2019)
32. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. pp. 124–134. Ieee (1994). https://doi.org/10.1109/SFCS.1994.365700
33. Stehlé, D., Steinfeld, R.: Making ntru as secure as worst-case problems over ideal lattices. In: EUROCRYPT 2011. pp. 27–47. Springer (2011). https://doi.org/10.1007/978-3-642-20465-4_4
34. Stewart, G.W.: On the continuity of the generalized inverse. SIAM Journal on Applied Mathematics **17**(1), 33–45 (1969). https://doi.org/10.1137/0117004
35. Wisiol, N., Gersch, P., Seifert, J.P.: Cycle-accurate power side-channel analysis using the chipwhisperer: A case study on gaussian sampling. In: International Conference on Smart Card Research and Advanced Applications. pp. 205–224. Springer (2022). https://doi.org/10.1007/978-3-031-25319-5_11

36. Yu, Y., Jia, H., Li, L., Ran, D., Qiu, Z., Zhang, S., Lin, X., Wang, X.: Hufu: Hash-and-sign signatures from powerful gadgets. Algorithm Specifications and Supporting Documentation (2023)
37. Zhang, S., Lin, X., Yu, Y., Wang, W.: Improved power analysis attacks on falcon. In: EUROCRYPT 2023. pp. 565–595. Springer (2023). https://doi.org/10.1007/978-3-031-30634-1_19

Supplementary Material

A Proofs of Lemmas and Theorems

The proof of Lemma 1 in Section 3.2 is given below.

Lemma 1 (Uncorrelatedness). *Let* $\mathbf{x} = \mathbf{s} \in \mathbb{Z}^n$ *and* $\mathbf{e} = \boldsymbol{\lambda} - \mathbf{r} \in \mathbb{R}^n$ *be the independent input and noise vectors, respectively, in Eq.* (5)*. Then,* $\mathbf{x}$ *and* $\mathbf{e}$ *are uncorrelated, i.e.,* $\mathbf{K_{xe}} = Cov[\mathbf{x}, \mathbf{e}] = \mathbf{0}$.

Proof. In the proposed MLR model $\mathbf{y} = \mathbf{x}\mathbf{A}^{-1} + \mathbf{e}$, for any $\mathbf{x}$, we have

$$\begin{aligned}
\mathbb{E}_{\mathbf{x}}[\mathbf{x}] &= \mathbb{E}_{\mathbf{s}}[\mathbf{s}] && \text{(by the definition of } \mathbf{x}\text{)}\\
&= \mathbb{E}_{\mathbf{r},\mathbf{u}}[(\mathbf{r} - \mathbf{u})\mathbf{A}] && \text{(by Eq. (3))}\\
&= \mathbb{E}_{\mathbf{r}}[\mathbb{E}_{\mathbf{u}|\mathbf{r}}[(\mathbf{r} - \mathbf{u})\mathbf{A} \mid \mathbf{r}]] && \text{(law of total expectation)}\\
&= \mathbb{E}_{\mathbf{r}}[(\mathbf{r} - \mathbb{E}_{\mathbf{u}|\mathbf{r}}[\mathbf{u} \mid \mathbf{r}])\mathbf{A}] && \text{(constant matrix } \mathbf{A}\text{)}\\
&= \mathbb{E}_{\mathbf{r}}[(\mathbf{r} - \mathbf{r})\mathbf{A}] && (\mathbb{E}_{\mathbf{u}|\mathbf{r}}[\mathbf{u}|\mathbf{r}] = \mathbf{r})\\
&= \mathbf{0},
\end{aligned}$$

where $\mathbb{E}_{\mathbf{u}|\mathbf{r}}[\mathbf{u} \,|\, \mathbf{r}] = \mathbf{r}$ holds because u_i in Eq. (3) is distributed according to Q_{σ_i, r_i} for all $i \in \{1, 2, \ldots, n\}$. It follows that

$$\begin{aligned}
\mathbf{K_{xe}} = Cov[\mathbf{x}, \mathbf{e}] &= \mathbb{E}_{\mathbf{x},\mathbf{e}}[\mathbf{x}^T\mathbf{e}] && (\mathbb{E}_{\mathbf{x}}[\mathbf{x}] = \mathbb{E}_{\mathbf{e}}[\mathbf{e}] = \mathbf{0})\\
&= \mathbb{E}_{\mathbf{e}}[\mathbb{E}_{\mathbf{x}|\mathbf{e}}[\mathbf{x} \mid \mathbf{e}]^T\mathbf{e}] && \text{(law of total expectation)}\\
&= \mathbb{E}_{\mathbf{e}}[\mathbb{E}_{\mathbf{y}|\mathbf{e}}[(\mathbf{y} - \mathbf{e})\mathbf{A} \mid \mathbf{e}]^T\mathbf{e}] && \text{(by the definition of } \mathbf{x}\text{)}\\
&= \mathbb{E}_{\mathbf{e}}[((\mathbf{e} - \mathbf{e})\mathbf{A})^T\,\mathbf{e}] && (\mathbb{E}_{\mathbf{y}|\mathbf{e}}[\mathbf{y}\mathbf{A} \mid \mathbf{e}] = \mathbf{e}\mathbf{A})\\
&= \mathbf{0},
\end{aligned}$$

where $\mathbb{E}_{\mathbf{y}|\mathbf{e}}[\mathbf{y}\mathbf{A} \mid \mathbf{e}] = \mathbb{E}_{\mathbf{y}|\mathbf{e}}[\mathbf{y} \mid \mathbf{e}]\mathbf{A} = \mathbf{e}\mathbf{A}$ holds because $\boldsymbol{\lambda}$ and $\mathbf{A}$ are constant, and hence $\mathbb{E}_{\mathbf{y}|\mathbf{e}}[\mathbf{y} \mid \mathbf{e}] = \mathbb{E}_{\mathbf{u}|\mathbf{r}}[\boldsymbol{\lambda} - \mathbf{u} \mid \boldsymbol{\lambda} - \mathbf{r}] = \mathbb{E}_{\mathbf{u}|\mathbf{r}}[\boldsymbol{\lambda} - \mathbf{u} \mid \mathbf{r}] = \boldsymbol{\lambda} - \mathbf{r} = \mathbf{e}$. □

The following is the proof of Lemma 2 in Section 3.2.

Lemma 2. *Let* $S_m = \{(\mathbf{s}_i, \mathbf{u}_i) \mid i \in \{1, 2, \ldots, m\}\}$ *be a set of* m *independent samples* $(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{A})$*. Let* $\mathbf{X} \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{E} \in \mathbb{R}^{m\times n}$ *be the input and noise matrices, respectively, in Eq.* (6)*. As* m *goes to infinity,* $\frac{1}{m}\mathbf{X}^T\mathbf{E}$ *converges to* $\mathbf{0}$ *in probability, and* $\frac{1}{m}\mathbf{X}^T\mathbf{X}$ *converges to* $Cov[\mathbf{s}, \mathbf{s}]$ *in probability. In other words,*

$$\operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\mathbf{X}^T\mathbf{E}\right) = \mathbf{0} \text{ and } \operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\mathbf{X}^T\mathbf{X}\right) = Cov[\mathbf{s}, \mathbf{s}].$$

Proof. For any $\mathbf{X} \in \mathbb{Z}^{m\times n}$ and $\mathbf{E} \in \mathbb{R}^{m\times n}$, we have

$$\begin{aligned}
\operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\mathbf{X}^T\mathbf{E}\right) &= \operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i^T\mathbf{e}_i\right) \\
&= \mathbb{E}_{\mathbf{x},\mathbf{e}}[\mathbf{x}^T\mathbf{e}] && \text{(Theorem 1)} \\
&= Cov[\mathbf{x},\mathbf{e}] - \mathbb{E}_{\mathbf{x}}[\mathbf{x}]^T\mathbb{E}_{\mathbf{e}}[\mathbf{e}] \\
&= Cov[\mathbf{x},\mathbf{e}] && (\mathbb{E}_{\mathbf{x}}[\mathbf{x}] = \mathbf{0}) \\
&= \mathbf{0}. && \text{(Lemma 1)}
\end{aligned}$$

Since $x_i = s_i$, for any $\mathbf{X} \in \mathbb{Z}^{m\times n}$, we have

$$\begin{aligned}
\operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\mathbf{X}^T\mathbf{X}\right) &= \operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{s}_i^T\mathbf{s}_i\right) \\
&= \mathbb{E}_{\mathbf{s}}[\mathbf{s}^T\mathbf{s}] && \text{(Theorem 1)} \\
&= Cov[\mathbf{s},\mathbf{s}] - \mathbb{E}_{\mathbf{s}}[\mathbf{s}]^T\mathbb{E}_{\mathbf{s}}[\mathbf{s}] \\
&= Cov[\mathbf{s},\mathbf{s}]. && (\mathbb{E}_{\mathbf{s}}[\mathbf{s}] = \mathbf{0}) \ \square
\end{aligned}$$

The following is the proof of Theorem 4 in Section 3.3

Theorem 4 (Consistency of the Proposed OLS Estimator). *Let* $S_m = \{(\mathbf{s}_i, \mathbf{u}_i) \leftarrow W_{\Upsilon,Q}(\mathbf{B}) \mid i \in \{1,2,\ldots,m\}\}$ *be a set of* m *independent GPV signature samples. Let* $\mathbf{X} \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{Y} \in \mathbb{R}^{m\times n}$ *be the input and output matrices, respectively, in Eq.* (6)*. The estimator* $\widehat{\mathbf{A}} = (\mathbf{X}^T\mathbf{Y})^{-1}\mathbf{X}^T\mathbf{X}$ *of the OLS attack in Algorithm* 4 *is consistent, i.e.,* $\operatorname{plim}_{m\to\infty}\widehat{\mathbf{A}} = \mathbf{A}$.

Proof. For any $\mathbf{X} \in \mathbb{Z}^{m\times n}$ and $\mathbf{Y} \in \mathbb{R}^{m\times n}$, we have

$$\begin{aligned}
\widehat{\mathbf{A}^{-1}} - \mathbf{A}^{-1} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} - \mathbf{A}^{-1} && (\text{by the def. of } \widehat{\mathbf{A}^{-1}}) \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\mathbf{A}^{-1} + \mathbf{E}) - \mathbf{A}^{-1} && (\text{by the def. of } \mathbf{Y}) \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\mathbf{A}^{-1} + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{E} - \mathbf{A}^{-1} \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{E}.
\end{aligned}$$

It follows that

$$\begin{aligned}
\operatorname*{plim}_{m\to\infty}\left(\widehat{\mathbf{A}^{-1}} - \mathbf{A}^{-1}\right) &= \operatorname*{plim}_{m\to\infty}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{E} \\
&= \operatorname*{plim}_{m\to\infty} m(\mathbf{X}^T\mathbf{X})^{-1}\frac{1}{m}\mathbf{X}^T\mathbf{E} \\
&= \operatorname*{plim}_{m\to\infty}\left(\frac{1}{m}\mathbf{X}^T\mathbf{X}\right)^{-1}\frac{1}{m}\mathbf{X}^T\mathbf{E} \\
&= \left(\operatorname*{plim}_{m\to\infty}\frac{1}{m}\mathbf{X}^T\mathbf{X}\right)^{-1}\operatorname*{plim}_{m\to\infty}\frac{1}{m}\mathbf{X}^T\mathbf{E} && \text{- - - - (14)} \\
&= Cov[\mathbf{s},\mathbf{s}]^{-1}\cdot\mathbf{0} && \text{(by Lemma 2)} \\
&= \mathbf{0}.
\end{aligned}$$

Here, the equality (14) holds as follows. Since $Cov[\mathbf{s}, \mathbf{s}]$ is a positive semi-definite matrix, it is invertible except when $\boldsymbol{\beta} Cov[\mathbf{s}, \mathbf{s}] \boldsymbol{\beta}^T = 0$ for some $\boldsymbol{\beta} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$. Then, $\boldsymbol{\beta} Cov[\mathbf{s}, \mathbf{s}] \boldsymbol{\beta}^T$ can be expressed as follows.

$$\begin{aligned} \boldsymbol{\beta} Cov[\mathbf{s}, \mathbf{s}] \boldsymbol{\beta}^T &= \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_i \beta_j Cov[s_i, s_j] \\ &= Var\left[\sum_{i=1}^{n} \beta_i s_i\right] \\ &= Var[\langle \boldsymbol{\beta}, \mathbf{s} \rangle], \end{aligned}$$

where β_i and s_i are the ith coordinate of $\boldsymbol{\beta}$ and $\mathbf{s}$, respectively. Since $\mathbf{s}$ is distributed according to an n-dimensional discrete Gaussian distribution over lattice, $Var[\langle \boldsymbol{\beta}, \mathbf{s} \rangle] \neq 0$ for all $\boldsymbol{\beta} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$. Therefore, $Cov[\mathbf{s}, \mathbf{s}]$ is invertible and, by Theorem 2, the equality (14) holds.

Since $\mathbf{A}^{-1}$ is fixed as constant for any m and $\operatorname{plim}_{m \to \infty}(\widehat{\mathbf{A}^{-1}} - \mathbf{A}^{-1}) = \mathbf{0}$, it follows that

$$\operatorname*{plim}_{m \to \infty} \widehat{\mathbf{A}^{-1}} = \mathbf{A}^{-1}. \qquad \text{- - - - - - - - (15)}$$

Let $g : GL_n(\mathbb{R}) \to GL_n(\mathbb{R})$ be a function of calculating inverse matrix. Then g is continuous for every $GL_n(\mathbb{R})$ [34]. We now show the convergence of $\widehat{\mathbf{A}}$ to $\mathbf{A}$ as follows:

$$\begin{aligned} \operatorname*{plim}_{m \to \infty} \widehat{\mathbf{A}} &= \operatorname*{plim}_{m \to \infty} \left(\widehat{\mathbf{A}^{-1}}\right)^{-1} && \text{(by Eq. (8))} \\ &= \operatorname*{plim}_{m \to \infty} g\left(\widehat{\mathbf{A}^{-1}}\right) \\ &= g\left(\operatorname*{plim}_{m \to \infty} \widehat{\mathbf{A}^{-1}}\right) && \text{(by Theorem 3)} \\ &= g\left(g(\mathbf{A})\right) && \text{(by Eq. (15))} \\ &= \mathbf{A}. \end{aligned}$$

Since $\mathbf{A}$ is either $\mathbf{B}$ or $\tilde{\mathbf{B}}$, all rows of $\mathbf{A}$ are linearly independent, and hence $\mathbb{P}\left(g(\mathbf{A}) \notin GL_n(\mathbb{R})\right) = 0$. Since g is continuous on $GL_n(\mathbb{R})$, by Theorem 3, the last equality holds. $\square$

The following is the proof of Corollary 1 in Section 4.1.

Corollary 1. *Let* $\mathbf{z} = (z_1, z_2, \ldots, z_{2n}) \in \mathbb{Z}^{2n}$ *be the output of FFO sampler* $ffsampling_n$*, and let* $\bar{\mathbf{z}} = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_{2n}) \in \mathbb{Z}^{2n}$ *be the vector such that* $\bar{z}_i$ *is calculated from* $z_i = \bar{z}_i + \lfloor c_i \rfloor$ *in Algorithm* 2*. Here,* c_i*, serving as an input to SamplerZ for producing the output* z_i*, is equal to* $\tilde{t}_i$ *of* $\tilde{\boldsymbol{t}} = (\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_{2n})$ *in Lemma* 3*. Then for any signature* $\mathbf{s} \in \mathbb{Z}^{2n}$ *of Falcon and* $\bar{\mathbf{z}}$*, there exists* $\mathbf{r} = (r_1, r_2, \ldots, r_{2n}) \in [0, 1)^{2n}$ *such that*

$$\mathbf{s} = (\mathbf{r} - \bar{\mathbf{z}})\tilde{\mathbf{B}},$$

where $\bar{z}_i$ is distributed according to $D_{\mathbb{Z},\sigma_i,r_i}$ for $i \in \{1, 2, \ldots, 2n\}$.

Proof. Let $\mathbf{l} = (l_1, l_2, \ldots, l_{2n})$ where $l_i = \lfloor \tilde{t}_i \rfloor$, and let $\mathbf{r} = (r_1, r_2, \ldots, r_{2n})$ where $r_i = \tilde{t}_i - l_i$. Since $\tilde{\mathbf{t}} = \mathbf{l} + \mathbf{r}$ and $\mathbf{z} = \bar{\mathbf{z}} + \mathbf{l}$, by Lemma 3, we obtain that

$$\mathbf{s} = (\tilde{\mathbf{t}} - \mathbf{z})\tilde{\mathbf{B}} = (\mathbf{l} + \mathbf{r} - (\mathbf{l} + \bar{\mathbf{z}}))\tilde{\mathbf{B}} = (\mathbf{r} - \bar{\mathbf{z}})\tilde{\mathbf{B}}.$$

We now derive the distribution of $\bar{\mathbf{z}}$. Since $D_{\mathbb{Z},\sigma_i,c-a}(x)$ is the same as $D_{\mathbb{Z},\sigma_i,c}(x + a)$ for any $x, a \in \mathbb{Z}$ and $c \in \mathbb{R}$, we have

$$D_{\mathbb{Z},\sigma_i,\tilde{t}_i}(z_i) = D_{\mathbb{Z},\sigma_i,\tilde{t}_i}(\bar{z}_i + l_i) = D_{\mathbb{Z},\sigma_i,\tilde{t}_i - l_i = r_i}(\bar{z}_i),$$

for all $i \in \{1, 2, \ldots, 2n\}$ given l_i. Therefore, the distribution of $\bar{z}_i$ is $D_{\mathbb{Z},\sigma_i,r_i}$. □

The following is the proof of Theorem 5 in Section 4.2.

Theorem 5 (Consistent Estimator $\widehat{\mathbf{b}_1}$ in Eq. (12)). *Let $\mathbf{B}$ be the secret basis of Falcon, and $W_{\chi,D}(\tilde{\mathbf{B}})$ be the probabilistic algorithm whose outputs $(\mathbf{s}, \bar{\mathbf{z}})$ satisfy Eq. (9). Here, χ and D are distribution of r_i and discrete Gaussian distribution, respectively. From m independently measured samples $(\mathbf{s}, \bar{\mathbf{z}}) \leftarrow W_{\chi,D}(\tilde{\mathbf{B}})$, the MLR model in Eq. (6) for Falcon is constructed using $\mathbf{X}$ and $\mathbf{Y}$ consisting of m row vectors $\mathbf{x}_\mathrm{i} = \mathbf{s}_\mathrm{i}$ and $\mathbf{y}_\mathrm{i} = \boldsymbol{\lambda}_\mathrm{i} - \bar{\mathbf{z}}_\mathrm{i}$, respectively. As m goes to infinity, the estimator $\widehat{\mathbf{b}_1} = \frac{1}{k}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}'_1)^T$ converges to the first row $\mathbf{b}_1$ of the secret basis $\mathbf{B}$ in probability, where $\mathbf{y}'_1 \in \mathbb{Q}^m$ is the first column of $\mathbf{Y}$ consisting of m elements $\lambda_1 - \bar{z}_{i,1}$ and $k = \|(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}'_1\|^2$, i.e., $\operatorname{plim}_{m\to\infty} \widehat{\mathbf{b}_1} = \mathbf{b}_1$.*

Proof. Let $\mathbf{z}_i \in \mathbb{Z}^n$ be the ith column of $\mathbf{Z} \in \mathbb{Z}^{n \times n}$. Then, the matrix norm of $\mathbf{Z}$ is defined as

$$\|\mathbf{Z}\| := \max_{1 \le i \le n} \|\mathbf{z}_i\|.$$

It follows that, for any $\varepsilon > 0$, we have

$$0 \le \mathbb{P}(\|\widehat{\mathbf{v}_1} - \mathbf{v}_1\| > \varepsilon) \le \mathbb{P}(\|\widehat{\mathbf{A}^{-1}} - \mathbf{A}^{-1}\| > \varepsilon),$$

where $\widehat{\mathbf{v}_1} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}'_1$ is the first column of $\widehat{\mathbf{A}^{-1}}$ in Eq (7) and $\mathbf{v}_1$ is the first column of $\mathbf{A}^{-1}$. Since $\lim_{m\to\infty} \mathbb{P}(\|\widehat{\mathbf{A}^{-1}} - \mathbf{A}^{-1}\| > \varepsilon) = 0$ by Theorem 4, it is clear that $\lim_{m\to\infty} \mathbb{P}(\|\widehat{\mathbf{v}_1} - \mathbf{v}_1\| > \varepsilon) = 0$ by squeeze theorem (sandwich theorem). In other words, the estimator $\widehat{\mathbf{v}_1}$ converges to $\mathbf{v}_1$ in probability.

Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be the function defined as follows:

$$f(\mathbf{x}) := \frac{1}{\|\mathbf{x}\|^2}\mathbf{x}^T.$$

The function f is continuous for $\mathbb{R}^n \backslash \{\mathbf{0}\}$ because L2 norm and the transpose mapping are continuous for $\mathbb{R}^n \backslash \{\mathbf{0}\}$. Since $\mathbb{P}(\mathbf{v}_1 = \mathbf{0}) = 0$, by Theorem 3, it follows that

$$\operatorname*{plim}_{m\to\infty} f(\widehat{\mathbf{v}_1}) = f\left(\operatorname*{plim}_{m\to\infty} \widehat{\mathbf{v}_1}\right) = f(\mathbf{v}_1) = \mathbf{b}_1. \qquad \square$$

B Hyperparameters of MLP

This appendix presents the hyperparameters of the MLP to ensure the reproducibility of the experiments and shares the optimized parameters that were empirically verified through the study. In Table 4, “FC” denotes a fully connected layer, and the following number represents the layer depth. The MLP adjusts its input size to match the number of power traces corresponding to z_0 and b.

Table 4. Hyperparameters of MLP model for binary classification

	Input Size	**Output Size**	**Activation function**	**Dropout**	**Negative slope**
FC1	Input size	91	LeakyReLU	0.2	0.01
FC2	91	400	LeakyReLU	0.2	0.01
FC3	400	128	LeakyReLU	0.2	0.01
FC4	128	32	LeakyReLU	0.2	0.01
FC5	32	1	LeakyReLU	0.2	0.01
FC6	1	1	Sigmoid	-	-