symmetry

MDPI

*Article*

# Exploring the Intersection of Lattice Attacks and Blockchain Technology: A Heuristic Approach Using TPM2.0 ECDSA to Ascertain and Approach the Boundary

**Baohua Zhao [1,2,3,*], Xiao Zhang [4], Zhihao Wang [2,3], Shucai Wang [2,3], Fajiang Yu [5] and Yaomin Jia [5]**

[1] School of Computer Science, Beijing University of Technology, Beijing 100124, China
[2] State Grid Smart Grid Research Institute Co., Ltd., Beijing 102209, China
[3] State Grid Laboratory of Grid Advanced Computing and Applications, Beijing 102209, China
[4] State Grid Corporation of China Co., Ltd., Beijing 100031, China
[5] School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China
[*] Correspondence: zhaobh@emails.bjut.edu.cn

**Abstract:** Lattice attacks can compromise the security of encryption algorithms used in blockchain networks, allowing attackers to tamper with transaction records, steal private keys, and execute other forms of attacks. With symmetric encryption, both parties can encrypt and decrypt messages using the same key. Lattice attacks on digital signature algorithms (ECDSA) involve forming a basis and setting a target vector from signatures, then solving the closest vector problem (CVP) or shortest vector problem (SVP) in the generated lattice to obtain the private key. Prior research focused on obtaining leakage information from the signature's random nonce to facilitate a CVP or SVP solution. This study establishes a clear boundary for a successful ECDSA attack and introduces a "double basis" lattice version that expands the boundary or reduces the necessary signatures by nearly half with the same lattice rank. To approach the boundary, a heuristic strategy is employed to shift the target vector in different directions with a feasible step size, using tests on the Trusted Platform Module (TPM) 2.0 ECDSA. The distance from the closest moved target vector to the boundary is reduced by a ratio of 424 to 179 to the minimal length of orthogonal vectors in the formed basis. Experimental results show that moving attempts in two directions with the original basis and 84 signatures take approximately 247.7 s on the experiment computer.

**Keywords:** elliptic curve digital signature algorithm; blockchain; closest vector problem; heuristic strategy; trusted platform module TPM2.0; symmetric encryption

check for updates

**Citation:** Zhao, B.; Zhang, X.; Wang, Z.; Wang, S.; Yu, F.; Jia, Y. Exploring the Intersection of Lattice Attacks and Blockchain Technology: A Heuristic Approach Using TPM2.0 ECDSA to Ascertain and Approach the Boundary. *Symmetry* **2023**, *15*, 913. https://doi.org/10.3390/sym15040913

Academic Editors: Calogero Vetro, Miodrag J. Mihaljevic and Chin-Ling Chen

Received: 8 February 2023
Revised: 14 March 2023
Accepted: 7 April 2023
Published: 14 April 2023

## 1. Introduction

With the increasing adoption of blockchain technology in various industries, ensuring the security of blockchain systems has become an important concern. However, recent research has shown that lattice-based attacks, a type of cryptographic attack, could pose a threat to the security of blockchain systems, including the symmetric encryption used to encrypt and decrypt messages. Lattice-based attacks exploit the mathematical properties of lattices, a type of mathematical structure, to break cryptographic systems. These attacks could compromise the security of the underlying encryption and signature schemes used in blockchain technology, including symmetric encryption. In this context, it is important to understand the relationship between lattice-based attacks and the security of blockchain systems and to explore potential solutions to mitigate such attacks, including symmetric encryption methods that provide an additional layer of security.

In 1996, Boneh et al. [1] first proposed the lattice method to attack the Diffie-Hellman key exchange algorithm; the method was then extended to attack other crypto-ecosystems such as Rivest–Shamir–Adleman (RSA) [2], Digital Signature Algorithm (DSA) [3–6], and Elliptic Curve Digital Signature Algorithm (ECDSA) [7]. The general method of lattice attacks on (EC)DSA can be divided into three main steps [8]. First, the attacker undertakes the task of securing the private key into a hidden number problem (HNP) or extended hidden number problem (EHNP) from the congruent equations with the signatures. Next, the HNP or EHNP is transformed into a lattice closest vector problem (CVP) or shortest vector problem (SVP). Finally, attempts are made to solve the CVP or SVP via lattice-related techniques—such as lattice basis reduction [9] and the Babai algorithm [10]—while the private key is embedded in the solution.

To help identify the correct vector solution, existing work has focused on exploiting random noncoded leaks in the cryptographic computation process, such as timing, cache activity, and side-channels such as energy consumption. The most common nonce leaks are where the most significant bits (MSBs) or least significant bits (LSBs) are set or cleared. Weiser et al. [11] pointed out the locations where nonce leakage might occur in the implementations of ECDSA.

In 2011, Brumley et al. [12] exploited the timing leakage in the implementation of the Montgomery Ladder algorithm to obtain the MSBs of nonces to break the ECDSA in 8000 Transport Layer Security (TLS) handshakes. Moghimi et al. [13] used timing side-channel and lattice attacks to fully recover ECDSA private keys in hardware and firmware trusted platform modules (TPMs). They exploit the vulnerability that the CPU time of an encrypted computation is related to the nonce used by ECDSA. A shorter nonce (with a partial MSB of 0) leads to faster computations; the attacker can find the shorter nonce by filtering the execution time. They attack from three different angles: the system level with administrator privileges to accurately measure the time, the user level with a less privileged API to measure the execution time, and the remote level where the time can only be measured over the network [14], which was seen as the first work to use a cache side-channel attack to obtain nonce leakage. The task of recovering the DSA private key was turned into an EHNP. Brumley et al. [15] recovered a 160-bit private key by using a cache-timing template attack to obtain the LSBs of the nonce. In 2014, Yarom et al. proposed a cache attack called Flush + Reload [16]; they used it to precisely measure the computation of RSA signing and extracted the private key successfully. Some work has tried to migrate the Flush + Reload method to attack ECDSA. Benger et al. [17] applied the Flush + Reload technique to obtain cache activities, infer the LSBs of the nonce in OpenSSL ECDSA, and solve the SVP to obtain the private key. OpenSSL uses window Non-Ajacent-Form (wNAF) to recode the random nonce with a fixed size window and a value $d_i$ for each other window. For point multiplication, when $d_i$ is nonzero, double and add operations are carried out; otherwise, only double operations are carried out. With a spy program running to monitor the cache activities, the attacker can obtain the operation sequences related to the nonce to obtain some leakage information. Fan et al. [18] proposed an efficient method to obtain information from side-channels with only a few signatures, under the assumption that Flush + ReLoad is implemented perfectly. Wang et al. [19] utilized the Flush + ReLoad technique to extract both MSBs and LSBs of the nonce and recovered the private key with only 85 signatures.

In 2014, Aranha et al. [20] used power analysis to obtain one-bit leakage of the nonce of ECDSA then recovered the private key with Gallant–Lambert–Vanstone (GLV)/Galbraith–Lin–Scott (GLS) decomposition. Genkin et al. [21] extracted the LSBs of nonces from mobile devices via the power of electromagnetic attack method. In 2016, Belgarric et al. [22] also used the same side-channel attack to obtain LSBs and private keys on Android smartphones. Zhang et al. [23] used a power attack to extract LSBs of the nonce in the Chinese SM2 Digital Signature Algorithm and recovered the private key with an instance of HNP.

The fault injection attack can also be used to obtain nonce bits leakage. In 2012, Nguyen et al. [24] used a fault injection attack to obtain LSBs of RSA and LSBs of DSA nonces. Cao et al. [25] used a different fault injection attack to obtain the LSBs of a ECDSA nonce.

When some fixed methods have been patched into the ECDSA implementations, it has become more difficult to obtain the leakage information of the random nonce. Albrecht et al. [26] noted the presence of a “lattice barrier” for the attack when the leaked bit length was short. The development of effective methods to mitigate lattice-based attacks is crucial to ensure the security and confidentiality of data in the blockchain network, including those protected by symmetric encryption methods.

In this paper, we try to ascertain the boundary of the lattice attack on ECDSA and move the target vector closer to the boundary. We have made the following main contributions:

(1) We ascertain that the boundary of the lattice attack on ECDSA is half the minimal length of the vectors in the Gram–Schmidt orthogonal basis formed from signatures by proving a theorem that states the condition that the Babai algorithm outputs the correct closest vector.

(2) We form a variant of the lattice basis to attack ECDSA called the double basis, which can expand the boundary of a successful attack or reduce the number of required signatures by almost half with the same lattice rank.

(3) Using heuristics in the TPM2.0 ECDSA experiments, we attempted to move the target vector in different directions to move closer to the boundary, the larger the step size, the better. In our experiments, the distance from the closest moving target vector to the boundary is reduced in proportion to the minimum length of the orthogonal vector, from 424 to 179. On our experimental computer, it took approximately 247.7 s to complete the move attempt with the original basis and 84 signatures in both directions.

The remainder of this paper is organized as follows: Section 2 introduces the necessary background, Section 3 presents the steps of the lattice attack on ECDSA in general, Section 4 proves a theorem to describe the boundary of the lattice attack, Section 5 describes the boundary value experiment, Section 6 forms a variant of the lattice basis, Section 7 states the process of moving the target vector to move closer to the boundary, and Section 8 draws the conclusions and discusses future work.

## 2. Background

In this section, we give a brief introduction to ECDSA and lattices.

### 2.1. ECDSA

The elliptic curve digital signature algorithm (ECDSA) is widely used. To sign one digest $h(m)$ of a message $m$, the ECDSA takes a private key $\alpha$ and a set of public parameters, including a generator point $G$ with order $q$ on an elliptic curve $E$ over a finite field. The private key $\alpha \in Z_q^*$. The signer takes the following actions:

(1) Generates a per message random nonce $k \in Z_q^*$;

(2) Computes $kG$, and takes the x-coordinate of $kG$ as $r$;

(3) Computes the value $s = k^{-1}(h(m) + \alpha r) \mathrm{mod} q$;

(4) Takes the value pair $(r, s)$ as the signature.

### 2.2. Lattice and Closest Vector Problem

A lattice is a discrete subgroup of vector space Rm. Let $B = \{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-1}\}$ be $n$ linearly independent vectors in $R^m$, the set of integer linear combinations of $\boldsymbol{b}_i$ forms a lattice $L$ and the vectors $B$ are called a lattice basis. In this work, only the full-rank lattice is considered, which means $m = n$.

$$L = \{\sum_{i=0}^{n-1} z_i \boldsymbol{b}_i : z_i \in Z\} \tag{1}$$

The closest vector problem (CVP) is a computational problem in a lattice: given a lattice $L$ spanned by a basis $B$ and a vector $\boldsymbol{t}$ in $R^m$, find a vector $\boldsymbol{v}$ in $L$ such that the distance $||\boldsymbol{v} - \boldsymbol{t}||$ is minimal for all $\boldsymbol{v}$ in $L$. $||\boldsymbol{v}||$ denotes the Euclidean norm of vector $\boldsymbol{v}$.

Although CVP is known to be an NP-hard problem, the Babai nearest plane algorithm [10] can find a vector not far from the closest vector in polynomial time. The Babai algorithm uses Gram–Schmidt orthogonalization as a subroutine, which computes an orthogonal basis $\{\boldsymbol{b}_0^*, \boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_{n-1}^*\}$ for a basis $B$.

## 3. General Lattice Attack on ECDSA

Suppose we have obtained $n-1$ ECDSA signatures.

$$\begin{cases} s_i = k_i^{-1}(h(m) + \alpha r_i) \bmod q \\ r_i = x_i, (x_i, y_i) = k_i G \end{cases} (i = 1, 2, \ldots, n-1) \tag{2}$$

and there are:

$$k_i + C_i\alpha + D_i = 0 \bmod q \ (i = 1, 2, \ldots, n-1) \tag{3}$$

where $C_i = -r_i s_i^{-1} \bmod q$, $D_i = -h(m)s_i^{-1} \bmod q$.

These equations can also be expressed as:

$$k_i + C_i\alpha + D_i = z_i\text{q} \ (i = 1, 2, \ldots, n-1) \tag{4}$$

$$C_i(-\alpha) + z_i\text{q} - D_i = k_i \ (i = 1, 2, \ldots, n-1) \tag{5}$$

Adding one equation $(-1)(-\alpha) - 0 = \alpha$, we have the following joint equations:

$$\begin{cases} (-1)(-\alpha) - 0 = \alpha \ (\text{i} = 0) \\ C_i(-\alpha) + z_i\text{q} - D_i = k_i \ (i = 1, 2, \ldots, n-1) \end{cases} \tag{6}$$

These joint equations can be expressed as matrices and vectors operations:

$$\begin{pmatrix} -1 & & & \\ C_1 & q & & \\ \vdots & & \ddots & \\ C_{n-1} & & & q \end{pmatrix} \begin{pmatrix} -\alpha \\ z_1 \\ \vdots \\ z_{n-1} \end{pmatrix} - \begin{pmatrix} 0 \\ D_1 \\ \vdots \\ D_{n-1} \end{pmatrix} = \begin{pmatrix} \alpha \\ k_1 \\ \vdots \\ k_{n-1} \end{pmatrix} \tag{7}$$

Let $\boldsymbol{b}_0 = (-1, C_1, \ldots, C_{n-1})^T$, $\boldsymbol{b}_1 = (0, q, 0, \ldots, 0)^T$, $\ldots$ , $\boldsymbol{b}_{n-1} = (0, \ldots, 0, q)^T$ be linearly independent vectors; then, the set of these vectors $B$ can be seen as a lattice basis.

$$B = \{\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n-1}\} \tag{8}$$

The lattice generated by $B$ is the set $L(B)$.

$$L(B) = \{\sum_{i=0}^{n-1} z_i \boldsymbol{b}_i : z_i \in Z\} \tag{9}$$

The vector $(0, D_1, \ldots, D_{n-1})^T$ in Equation (7) can be seen as a target vector $\boldsymbol{t}$ not in the lattice $L(B)$.

$$t = (t_0, \ldots, t_{n-1})^T = (0, D_1, \ldots, D_{n-1})^T \tag{10}$$

The vector $(-\alpha, z_1, \ldots, z_{n-1})^T$ is an integer coefficient vector for the lattice basis $B$, which determines a vector $\boldsymbol{v}$ in $L(B)$. The distance between $\boldsymbol{v}$ and $\boldsymbol{t}$ is the length of vector $(\alpha, k_1, \ldots, k_{n-1})^T$.

$$||\boldsymbol{v} - \boldsymbol{t}|| = \sqrt{\alpha^2 + \sum_{1}^{n-1} k_i^2} \tag{11}$$

If this distance is sufficiently short, then $\boldsymbol{v}$ can be obtained by solving the CVP of $L(B)$, and the private key $\alpha$ is the first element of $\boldsymbol{v}$.

## 4. Boundary of Successful Attack

If the correct closest vector described in Equation (7) was found, the lattice attack succeeded. The Babai algorithm [10] is a useful method to solve the CVP [27] in the situation proposed by Theorem 1 [28]. In this section, we prove that we need this theorem for the description of our work.

**Theorem 1.** *Let* $\{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-1}\}$ *be an ordered basis for a lattice L. Let* $\left\{\boldsymbol{b}_0^*, \boldsymbol{b}_1^*, ..., \boldsymbol{b}_{n-1}^*\right\}$ *be the corresponding Gram–Schmidt basis. If the distance of target vector* $\boldsymbol{t}$ *to its nearest lattice vector* $\boldsymbol{v}$ *is less than* $min(\frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|)$ $(i = 0, 1, ..., n-1)$*, the Babai algorithm always outputs the vector* $\boldsymbol{v}$*.*

First, the Babai algorithm computes the Gram–Schmidt basis $\{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-1}\}$. Define $U$ as the space spanned by basis $\{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-2}\}$. This nearest plane algorithm then tries to find a vector $\boldsymbol{y}^{n-1} \in L$ such that the distance from $\boldsymbol{t}$ to the plane formed by $\boldsymbol{y}^{n-1} + U$ is minimal.

Let target vector $\boldsymbol{t}$ be $(t_0, ..., t_{n-1})^T$, which can be written as $\boldsymbol{t} = \sum_{i=0}^{n-1} r_i \boldsymbol{b}_i^*, r_i \in R$, and $t_i = r_i||\boldsymbol{b}_i^*||$. According to Lemma 18.1.1 in [28], when $\boldsymbol{y}^{n-1} = [r_{n-1}]\boldsymbol{b}_{n-1}$, the distance between $\boldsymbol{t}$ and the plane $\boldsymbol{y}^{n-1} + U$ is minimal. Since $t_{n-1} = r_{n-1}\left|\left|\boldsymbol{b}_{n-1}^*\right|\right|$, then $r_{n-1} = \boldsymbol{t}_{n-1}/\left|\left|\boldsymbol{b}_{n-1}^*\right|\right|$. Let $\boldsymbol{t}\prime$ be the orthogonal projection of $\boldsymbol{t}$ on the plane $\boldsymbol{y}^{n-1} + U$, and let $L\prime = L \cap U$ be the sublattice spanned by the basis $\{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-2}\}$. Let $\boldsymbol{t}'' = \boldsymbol{t}\prime - \boldsymbol{y}^{n-1} \in U$. Then, the algorithm attempts to solve the CVP of the target $\boldsymbol{t}''$ with the lattice $L\prime$ to obtain $\boldsymbol{y}^{n-2}$. Similarly, $\boldsymbol{y}^{n-2} = r_{n-2}\left|\left|\boldsymbol{b}_{n-2}^*\right|\right|$, $r_{n-2} = t_{n-2}/\left|\left|\boldsymbol{b}_{n-2}^*\right|\right|, \ldots, \boldsymbol{y}^0 = r_0\left|\left|\boldsymbol{b}_0^*\right|\right|$, $r_0 = t_0/||\boldsymbol{b}_0^*||$. Finally, the Babai algorithm outputs $\boldsymbol{v} = \boldsymbol{y}^{n-1} + ... + \boldsymbol{y}^0$ as the closest vector.

In other form, $\boldsymbol{v} = \sum_{i=0}^{n-1} [r_i]\boldsymbol{b}_i$, then $\boldsymbol{v} - \boldsymbol{t} = \sum_{i=0}^{n-1} [r_i]\boldsymbol{b}_i - \sum_{i=0}^{n-1} r_i \boldsymbol{b}_i^*$. Since $r_i - [r_i] \leq \frac{1}{2}, \left|r_i\left|\left|\boldsymbol{b}_i^*\right|\right| - [r_i]\left|\left|\boldsymbol{b}_i^*\right|\right|\right| \leq \frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|$ and $\left|[r_i]\left|\left|\boldsymbol{b}_i^*\right|\right| - \boldsymbol{t}_i\right| \leq \frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|$ for $0 \leq i \leq n-1$. $[r_i]||\boldsymbol{b}_i^*||$ is the projection of $\boldsymbol{v}$ on $\boldsymbol{b}_i^*$, so the vector $\boldsymbol{v} - \boldsymbol{t}$ lies in the range of:

$$\{\sum_{i=0}^{n-1} r_i \boldsymbol{b}_i^* : |r_i| \leq \frac{1}{2}\} \tag{12}$$

which forms a hyperrectangle. Its center is the output vector of the Babai algorithm.

For a target vector $\boldsymbol{t}$ with a distance within $\min(\frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|)$ to vector $\boldsymbol{v}$, that is, $\boldsymbol{v} - \boldsymbol{t} \leq \min(\frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|)$, the distance from $\boldsymbol{t}$ to $\boldsymbol{v}$'s adjacent point $(\boldsymbol{v}\prime)||\boldsymbol{v}\prime - \boldsymbol{t}|| = ||\boldsymbol{b}_i|| - ||\boldsymbol{v} - \boldsymbol{t}||$ for $i = 0, 1, ..., n-1$. Since the Gram–Schmidt orthogonal basis holds $||\boldsymbol{b}_i|| \geq ||\boldsymbol{b}_i^*||$, $\left|\left|\boldsymbol{v}\prime - \boldsymbol{t}\right|\right| \geq \left|\left|\boldsymbol{b}_i^*\right|\right| - \left|\left|\boldsymbol{v} - \boldsymbol{t}\right|\right| \geq \min(\frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|)$. Then, all the adjacent points have a longer distance to the target vector $\boldsymbol{t}$ compared with $\boldsymbol{v}$. The output lattice point of the Babai algorithm is the closest vector to the target vector.

## 5. Minimal Length of $b_i^*$

For the attack to succeed the distance from the target to the correct closest vector should be in the boundary of half the minimal length of $\boldsymbol{b}_i^* (0 \leq i \leq n-1)$, as shown in Theorem 1. The orthogonal basis $\left\{\boldsymbol{b}_0^*, \boldsymbol{b}_1^*, ..., \boldsymbol{b}_{n-1}^*\right\}$ in Theorem 1 for the original basis $\{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-1}\}$ is calculated by Gram–Schmidt orthogonalization. The idea is to first set $\boldsymbol{b}_1^* = \boldsymbol{b}_1$ and then compute $\boldsymbol{b}_i^* = \boldsymbol{b}_i - \sum_{j=0}^{i-1} u_{i,j}\boldsymbol{b}_j^*, (0 \leq i \leq n-1)$, where $u_{i,j} = <\boldsymbol{b}_i, \boldsymbol{b}_j^*> / \left|\left|\boldsymbol{b}_j^*\right|\right|^2$. In this section, we attempt to obtain the minimal $||\boldsymbol{b}_i^*||$ for Equation (7) via experiments, as illustrated in Figure 1.

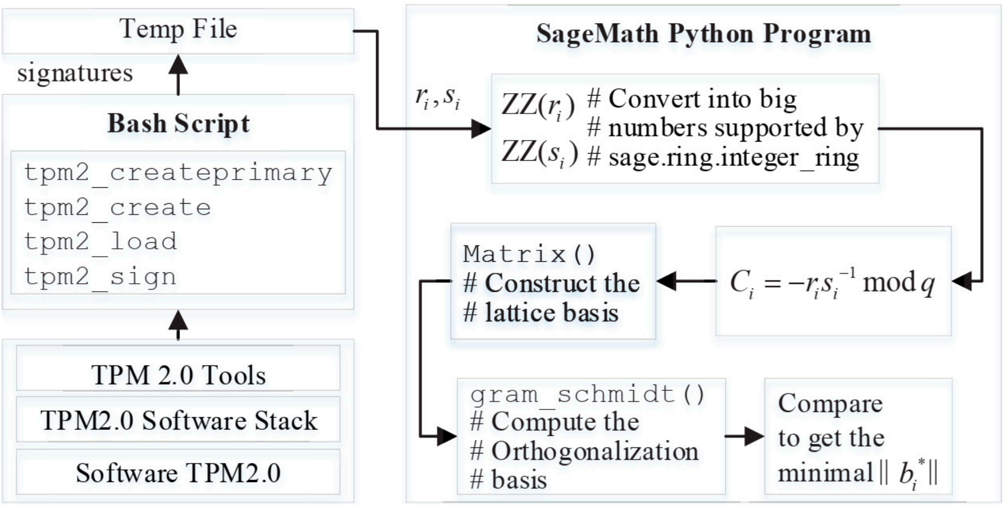

**Figure 1.** Diagram of the experiment to obtain $\min(||\boldsymbol{b}_i^*||)$.

We performed the experiments by using open-source software, including IBM's Software TPM2.0 [29], Intel's TPM2.0 software stack [30], TPM2.0 Tools [31], and SageMath9.1.

First, we launched TPM2.0 software and the TPM2 Access Broker (TAB) and Resource Manager (RM) service daemons (tpm2-abrmd service). To obtain the ECDSA signature pairs, we created a bash script that runs various command tools provided by the TPM2.0 Tools. The bash script first runs the command tpm2_createprimary with the argument ecc256 to construct a primary key, then runs the command tpm2_create to generate an ECC sign key pair under the newly formed primary key. After loading the ECC sign key into the TPM by command tpm2_load, the script requests that the TPM sign the pre-generated message digest computed by SHA256 with the command tpm2_sign. With the same sign key, we can obtain more signatures by executing tpm2_sign more times. If we want the signature to have a new and different sign key, the bash script should re-execute tpm2_create, tpm2_load, and tpm2_sign, under the same primary key. When the command tpm2_sign with ECDSA is executed, the software TPM2.0 invokes the function BnSignEcdsa(). We modified the source code of the function BnSignEcdsa() to let the software TPM2.0 output the generated signatures, the random nonces, and the generator order q to temporary files directly.

Upon the signatures being saved into the temp file, the SageMath Python program—written by us for this experiment—constructs the lattice basis and computes its Gram–Schmidt orthogonal basis to obtain the minimal $||\boldsymbol{b}_i^*||$. The Python program first loads the signatures and the parameter $q$ from the temporary files to the SageMath environment. Then, the program converts the strings to type sage.ring.integer_ring, which natively supports large number modular computation, and computes $C_i = -r_i s_i^{-1} \text{mod} q, 0 \le i \le n-1$. Now, the lattice basis can be constructed from $C_i$ and $q$ with the matrix() utility. Finally, the program obtains the Gram–Schmidt basis with the function gram_schmidt() and finds the minimal length of $\boldsymbol{b}_i$, $0 \le i \le n-1$.

In our experiments, we generated 199 signatures. With equation $(-1)(-\alpha) - 0 = \alpha$, we can construct 199 different lattice bases with ranks from 2 to 200. Every lattice basis has a minimal $||\boldsymbol{b}_i^*||$. All the half minimal $||\boldsymbol{b}_i^*||$'s bit lengths are shown in Figure 2. We can see when the rank is under approximately 40, the minimal $||\boldsymbol{b}_i^*||$ rises quickly with the growth of the rank. When the rank is between approximately 40 and 70, the minimal $||\boldsymbol{b}_i^*||$ rises slowly. When the rank is greater than approximately 70, the minimal $||\boldsymbol{b}_i^*||$ is basically stable. This means, $\min(||\boldsymbol{b}_i^*||)$ does not continuously increase with increasing lattice rank.

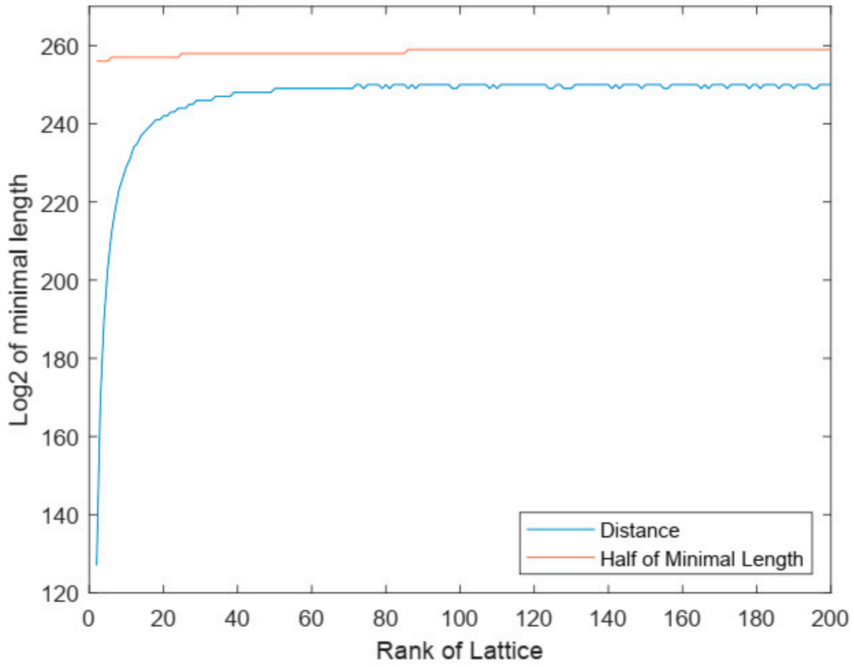

**Figure 2.** The bit length of the minimal $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ and the distance between $\boldsymbol{v}$ and $\boldsymbol{t}$ for Equation (7).

The maximum of the minimal $\frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|$ obtained in our experiments is shown in Table 1, its corresponding bit length is 250 with lattice rank 105. The default curve used by the software TPM2.0 for ECDSA is NIST-p256 with a 256-bit order $q$. The maximum of the minimal approximately $\frac{1}{2}\left|\left|\boldsymbol{b}_i^*\right|\right|$ equals $0.013q$.

**Table 1.** Some $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ )s and some distances between $\boldsymbol{v}$ and $\boldsymbol{t}$ for Equation (7).

| Item | Value | Bit Length | Rank |
|---|---|---|---|
| $\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|)$ | 0X61AF733A022AB4CA5F264CAC644AEDAE | 127 | 2 |
| $\max(\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|))$ | 0X38DA9F271B1BF4296697B0E5D094494EB85004067B384C8A14AEE695030A8CC | 250 | 105 |
| $\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|)$ | 0X3123B127700F8CD8418EC35B6250F5C8424324B3870EFD3E9B5126DFA4D3E23 | 250 | 200 |
| $q$ | 0xFFFFFFFF00000000FFFFFFFFFFFFFFFFBCE6FAADA7179E84F3B9CAC2FC632551 | 256 | |
| $\min(\|\|\boldsymbol{v}-\boldsymbol{t}\|\|)$ | 0XCD0B58F65D1F87F005FF9FA142D2CEDCC0D8A746886933746546CB1BA5D7598F | 256 | 2 |
| $\|\|\boldsymbol{v}-\boldsymbol{t}\|\|$ | 0X614C3BBC9CB9D043F131FDCB8B54EA3E032620EFF5768CC9FD73B4F9595BB569F | 259 | 105 |
| $\max(\|\|\boldsymbol{v}-\boldsymbol{t}\|\|)$ | 0X85F80F52F5C1645FAD8A8D5A27EC4733765F02EA90B0D44D4260BDD977C7DC6E7 | 259 | 200 |

If we want to continue attacking by attempting to solve the CVP with the basis $B$ and target $\boldsymbol{t}$ in Equation (7), the closest vector we want to find is $(-\alpha, z_1, \ldots, z_{n-1})^T$; the distance between $\boldsymbol{v}$ and $\boldsymbol{t}$ can be calculated by Equation (11). The bit length of the distance in our experiments is also shown in Figure 2. We can see the distance rises very slowly at the start then remains almost stable with the growth of the lattice rank. The range of the distance bit length is from 256 to 259. When the rank is 105, the $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ we obtained is the maximal distance; the distance between $\boldsymbol{v}$ and $\boldsymbol{t}$ is listed in Table 1, $\left|\left|\boldsymbol{v}-\boldsymbol{t}\right|\right|_{105} \approx 438.1 \times \frac{1}{2}\min\left(||\boldsymbol{b}_i^*||\right)_{105}$. The minimal and maximal distances between $\boldsymbol{v}$ and $\boldsymbol{t}$ obtained in our experiments and the corresponding $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ are also listed in Table 1.

The corresponding lattice ranks are 2 and 200, and $\left|\left|v-t\right|\right|_2 \approx 7.1 \times 10^{38} \times \frac{1}{2}\min\left(||b_i^*||\right)_2$ and $\left|\left|v-t\right|\right|_{200} \approx 698 \times \frac{1}{2}\min\left(||b_i^*||\right)_{200}$.

The ratio of $||v-t||$ to $\frac{1}{2}\min(\left|\left|b_i^*\right|\right|)$ can indicate the boundary of the lattice attack. The smaller the ratio, the more likely we were to have a successful attack. If the ratio was less than 1, we could obtain the secret key. The ratio in our experiments with the lattice rank from 26 to 200 is shown in Figure 3; the ratio with rank from 2 to 25 is not shown because it is too large and meaningless. We can see when the rank is under approximately 50, the ratio declines quickly with the growth of the rank. When the rank is between approximately 50 and 70, the ratio declines slowly. When the rank is greater than approximately 70, the ratio is basically stable. This means, the ratio does not always decrease with increasing lattice rank. The minimal ratio is 424.0 in our experiments with a lattice rank of 84, but the distance $||v-t||_{84}$ is still much longer than $\frac{1}{2}\min\left(||b_i^*||\right)_{84}$. According to Theorem 1, the Babai algorithm cannot output the correct closest vector. In the next section, we tried to increase the minimal length of $b_i^*$.

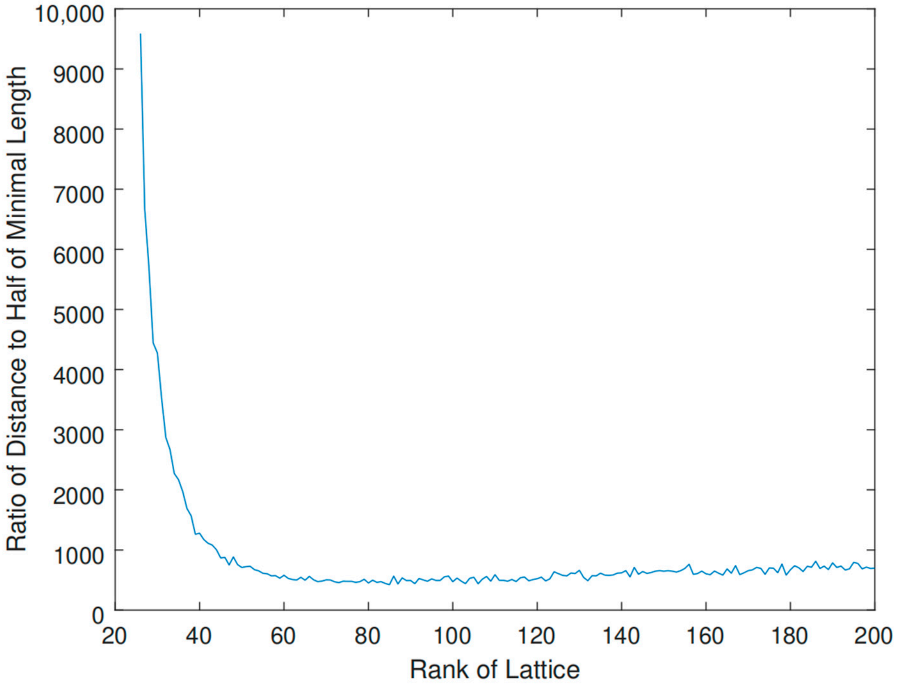

**Figure 3.** The ratio of $||v-t||$ to $\frac{1}{2}\min(\left|\left|b_i^*\right|\right|)$ for Equation (7).

## 6. Doubling Lattice Basis

To increase $\frac{1}{2}\min(\left|\left|b_i^*\right|\right|)$ and extend the boundary of attack, we tried a variant of the lattice basis construction called the double basis. For Equation (5) with one signature, both sides of the equation were multiplied by $-1$, generating a new equation for each $0 \le i \le n-1$:

$$(-C_i)(-\alpha) + (-z_i q) - (-D_i) = (-k_i) \tag{13}$$

We added these new equations to the joint Equation (6), the new joint equations are Equation (14), and the form of matrix and vector operations is expressed by Equation (15). Now, with the same number of signatures for Equation (7), $n-1$, we could construct a lattice with rank $2n-1$, which almost doubles the original rank $n$.

$$\begin{cases} (-1)(-\alpha) - 0 = \alpha \ (\mathrm{i} = 0) \\ C_i(-\alpha) + z_i \mathrm{q} - D_i = k_i \ (i = 1, 2, \ldots, n-1) \\ (-C_i)(-\alpha) + (-z_i q) - (-D_i) = (-k_i) \ (i = 1, 2, \ldots, n-1) \end{cases} \tag{14}$$

$$\begin{pmatrix} -1 & & & & & \\ C_1 & q & & & & \\ -C_1 & & q & & & \\ \vdots & & & \ddots & & \\ C_{n-1} & & & & q & \\ -C_{m-1} & & & & & q \end{pmatrix} \begin{pmatrix} -\alpha \\ z_1 \\ -z_1 \\ \vdots \\ z_{n-1} \\ -z_{m-1} \end{pmatrix} - \begin{pmatrix} 0 \\ D_1 \\ -D_1 \\ \vdots \\ D_{n-1} \\ -D_{m-1} \end{pmatrix} = \begin{pmatrix} \alpha \\ k_1 \\ -k_1 \\ \vdots \\ k_{n-1} \\ -k_{m-1} \end{pmatrix} \tag{15}$$

The double basis is:

$$B = \{\boldsymbol{b}_0, \boldsymbol{b}_1, ..., \boldsymbol{b}_{n-1}\} \tag{16}$$

where $\boldsymbol{b}_0 = (-1, C_1, -C_1, ..., C_{n-1}, -C_{n-1})^T$, $\boldsymbol{b}_1 = (0, q, 0, ..., 0)^T$, ... , $\boldsymbol{b}_{n-1} = (0, ..., 0, q)^T$. For the constructed variant lattice, the target vector is:

$$t = (t_0, ..., t_{2n-2})^T = (0, D_1, -D_1, ..., D_{n-1}, -D_{n-1})^T \tag{17}$$

The distance between the correct $\boldsymbol{v}$ and $\boldsymbol{t}$ is:

$$||\boldsymbol{v} - \boldsymbol{t}|| = \sqrt{\alpha^2 + \sum_{1}^{n-1} 2k_i^2} \tag{18}$$

We carried out similar experiments in Section 5 to obtain the minimal length of $\boldsymbol{b}_i^* (0 \leq i \leq 2n-1)$ for the newly constructed double lattice basis. Both the bit length of $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ and the distance from the correct vector to the target for Equation (15) in our experiments are shown in Figure 4. They exhibit the same change trend with the bit length for the original basis in Figure 2.

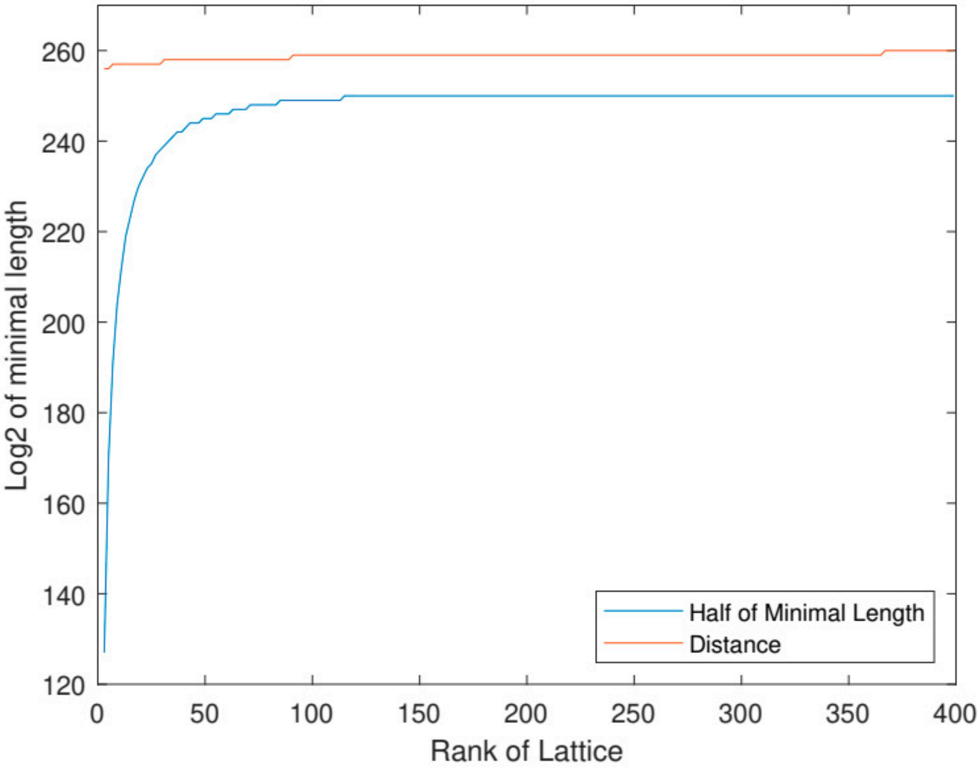

**Figure 4.** The bit length of the minimal $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ and the distance between $\boldsymbol{v}$ and $\boldsymbol{t}$ for Equation (15).

The maximum of $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ with the double basis obtained in our experiments is shown in Table 2, and its corresponding bit length is 250 with lattice rank 371. The maximum of $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ approximately equals $0.019q$ increased 35.9% over the original basis for Equation (7). Although we still used the 199 generated signatures in Section 5, we could obtain a larger $\max(\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|))$ by constructing the double basis. When the rank is 371, the distance between $\boldsymbol{v}$ and $\boldsymbol{t}$ is listed in Table 2. Additionally, listed in Table 2, for the double basis, are the minimal and maximal distances between $\boldsymbol{v}$ and $\boldsymbol{t}$ obtained in our experiments and the corresponding $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$, where the corresponding lattice ranks are 3 and 399.

**Table 2.** Some $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ s and some distances between $\boldsymbol{v}$ and $\boldsymbol{t}$ for Equation (15).

| Item | Value | Bit Length | Rank |
|---|---|---|---|
| $\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|)$ | 0X87CA530003E630A584C5268A5D19E160 | 127 | 3 (1 signature) |
| $\max(\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|))$ | 0X4D40A83F3F42DD05934BEF7FB76C4F10E0<br>D81842975B37DAE526F312C70FCE3 | 250 | 371 (185 signature) |
| $\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|)$ | 0X433FE3970F1C7375606C6D12F52994C18ED<br>9F40516086D8E33B9A88A96D6FA4 | 250 | 399 (199 signature) |
| $q$ | 0xFFFFFFFF00000000FFFFFFFFFFFFFFFF<br>BCE6FAADA7179E84F3B9CAC2FC632551 | 256 | |
| $\min(\|\|\boldsymbol{v}-\boldsymbol{t}\|\|)$ | 0XE82D9B5A290322FFD26F0DF3301FA45C0E<br>01B2BC501E07E8B879EA87B9DBB51E | 256 | 3 (1 signature) |
| $\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|)$ | 0XB7217DB2B20AA2438103A19D84C03438716<br>867F78FE7732EFB99EEA38B8B253E4 | 259 | 371 (185 signature) |
| $\max(\frac{1}{2}\min(\left\|\left\|\boldsymbol{b}_i^*\right\|\right\|))$ | 0XBD263DC52AE0729C3E2D9CF3CF63F0F3E<br>368632FC388933DEDD76A3C6EECCE08D | 260 | 399 (199 signature) |

For the double basis: $\left|\left|\boldsymbol{v}-\boldsymbol{t}\right|\right|_3 \approx 5.8 \times 10^{38} \times \frac{1}{2}\min(||\boldsymbol{b}_i^*||)_3$, $||\boldsymbol{v}-\boldsymbol{t}||_{371} \approx 606.9\times \frac{1}{2}\min(||\boldsymbol{b}_i^*||)_{371}$, and $\left|\left|\boldsymbol{v}-\boldsymbol{t}\right|\right|_{399} \approx 720.0 \times \frac{1}{2}\min(||\boldsymbol{b}_i^*||)_{399}$. The ratio of $||\boldsymbol{v}-\boldsymbol{t}||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ in our experiments with the lattice rank from 53 to 399 (only odd numbers) is shown in Figure 5; the ratio with rank from 3 to 51 (only odd numbers) is not shown because it is too large and meaningless. It has the same change trend with the ratio for the original basis in Figure 3. The minimal ratio is 433.0 in our experiments with lattice rank 171, but the distance $||\boldsymbol{v}-\boldsymbol{t}||_{171}$ is still much longer than $\frac{1}{2}\min(||\boldsymbol{b}_i^*||)_{171}$. According to Theorem 1, the Babai algorithm cannot output the correct closest vector. Although $\max(\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|))$ can be extended by constructing the double basis, it cannot reduce the minimal ratio of $||\boldsymbol{v}-\boldsymbol{t}||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$. In the next section, we tried moving the target vector into the boundary of the attack.

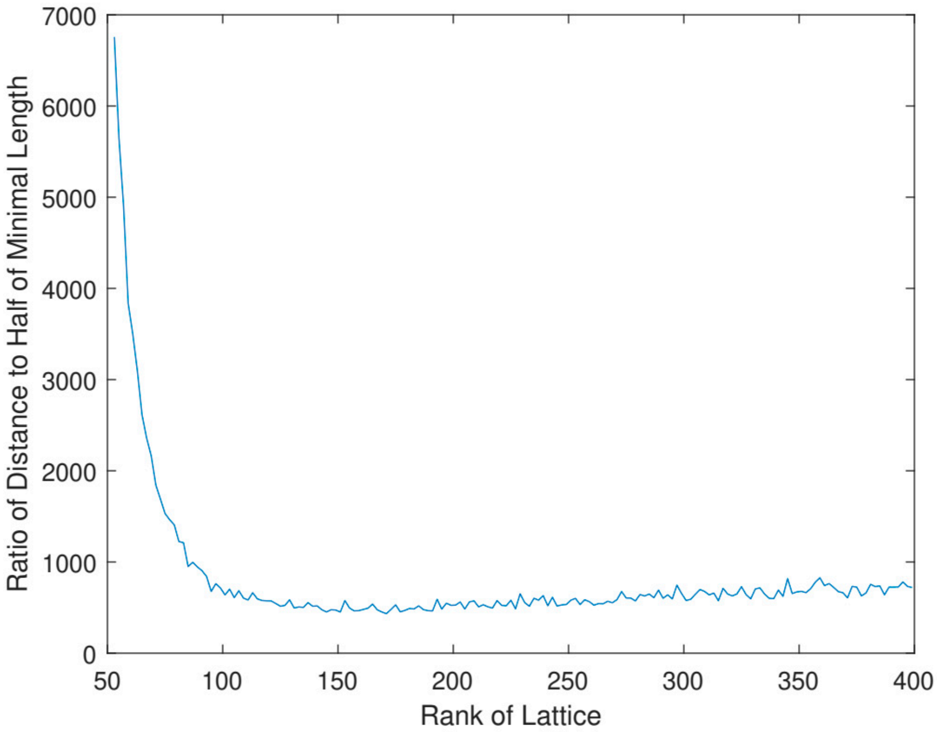

**Figure 5.** The ratio of $||\boldsymbol{v}-\boldsymbol{t}||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ for Equation (15).

## 7. Moving Target Vector

In this section, except for extending the boundary for a successful attack, we tried moving the target vector into the boundary of attack.

The distance between the correct vector $\boldsymbol{v}$ and the original target $\boldsymbol{t} = (0, D_1, ..., D_{n-1})^T$ or $(0, D_1, -D_1 ..., D_{n-1}, -D_{n-1})^T$ can be calculated by Equation (11) or (18). The idea destination $\boldsymbol{d}$ of the target vector is to stay in the boundary of a successful attack.

Let:

$$w = \begin{cases} \frac{1}{2\sqrt{n}}\min(\boldsymbol{b}_i^*), \text{for equation}(7) \\ \frac{1}{2\sqrt{2n-1}}\min(\boldsymbol{b}_i^*), \text{for equation}(15) \end{cases} \tag{19}$$

If the distance on every dimension between the correct $\boldsymbol{v}$ and $\boldsymbol{d}$ is less than $w$, then the total distance must be less than $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$, which is derived from Equation (20). The idea target moving with the original basis for Equation (7) is expressed by Equation (21) and the idea target moving with the double basis for Equation (15) is expressed by Equation (22).

However, in real situations, we do not know the private key $\alpha$ and the random nonces $k_0, k_1, ..., k_{n-1}$, thus we cannot have an ideal target moving. We can only attempt to move the target vector $\boldsymbol{t}$ with the step size $w$. $\boldsymbol{t}$ with the original basis in Equation (7) has n dimensions; the position on each dimension should be moved by a different number of steps. Then, the final distance between the moved $\boldsymbol{t}$, $d$, and the correct $\boldsymbol{v}$ could be less than $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$. Since $0 < \alpha < q$ and $0 < k_i < q$, $(0 \leq i \leq n-1)$, the maximum number of attempts on each dimension are $(\frac{q}{w})$ and the total number of attempts could be $(\frac{q}{w})^n$. Although $\boldsymbol{t}$ with the double basis in Equation (15) has $2n-1$ dimensions and $D_i$ and $-D_i(0 \leq i \leq n-1)$ have the same length, we can move the target with the same number of steps on these two dimensions in our attempts. Then, the total number of attempts is still $(\frac{q}{w})^n$.

Sections 5 and 6 continue the experiments. Figure 6 depicts the bit length of the step size $w$ obtained in our trials using the original foundation in Equation (7). Figure 7 depicts the bit length using the double basis in Equation (15). They exhibit the same change trend of $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ in Figures 2 and 4. The maximum step size obtained in our experiments is listed in Table 3. The corresponding number of attempts ($\frac{q}{w}$) on one dimension is shown in Figure 8 (the number with rank from 2 to 25 is not shown because it is too large and meaningless) and Figure 9 (the number with rank from 3 to 51, only odd numbers, is not shown because it is too large and meaningless). They have a similar change trend with the ratio of $||\boldsymbol{v}-\boldsymbol{t}||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ in Figures 3 and 5. The minimum number of attempts ($\frac{q}{w}$) is 690 and 709 for the original and double basis, respectively. Although $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ can be extended by constructing the double basis, it cannot extend the step size and reduce the number of attempts.

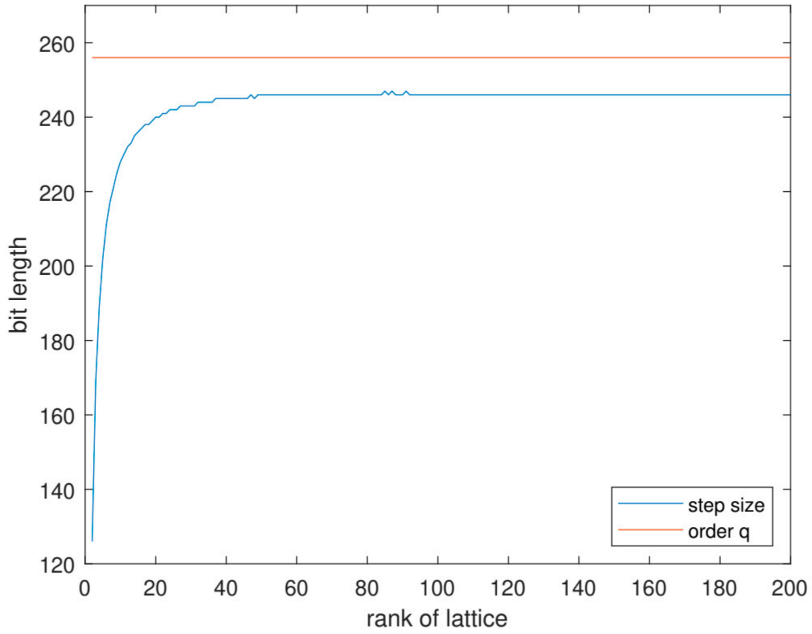

**Figure 6.** The bit length of the step size with the original basis for Equation (7).

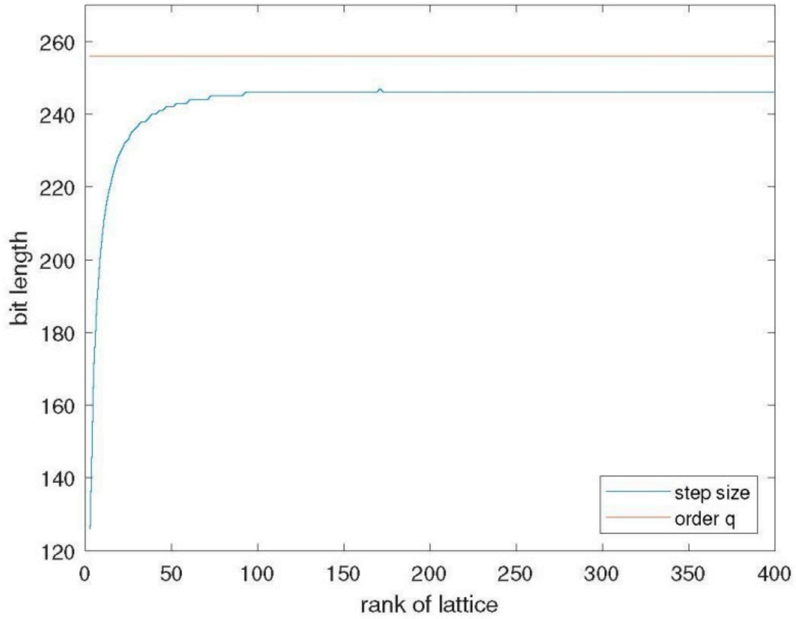

**Figure 7.** The bit length of the step size with the double basis for Equation (15).

**Table 3.** The value of the maximal step size.

| Item | Value | Bit Length | Rank |
|---|---|---|---|
| with original basis for Equation (7) | 0X5EDAB1E5C6048F0D24629C92405DA8A8E D164889CBCB931F2E417A1341AE4A | 247 | 85 (84 signatures) |
| with double basis for Equation (15) | 0X5C65C707378D97AABA31AB46F37EC84E6 1F4AD58E1931FE9DC789859B5B58C | 247 | 177 (85 signatures) |

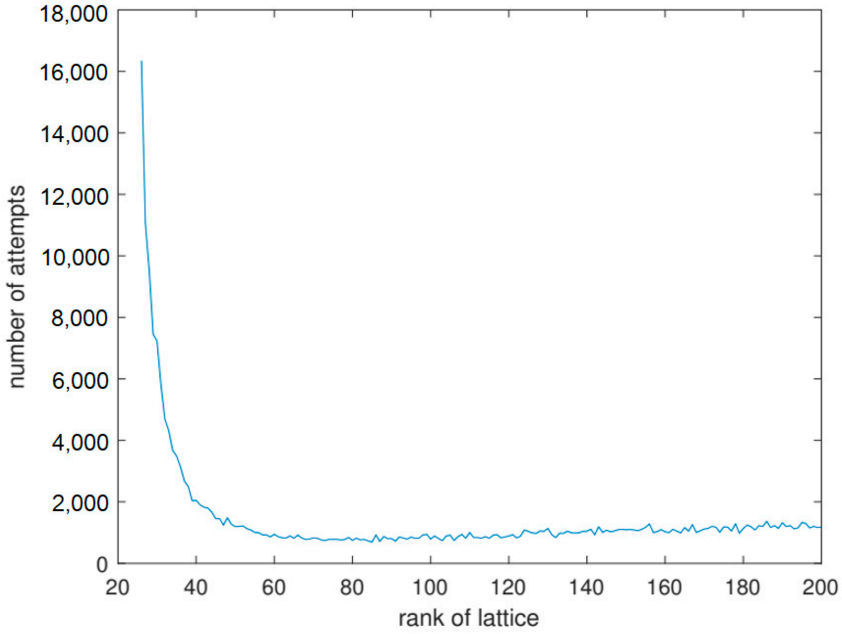

**Figure 8.** The number of attempts in one dimension with the original basis for Equation (7).

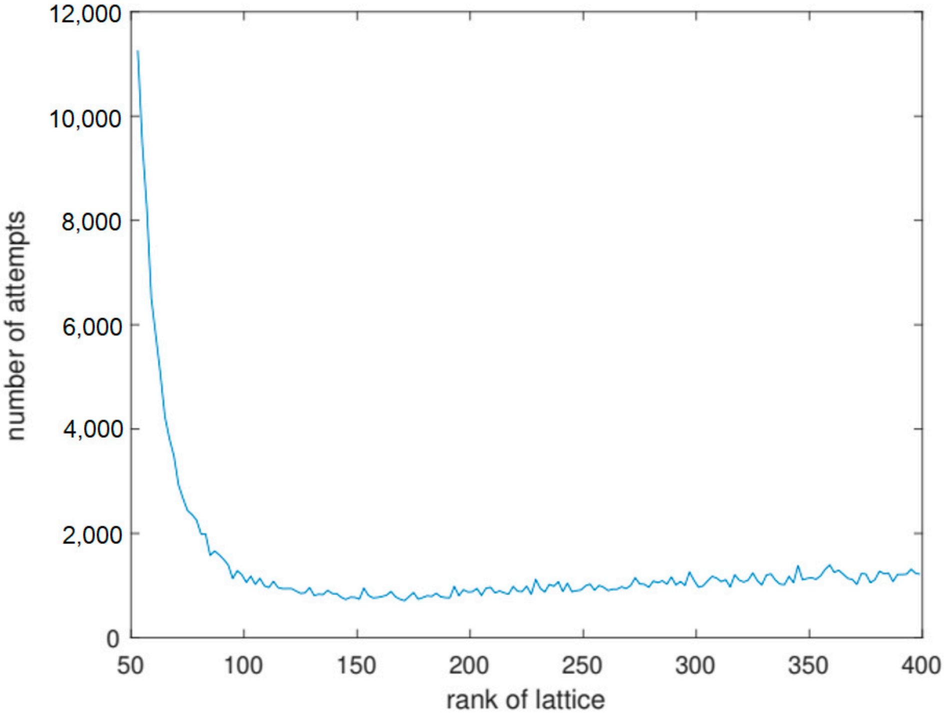

**Figure 9.** The number of attempts in one dimension with the double basis for Equation (15).

The total number of attempts $(\frac{q}{w})^n$ truly reflects the difficulty of the lattice attack; its bit length is shown in Figures 10 and 11. Both increase with increasing lattice rank. The minimal number of total attempts is $1.6 \times 10^{78} \approx 2^{260}$ and $1.2 \times 10^{78} \approx 2^{259}$ with rank 2 for the original basis and rank 3 for the double basis, respectively, which is more difficult than key guessing via use of brute force with $2^{256}$. However, if some leakage information of the random nonces is known, the lattice attack has a great advantage.

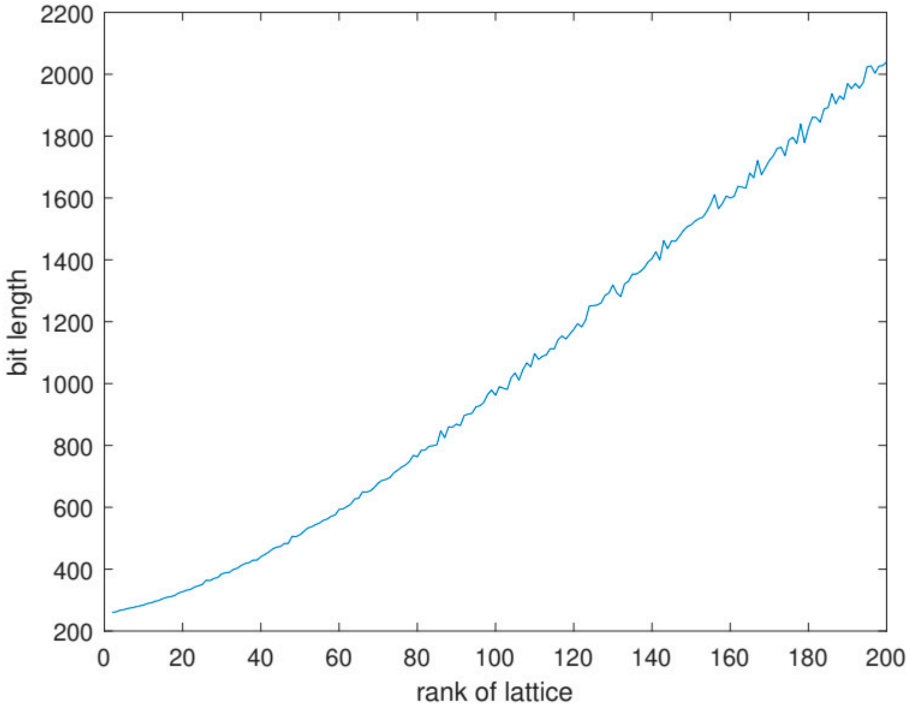

**Figure 10.** The bit length of the total number of attempts with the original basis for Equation (7).

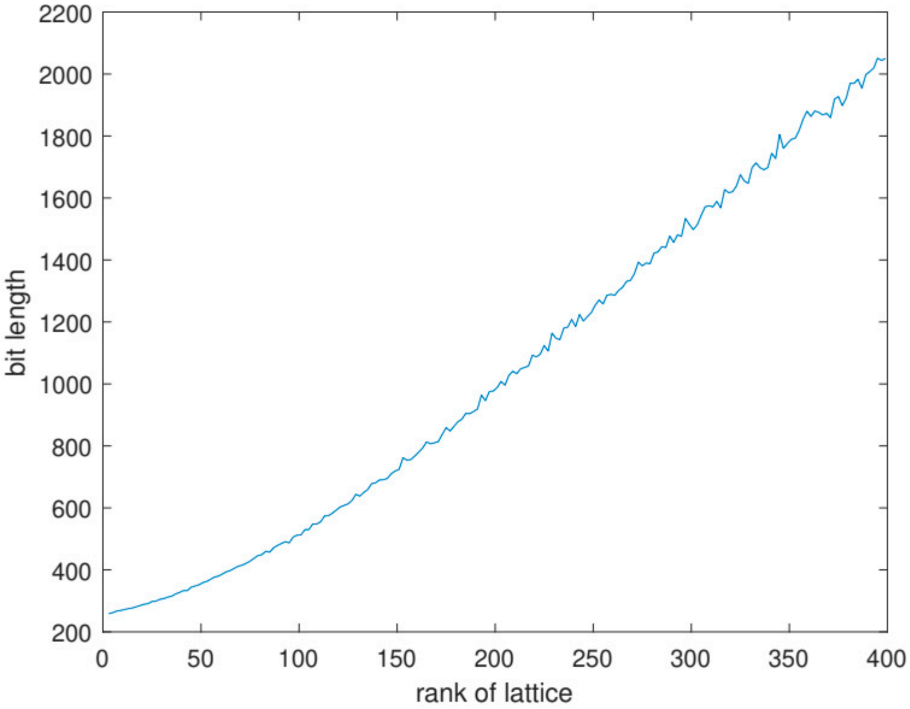

**Figure 11.** The bit length of the total number of attempts with the double basis for Equation (15).

To observe how close the moved target vector $\boldsymbol{d}$ was to the boundary of lattice attack, we attempted moving the target vector with the maximal step size obtained in our experiments in only two directions: first, for the private key dimension, and second, for the other random nonce dimensions. The private key dimension represents $t_0$ of the target vector $\boldsymbol{t}$ in Equation (10) or (17) and the random nonce dimensions represent $t_i(i > 0)$. The move starts from $\left[\frac{q}{w}\right]$ to 0 steps in each direction. The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ obtained in our experiments is shown in Figures 12 and 13, which reveals the closeness to the attack boundary. Due to the number of attempts being too large to display all ratios, only some samplings were selected; the sampling rule equated to the selection of one ratio for every 1000 attempts. The "index" in Figures 12 and 13 represents the sample ratio index. We can see the distance from the moved target vector $\boldsymbol{d}$ to the boundary was changed with the change in step size in two directions. For the closest moved target vector $\boldsymbol{d}$ obtained in our experiments, the ratio of $||\boldsymbol{v} - \boldsymbol{d}||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ is 179.0 and 187.0 with the original basis for Equation (7) and the double basis for Equation (15), which is significantly less than the initial target's minimum ratio, 424.0 and 433.0, of $||\boldsymbol{v} - \boldsymbol{t}||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ in Sections 5 and 6.

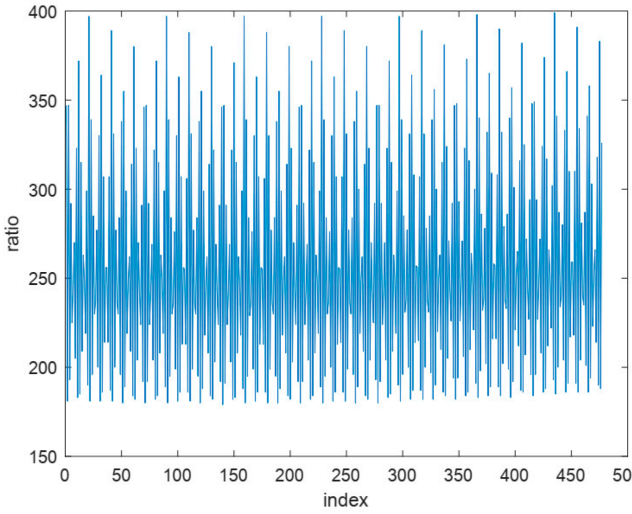

**Figure 12.** The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ with the original basis for Equation (7).

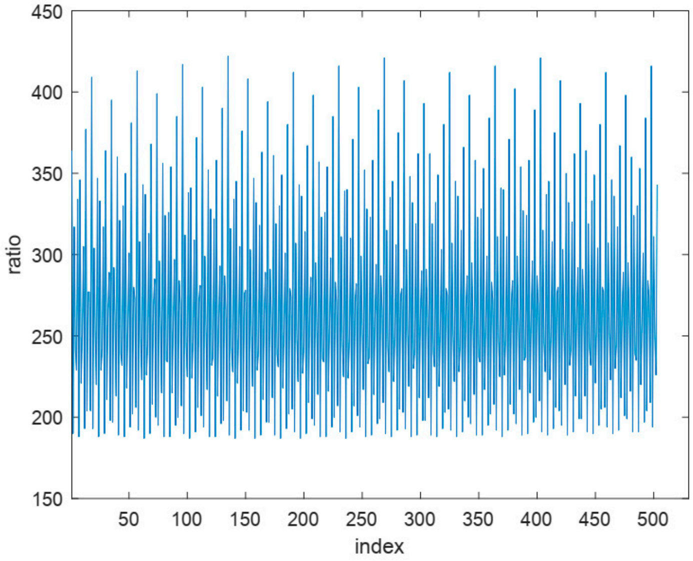

**Figure 13.** The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ with the double basis for Equation (15).

It is obvious that the more directions in which we attempted to move, the closer the distance from the moved target vector to the attack boundary became; it also took more time to finish the attempts. Our experiments ran on one VMWare Ubuntu 18.04.6 TLS with 1 processor (4 cores) and 4 gigabytes (GB) Random Access Memory (RAM). The host physical machine ran Windows 10 with Intel(R) Core(TM) i7-9700 CPU @ 3.00 GHz and 32 GB RAM(Intel Corporation in Santa Clara, California, the United States). With the original basis (84 signatures, lattice rank 85) and double basis (85 signatures, lattice rank 171), it took 247.7 s and 354.4 s to complete all moving attempts in two directions, respectively. The idea target moving, expressed by Equations (21) and (22), represents moving with different step numbers on each dimension. It was impossible to finish the idea moving attempts in such a limited amount of time.

$$||\boldsymbol{v}-\boldsymbol{d}|| < \begin{cases} \sqrt{\sum\limits_{0}^{n-1} w^2} = \frac{1}{2}\min(||\boldsymbol{b}_i^*||), \text{ for equation 7} \\ \sqrt{\sum\limits_{0}^{2n-2} w^2} = \frac{1}{2}\min(||\boldsymbol{b}_i^*||), \text{ for equation 15} \end{cases} \tag{20}$$

$$\begin{pmatrix} -1 & & & \\ C_1 & q & & \\ \vdots & & \ddots & \\ C_{n-1} & & & q \end{pmatrix} \begin{pmatrix} -\alpha \\ z_1 \\ \vdots \\ z_{n-1} \end{pmatrix} - \begin{pmatrix} 0+[\alpha/w]w \\ D_1+[k_1/w]w \\ \vdots \\ D_{n-1}+[k_{n-1}/w]w \end{pmatrix} = \begin{pmatrix} \alpha \mathrm{mod} w \\ k_1 \mathrm{mod} w \\ \vdots \\ k_{n-1} \mathrm{mod} w \end{pmatrix} \tag{21}$$

$$\begin{pmatrix} -1 & & & & & \\ C_1 & q & & & & \\ -C_1 & & q & & & \\ \vdots & & & \ddots & & \\ C_{n-1} & & & & q & \\ -C_{m-1} & & & & & q \end{pmatrix} \begin{pmatrix} -\alpha \\ z_1 \\ -z_1 \\ \vdots \\ z_{n-1} \\ -z_{m-1} \end{pmatrix} - \begin{pmatrix} 0+[\alpha/w]w \\ D_1+[k_1/w]w \\ -D_1-[k_1/w]w \\ \vdots \\ D_{n-1}+[k_{n-1}/w]w \\ -D_{m-1}-[k_{n-1}/w]w \end{pmatrix} = \begin{pmatrix} \alpha \mathrm{mod} w \\ k_1 \mathrm{mod} w \\ -k_1 \mathrm{mod} w \\ \vdots \\ k_{n-1} \mathrm{mod} w \\ -k_{m-1} \mathrm{mod} w \end{pmatrix} \tag{22}$$

Suppose we know some leakage information of the random nonces, we could filter out the signatures whose one MSB or two MSBs of the random nonce is 0. We performed similar experiments of attempting to move the target in two directions with the filtered signatures. The sample ratios of $||\boldsymbol{v}-d||$ to $\frac{1}{2}\min(||\boldsymbol{b}_i^*||)$ obtained in our experiments are shown in Figures 14–17. For the closest moved target vector $d$ obtained in our experiments, the ratio of $||\boldsymbol{v}-d||$ to $\frac{1}{2}\min(||\boldsymbol{b}_i^*||)$ is 45.6 with the original basis and two clear MSBs of the random nonces. We can see where the greater the random nonce's MSBs were clear, the closer the distance between the moved target vector and the attack boundary would become, but it is also more difficult to obtain more leakage information of the random nonce.

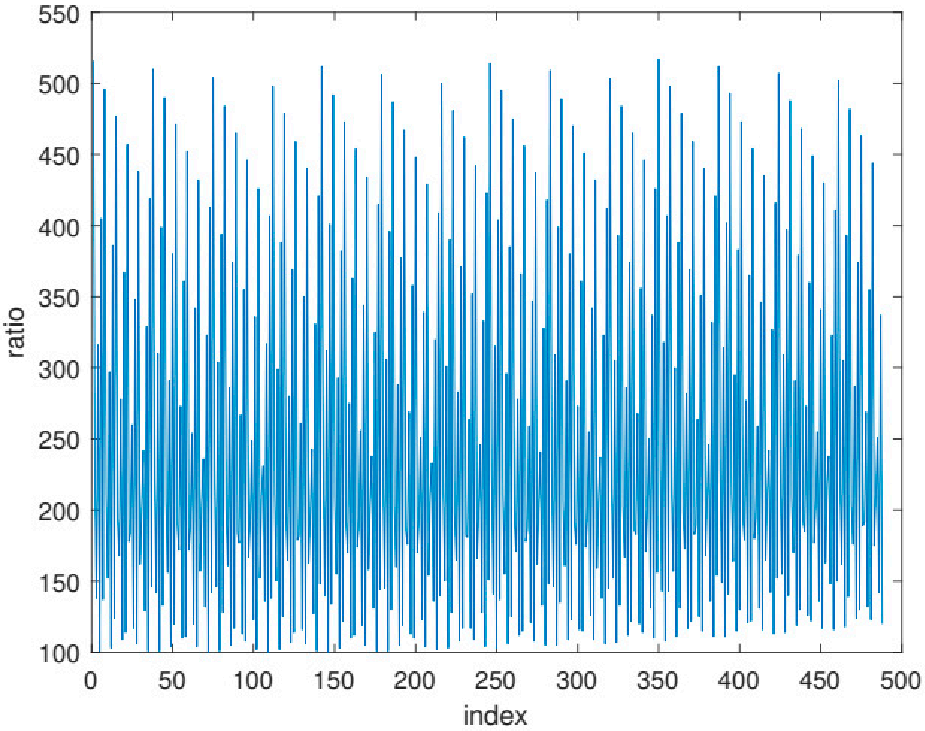

**Figure 14.** The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|b_i^*\right|\right|)$ with the original basis for Equation (7) and one clear MSB.

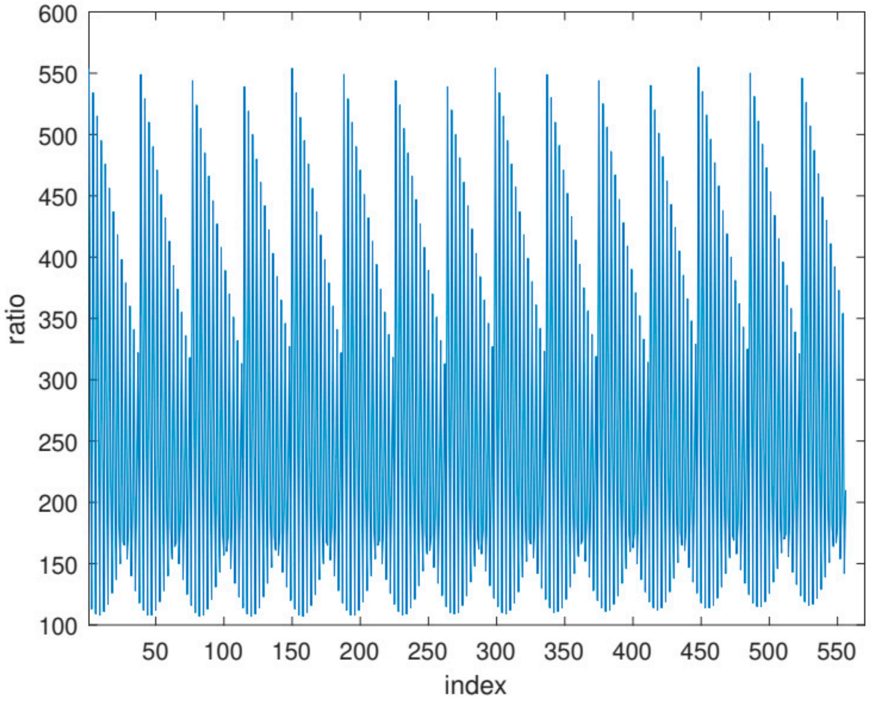

**Figure 15.** The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|b_i^*\right|\right|)$ with the double basis for Equation (15) and one clear MSB.

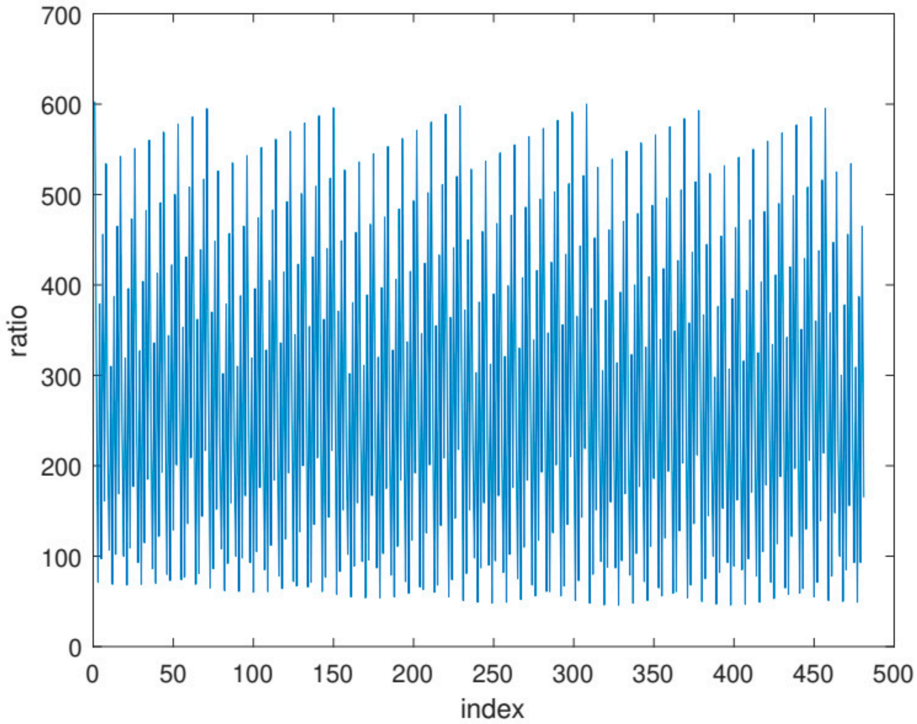

**Figure 16.** The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ with the original basis for Equation (7) and two clear MSBs.

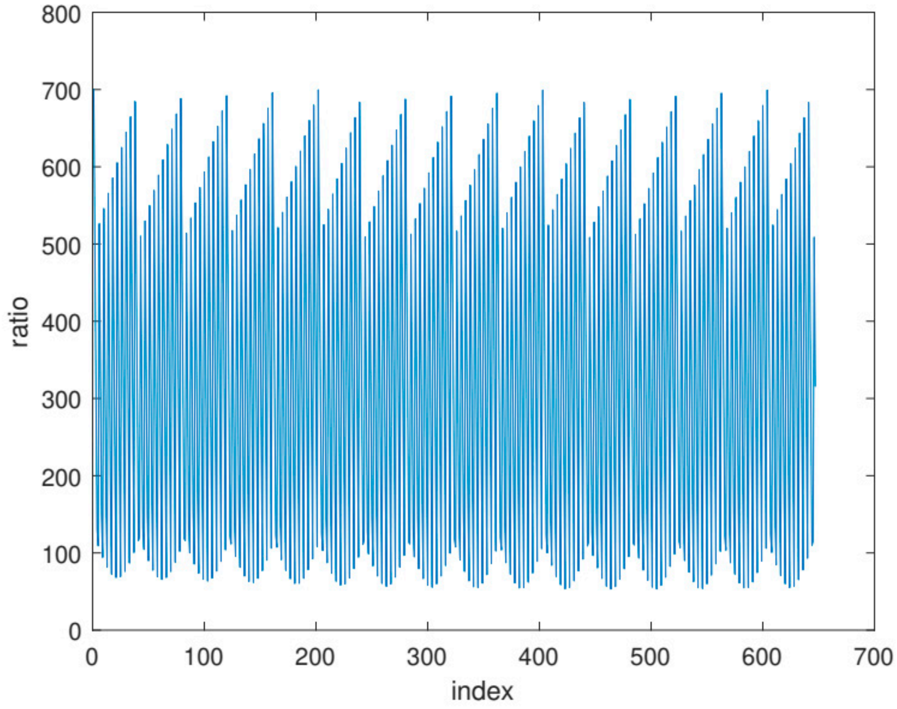

**Figure 17.** The ratio of $||v - d||$ to $\frac{1}{2}\min(\left|\left|\boldsymbol{b}_i^*\right|\right|)$ with the double basis for Equation (15) and two clear MSBs.

## 8. Conclusions and Future Work

The findings presented in this paper have significant implications for enhancing the overall security of blockchain systems, including the use of symmetric encryption to protect communication between parties. By establishing the boundary for a successful lattice attack on ECDSA and implementing effective measures to mitigate such attacks, including the use of heuristics and the exploration of alternative basis structures, this study contributes to strengthening the security of blockchain systems. As a result, data encrypted using symmetric encryption methods in the blockchain network are better protected against potential attacks, ensuring the confidentiality and integrity of transmitted data between parties.

To improve the success rate of lattice attacks, the authors developed a variant of the lattice basis, the "double basis", which allows for expansion of the boundary or reduction in the required signatures by almost half with the same lattice rank, achieved by multiplying both sides of one signature equation by one. In addition, the authors utilized a heuristic approach and conducted experiments on TPM2.0 ECDSA to move the target vector closer to the boundary in different directions with a step size determined by half the minimal length of the orthogonal vectors. The distance from the closest moved target vector to the boundary was reduced in the authors' experiments with a ratio from 424 to 179, relative to the minimal length of orthogonal vectors. The experiment was conducted using the original basis and 84 signatures (lattice rank 85), and the moving attempts in two directions could be completed in approximately 247.7 s on the experiment computer. If the authors had filtered out the signatures with random nonces having two clear MSBs, the ratio could be reduced even further by only attempting to move in two directions.

In the future, we will try to form a new variant of the lattice basis to expand the boundary and attempt step size further, which requires fewer signatures. We may move the target vector with the same step number in all directions. Then, the total number of attempts could be reduced from $(\frac{q}{w})^n$ to $(\frac{q}{w})^2$.

**Author Contributions:** Conceptualization, B.Z. and Z.W.; methodology, X.Z.; investigation, S.W.; data curation, X.Z. and F.Y.; writing—original draft preparation, S.W. and Y.J.; writing—review and editing, B.Z. and Y.J.; supervision, B.Z. and F.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is supported by the science and technology project of State Grid Corporation of China: "Research on security technology of substation embedded automation Equipment based on trusted computing" (5108-202240041A-1-1-ZN).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data will be provided upon request.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Boneh, D.; Durfee, G. Cryptanalysis of RSA with private key d less than N/sup 0.292/. *IEEE Trans. Inf. Theory* **2000**, *46*, 1339–1349. [CrossRef]
2. Mushtaq, M.; Mukhtar, M.A.; Lapotre, V.; Bhatti, M.K.; Gogniat, G. Winter is here! A decade of cache-based side-channel attacks, detection & mitigation for RSA. *Inf. Syst.* **2020**, *92*, 101524.
3. Howgrave-Graham, N.A.; Smart, N.P. Lattice Attacks on Digital Signature Schemes. *Des. Codes Cryptogr.* **2001**, *23*, 283–290. [CrossRef]
4. Jingwei, H.; Senlin, L.; Limin, P. Computer-aided intelligent design using deep multiobjective cooperative optimization algorithm. *Future Gener. Comput. Syst.* **2021**, *124*, 49–53.
5. Bisheh-Niasar, M.; Azarderakhsh, R.; Mozaffari-Kermani, M. Cryptographic Accelerators for Digital Signature Based on Ed25519. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2021**, *29*, 1297–1305. [CrossRef]
6. Dimopoulos, C.; Fournaris, A.P.; Koufopavlou, O. Machine Learning Attacks and Countermeasures on Hardware Binary Edwards Curve Scalar Multipliers. *J. Sens. Actuator Netw.* **2021**, *10*, 56. [CrossRef]
7. Nguyen, P.Q.; Shparlinski, I. The Insecurity of the Elliptic Curve Digital Signature Algorithm with Partially Known Nonces. *Des. Codes Cryptogr.* **2003**, *30*, 201–217. [CrossRef]
8. Ma, Z.; Li, B.; Cai, Q.; Yang, J. Applications and developments of the lattice attack in side channel attacks. In Proceedings of the International Conference on Applied Cryptography and Network Security, Rome, Italy, 22–25 June 2020; pp. 435–452.
9. Lenstra, A.K.; Lenstra, H.W.; Lovász, L. Factoring polynomials with rational coefficients. *Math. Ann.* **1982**, *261*, 515–534. [CrossRef]

10. Mehibel, N.; Hamadouche, M. A new enhancement of elliptic curve digital signature algorithm. *J. Discret. Math. Sci. Cryptogr.* **2020**, *23*, 743–757. [CrossRef]
11. Weiser, S.; Schrammel, D.; Bodner, L.; Spreitzer, R. Big numbers-big troubles: Systematically analyzing nonce leakage in (ec) dsa implementations. In Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), Boston, MA, USA, 12–14 August 2020; pp. 1767–1784.
12. Brumley, B.B.; Tuveri, N. Remote timing attacks are still practical. In Proceedings of the European Symposium on Research in Computer Security, Leuven, Belgium, 12–14 September 2011; pp. 355–371.
13. Moghimi, D.; Sunar, B.; Eisenbarth, T.; Heninger, N. Tpm-fail: Tpm meets timing and lattice attacks. In Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), Boston, MA, USA, 12–14 August 2020; pp. 2057–2073.
14. Eskandari, Z.; Ghaemi Bafghi, A. Extension of cube attack with probabilistic equations and its application on cryptanalysis of KATAN cipher. *ISC Int. J. Inf. Secur.* **2020**, *12*, 1–12.
15. Brumley, B.B.; Hakala, R.M. Cache-timing template attacks. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, 6–10 December 2009; pp. 667–684.
16. Yarom, Y.; Falkner, K. Flush + reload: A high resolution, low noise, l3 cache side-channel attack. In Proceedings of the 23rd USENIX Security Symposium (USENIX Security 14), San Diego, CA, USA, 20–22 August 2014; pp. 719–732.
17. Belgarric, P.; Fouque, P.-A.; Macario-Rat, G.; Tibouchi, M. Side-channel analysis of weierstrass and koblitz curve ecdsa on android smartphones. In Proceedings of the Cryptographers' Track at the RSA Conference, Francisco, CA, USA, 29 February–4 March 2016; pp. 236–252.
18. Fan, S.; Wang, W.; Cheng, Q. Attacking openssl implementation of ecdsa with a few signatures. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS'16, Vienna, Austria, 24–28 October 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 1505–1515.
19. Wang, W.; Fan, S. Attacking OpenSSL ECDSA with a small amount of side-channel information. *Sci. China Inf. Sci.* **2017**, *61*, 032105. [CrossRef]
20. Albrecht, M.R.; Heninger, N. On bounded distance decoding with predicate: Breaking the "lattice barrier" for the hidden number problem. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, 17–21 October 2021; pp. 528–558.
21. Karthik, C.S.; Manurangsi, P. On closest pair in euclidean metric: Monochromatic is as hard as bichromatic. *Combinatorica* **2020**, *40*, 539–573. [CrossRef]
22. Babai, L. On lovász'lattice reduction and the nearest lattice point problem. *Combinatorica* **1986**, *6*, 1–13. [CrossRef]
23. Zhang, K.; Xu, S.; Gu, D.; Gu, H.; Liu, J.; Guo, Z.; Liu, R.; Liu, L.; Hu, X. Practical partial-nonce-exposure attack on ecc algorithm. In Proceedings of the 2017 13th International Conference on Computational Intelligence and Security (CIS), Hong Kong, China, 15–18 December 2017; pp. 248–252.
24. Nguyen, P.Q.; Tibouchi, M. Lattice-based fault attacks on signatures. In *Fault Analysis in Cryptography*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 201–220.
25. Cao, W.; Feng, J.; Chen, H.; Zhu, S.; Wu, W.; Han, X.; Zheng, X. Two lattice-based differential fault attacks against ecdsa with wnaf algorithm. In Proceedings of the ICISC 2015, Seoul, Republic of Korea, 25–27 November 2015; pp. 297–313.
26. Boneh, D.; Venkatesan, R. Hardness of computing the most significant bits of secret keys in diffie-hellman and related schemes. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 1996; pp. 129–142.
27. Espitau, T.; Kirchner, P. The nearest-colattice algorithm: Time-approximation tradeoff for approx-CVP. *Open Book Ser.* **2020**, *4*, 251–266. [CrossRef]
28. Chaudhary, R.; Aujla, G.S.; Kumar, N.; Zeadally, S. Lattice-Based Public Key Cryptosystem for Internet of Things Environment: Challenges and Solutions. *IEEE Internet Things J.* **2018**, *6*, 4897–4909. [CrossRef]
29. Min, S.; Tan, L. A TPM2. 0 Key Migration Protocol and Security Analysis. *Acta Electonica Sin.* **2019**, *47*, 1449.
30. Trusted Computing Group's (tcg) tpm2 Software Stack, September 2021. Available online: Github.com/tpm2-software/tpm2tss (accessed on 14 March 2023).
31. Trusted Platform Module (tpm2.0) Tools, September 2021. Available online: Github.com/tpm2-software/tpm2tools (accessed on 14 March 2023).

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.