Side-Channel Attacks on Post-Quantum Signature Schemes based on Multivariate Quadratic Equations

- Rainbow and UOV -

Aesun Park[1], Kyung-Ah Shim[2*], Namhun Koo[2] and Dong-Guk Han[1†]

[1] Department of Financial Information Security, Kookmin University, Seoul, Republic of Korea
{aesons,christa}@kookmin.ac.kr

[2] Division of Mathematical Modeling, National Institute for Mathematical Sciences, Daejeon, Republic of Korea {kashim,nhkoo}@nims.re.kr

Abstract. In this paper, we investigate the security of Rainbow and Unbalanced Oil-and-Vinegar (UOV) signature schemes based on multivariate quadratic equations, which is one of the most promising alternatives for post-quantum signature schemes, against side-channel attacks. We describe correlation power analysis (CPA) on the schemes that yield full secret key recoveries. First, we identify a secret leakage of secret affine maps S and T during matrix-vector products in signing when Rainbow is implemented with equivalent keys rather than random affine maps for optimal implementations. In this case, the simple structure of the equivalent keys leads to the retrieval of the entire secret affine map T. Next, we extend the full secret key recovery to the general case using random affine maps via a hybrid attack: after recovering S by performing CPA, we recover T by mounting algebraic key recovery attacks. We demonstrate how this leakage on Rainbow can be practically exploited on an 8-bit AVR microcontroller using CPA. Consequently, our CPA can be applied to Rainbow-like multi-layered schemes regardless of the use of the simple-structured equivalent keys and UOV-like single layer schemes with the implementations using the equivalent keys of the simple structure. This is the first result on the security of multivariate quadratic equations-based signature schemes using only CPA. Our result can be applied to Rainbow-like multi-layered schemes and UOV-like single layer schemes submitted to NIST for Post-Quantum Cryptography Standardization.

Keywords: Side-channel attack · Correlation power analysis · Key recovery attack · Multivariate-Quadratic problem

1 Introduction

Security of the most widely used public-key cryptosystems (PKCs), such as RSA and ECDSA, is based on the hardness of the integer factorization problem or the (elliptic curve) discrete logarithm problem. However, these hard problems are known to be solvable by Shor's quantum algorithm in polynomial time [Sho97]. This means that the widely-used PKCs are susceptible to a large-scale quantum computer. Several types of cryptographic primitives exist, which are believed to be secure against a quantum

*Corresponding Author. This research was supported by the National Institute for Mathematical Sciences funded by Ministry of Science and ICT of Korea (B18120000).

†This work was supported as part of Military Crypto Research Center(UD170109ED) funded by Defense Acquisition Program Administration(DAPA) and Agency for Defense Development(ADD).

IACR Transactions on Cryptographic Hardware and Embedded Systems ISSN 2569-2925, Vol. 2018, No. 3, pp. 500–523
DOI:10.13154/tches.v2018.i3.500-523

computer including lattice-based, code-based, hash-based, and multivariate quadratic equations (MQ)-based cryptography. These cryptographic primitives are believed to be resistant against both classical and quantum attacks, and this has been increased confidence in their adoptability as post-quantum alternatives. Recently, the National Security Agency (NSA) has updated its Suite B algorithms to explicitly emphasize the importance of the migration to post-quantum cryptographic algorithms [NSA15]. NIST has started initiatives to develop post-quantum cryptographic standards. Submissions to NIST's Post-Quantum Cryptography Standardization are for post-quantum public-key encryption, key exchange and digital signature [NIS16a].

Multivariate quadratic public-key cryptosystem (MQ-PKC) relies on the hardness of solving large systems of multivariate quadratic equations, called the MQ-problem. In MQ-PKC, a public key is given by a system of multivariate quadratic equations, and a trapdoor is hidden in secret affine layers of the affine-substitution (quadratic)-affine (ASA) structure. Security of this ASA structure depends on the hardness of the Extended Isomorphism of Polynomials (EIP) problem [Pat96]. To build MQ-PKC, one starts with an easily invertible quadratic map $\mathcal{F} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ called a central map. Then one composes it with two invertible affine maps $S : \mathbb{F}_q^m \to \mathbb{F}_q^m$ and $T : \mathbb{F}_q^n \to \mathbb{F}_q^n$ to hide the special structure of $\mathcal{F}$. A public key is $\mathcal{P} = S \circ \mathcal{F} \circ T$ and the secret key is $(S, \mathcal{F}, T)$ that allows the inversion of the public key. A main challenge in MQ-PKC is the selection of the nonlinear quadratic part $\mathcal{F}$ which is invertible.

In 1988, Imai and Matsumoto proposed the first MQ-based encryption scheme [MI88]. Since then, several MQ-schemes have been proposed, however, most MQ-schemes have been broken by targeting the IP-problem. There are two main types of signature schemes: HFEv- variants [PCG01, PCY$^+$15] and Unbalanced Oil-and-Vinegar (UOV) variants [KPG99, DS05]. The operations of UOV type MQ-schemes comprise matrices and vectors, and they are simple and computed on small fields. Hence, MQ-schemes do not require much computational resources. Consequently, they are attractive for use on resource-limited devices such as smart cards [BERW08, CCC$^+$09]. MQ-signature schemes surpass other alternatives in terms of speed and signature size. At CHES 2012, Czypek *et al.* [CHT12] demonstrated the feasibility of MQ-signature schemes on an 8-bit AVR microprocessor. They showed that the MQ-signature schemes, Rainbow, and enTTS outperform RSA and ECDSA in terms of speed. On the practical side, such an MQ-signature scheme is a promising alternative to classical schemes, such as RSA, DSA, and ECDSA.

Side-channel attacks (SCAs) focus on the capabilities of an attacker to break a cryptographic algorithm by exploiting vulnerabilities in the underlying implementations rather than its mathematical structure. SCAs can be divided into invasive, semi-invasive, and non-invasive attacks based on the interface that is exploited by the attack. An invasive attack has no limits to what is done on a cryptographic device, whereas a non-invasive attack does not physically transform the cryptographic device. Timing attacks [Koc96], power analysis (PA) [KJJ99], and electromagnetic (EM) attacks [GMO01] are non-invasive attacks. Further, the attacker can maliciously inject faults into the cryptographic algorithm and investigate the faulty outputs, which can reveal some information about the secret key. This attack is known as the fault attack which is a semi-invasive attack. It is well-known that any implementation of a cryptographic algorithm not protected against SCAs can be easily broken. Hence, if a cryptographic algorithm is used in embedded systems then it should be protected against SCAs.

Many SCAs against post-quantum cryptography have been proposed. For example, timing attacks and PA against McEliece encryption based on Goppa codes [EGHP09, HMP10, MSSS11] and on MDPC codes on a low-cost microcontroller [vMG14a, vMG14b, CEvMS15] have been proposed. Timing attacks, PA, and fault attacks against NTRUEncrypt [LSCH10, SW07, ZWW13, KY11] based on lattices also have been proposed. Recently, at ACM CCS 2017, Pessl *et al.* [PBY17] and Espitau *et al.* [EFGT17] presented practical

SCAs on lattice-based signature schemes, BLISS and BLISS-B variants. Furthermore, the results of SCAs on hash-based signature schemes, a fault attack on XMSS, and differential power analysis (DPA) of XMSS and SPHINCS, have been proposed [CMP18, KGB^{+}18]. There are a few results on implementation security of MQ-schemes. One is SCAs against SFLASH [SGB01, HTS11], which recovers a random seed Δ used for the hash function SHA-1, not the secret key (S, T). Another result comprises fault attacks on MQ-schemes [HTS11, YL17]. Hashimoto *et al.* [HTS11] presented general fault attacks on MQ-PKC including Big Field type, such as Matsumoto-Imai, HFEv-, and SFLASH, as well as Single Field type, such as UOV, Rainbow, STS, and TTM/TTS. Yi and Li [YL17] proposed a fault attack with DPA on enTTS which is a special case of Rainbow. No investigation has been conducted on the security of the MQ-schemes using only PA. In this paper, we provide the first results on the security of MQ-signature schemes using correlation power analysis (CPA) and algebraic key recovery attacks (KRAs).

Our Contributions. Our main contributions are as follows:

- **CPA on Rainbow Implementation with Equivalent Keys in the form of Fig. 1.** We recover a full secret key of Rainbow by performing CPA on its implementation with equivalent keys in the form of Fig. 1 to reduce the secret key size as presented in [CHT12]. The first source of side-channel leakage is a matrix-vector product obtained by a secret affine map S in signing. After recovering S, we recover the other secret affine map T using the special structure of the equivalent keys, This leads to the second source of the side-channel leakage, although we cannot know intermediate values via the central map $\mathcal{F}$. We demonstrate how this leakage can be practically exploited on an 8-bit AVR microcontroller using CPA.
- **Hybrid Attack on Rainbow Implementation with Random Affine Maps.** We extend our attack to Rainbow implementation with random affine maps instead of the equivalent keys in the form of Fig. 1. In this case, after recovering S via CPA, we recover T by mounting algebraic KRAs using good keys.
- **CPA on UOV Implementation with the Equivalent Key in the form of Fig. 2.** Our attack can be applied to the single layer MQ-signature, UOV, if it is implemented with the equivalent key in the form of Fig. 2 as presented in [CHT12] because the structure of equivalent keys is similar to those of Rainbow.
- **CPA on Other MQ-signature Schemes.** This is the first result on the security of MQ-signature schemes, Rainbow and UOV, against the non-invasive attacks. Our CPA can be applied to UOV-like single layer scheme LUOV, as the equivalent key T in the form of Fig. 2 was used in its design and implementation used. Our hybrid attack can also be applied to Rainbow-like multi-layered schemes, Rainbow, and HiMQ-3.

Organization. The rest of the paper is organized as follows. In Section 2, we describe UOV and Rainbow signature schemes, and CPA. In Section 3, we present a full secret key recovery on Rainbow implementation with the equivalent keys in the form of Fig. 1 via CPA. We then describe how this leakage in Rainbow can be practically exploited on an 8-bit AVR microcontroller using CPA. In Section 4, we demonstrate a hybrid attack on Rainbow implementation with random affine maps combining CPA with algebraic KRAs. CPA on other MQ-signature schemes is discussed in Section 5. We also discuss possible countermeasures to protect our proposed attacks in Section 6. Finally, concluding remarks are given in Section 7.

2 Preliminaries

Here, we first describe two MQ-signature schemes, UOV [KPG99] and its layered version Rainbow [DS05]. We then introduce the concept of equivalent keys of MQ-schemes and CPA.

2.1 UOV

Let $\mathbb{F}_q$ be a finite field with q elements. We define a system of multivariate quadratic equations $\mathcal{P} = (\mathcal{P}^{(1)}, \cdots, \mathcal{P}^{(m)})$ with m equations in n variables as

$$\mathcal{P}^{(k)}(x_1, \ldots, x_n) = \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(k)} x_i x_j + \sum_{i=1}^{n} p_i^{(k)} x_i + p_0^{(k)},$$

for $k = 1, \ldots, m$ and $p_{ij}^{(k)}$, $p_i^{(k)}$, and $p_0^{(k)}$ are randomly chosen in $\mathbb{F}_q$. It is very important to choose a system $\mathcal{F} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ of m quadratic equations in n variables, called a central map, which must be easily inverted in the MQ-signature schemes.

We set $\mathcal{V} = \{1, \ldots, v\}$, $|\mathcal{V}| = v$ and $\mathcal{O} = \{v+1, \ldots, n\}$, $|\mathcal{O}| = o$. We call $x_1, \ldots, x_v$ as Vinegar variables and $x_{v+1}, \ldots, x_n$ as Oil variables. A system $\mathcal{F} = (\mathcal{F}^{(1)}, \ldots, \mathcal{F}^{(m)})$ of multivariate quadratic equations in n variables $x_1, \ldots, x_n$ is defined by

$$\mathcal{F}^{(k)}(\mathbf{x}) = \sum_{i\in O, j\in V} \alpha_{ij}^{(k)} x_i x_j + \sum_{i,j\in V, i\le j} \beta_{ij}^{(k)} x_i x_j + \sum_{i\in V\cup O} \gamma_i^{(k)} x_i + \eta^{(k)},$$

where $\mathbf{x} = (x_1, \ldots, x_n)$, $k = 1, \ldots, o$, $n = v + o$, and $m = o$. Note that Oil and Vinegar variables are not fully mixed, just like oil and vinegar in salad dressing. The central map $\mathcal{F}$ can be easily inverted: first, choose a vector $\mathbf{v} = (x_1, \ldots, x_v) \in \mathbb{F}_q^v$ of Vinegar values at random and plug $\mathbf{v}$ into $\mathcal{F}^{(k)}$ for $k = 1, \ldots, o$. Then, one gets a linear system of o equations with o variables $x_{v+1}, \ldots, x_n$ and solves the linear system by using the Gaussian elimination. To hide the structure of the central map $\mathcal{F}$ in the public key, it is composed with an invertible affine map $T : \mathbb{F}_q^n \to \mathbb{F}_q^n$. A public key is given by $\mathcal{P} = \mathcal{F} \circ T$ and a secret key is $(\mathcal{F}, T)$ which allows to invert the public key. Let $\mathcal{H} : \{0,1\}^* \to \mathbb{F}_q^m$ be a collision-resistant hash function.

■ **UOV**

- **KeyGen**(1^λ)
  1. For a security parameter λ, a public key is $PK = \mathcal{P} = \mathcal{F} \circ T$ and a secret key is $SK = (\mathcal{F}, T)$.
- **Sign**$(SK, \mathtt{m})$
  1. Given a message $\mathtt{m}$, compute $\mathbf{h} = \mathcal{H}(\mathtt{m}) \in \mathbb{F}_q^m$.
  2. To compute $\alpha = \mathcal{F}^{-1}(\mathbf{h})$, i.e., $\mathcal{F}(\alpha) = \mathbf{h}$, choose Vinegar values $(s_1, \ldots, s_v) \in \mathbb{F}_q^v$ at random and substitute $(s_1, \ldots, s_v)$ into o polynomials $\mathcal{F}^{(k)}$ $(1 \le k \le o)$ by obtaining a system of o linear equations with o variables. Find a solution $(s_{v+1}, \ldots, s_{v+o})$ of the linear system using Gaussian elimination. If the linear system does not have a solution, choose another vector of Vinegar values and try again, otherwise, α is $(s_1, \ldots, s_n)$.
  3. Compute $\sigma = \widetilde{T}(\alpha)$, where $\widetilde{T} = T^{-1}$. Output σ is a signature of $\mathtt{m}$.
- **Verify**$(PK, \mathtt{m}, \sigma)$
  1. Given $(\sigma, \mathtt{m})$, check $\mathcal{P}(\sigma) = \mathcal{H}(\mathtt{m})$. If it holds, accept the signature, otherwise reject it.

2.2 Rainbow

Ding and Schmidt [DS05] proposed a layered MQ-signature scheme, Rainbow, based on UOV to improve efficiency and reduce the key sizes. It has been submitted to NIST for Post-Quantum Cryptography Standards. Let $v_1, \ldots, v_{u+1}$ $(u \geq 1)$ be integers such that $0 < v_1 < v_2 < \ldots < v_u < v_{u+1} = n$. Define sets of integers $V_i = \{1, \ldots, v_i\}$ for $i = 1, \ldots, u$ and set $o_i = v_{i+1} - v_i$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ $(i = 1, \ldots, u)$. Then $|V_i| = v_i$ and $|O_i| = o_i$. For $k = v_1 + 1, \ldots, n$, we define multivariate quadratic polynomials in n variables $x_1, \ldots, x_n$ as follows:

$$\mathcal{F}^{(k)}(\mathbf{x}) = \sum_{i \in O_l, j \in V_l} \alpha_{ij}^{(k)} x_i x_j + \sum_{i,j \in V_l, i \leq j} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V_l \cup O_l} \gamma_i^{(k)} x_i + \eta^{(k)},$$

where $\mathbf{x} = (x_1, \ldots, x_n)$ and l is the only integer such that $k \in O_l$. These are Oil and Vinegar polynomials with x_i for $i \in V_l$ being Vinegar variables and x_j for $j \in O_l$ being Oil variables. The central map $\mathcal{F}(\mathbf{x}) = (\mathcal{F}^{(v_1+1)}(\mathbf{x}), \ldots, \mathcal{F}^{(n)}(\mathbf{x}))$ can be inverted using the Oil-Vinegar method as in UOV. Then, two invertible affine maps $S : \mathbb{F}_q^m \to \mathbb{F}_q^m$ and $T : \mathbb{F}_q^n \to \mathbb{F}_q^n$ are chosen to hide the special structure of the central map $\mathcal{F}$ in the public key, where T mixes the variables and S mixes the polynomials. A public key is the composition of three maps, $\mathcal{P} = S \circ \mathcal{F} \circ T$, and a secret key is $(S, \mathcal{F}, T)$. In general, Rainbow is denoted by Rainbow$(\mathbb{F}_q, v_1, o_1, \ldots, o_u)$ with $m = \sum_{i=1}^{u} o_i$ and $n = v_1 + m$. For $u = 1$, it is the original UOV scheme. We describe Rainbow with two layers, Rainbow $(\mathbb{F}_q, v_1, o_1, o_2)$ with $m = o_1 + o_2$ and $n = v_1 + m$.

■ **Rainbow**

– **KeyGen**(1^λ)

1. For a security parameter λ, a public key is $PK = \mathcal{P} = S \circ \mathcal{F} \circ T$ and a secret key is $SK = (S, \mathcal{F}, T)$.

– **Sign**$(SK, \mathtt{m})$

1. Given a message $\mathtt{m}$, compute $\mathbf{h} = \mathcal{H}(\mathtt{m}) \in \mathbb{F}_q^m$.
2. Compute $\alpha = \widetilde{S}(\mathbf{h})$, where $\widetilde{S} = S^{-1}$.
3. Compute $\beta = \mathcal{F}^{-1}(\alpha)$, i.e. $\mathcal{F}(\beta) = \alpha$.
   (a) First, choose $(s_1, \ldots, s_{v_1}) \in \mathbb{F}_q^{v_1}$ at random, substitute $(s_1, \ldots, s_{v_1})$ into o_1 polynomials $\mathcal{F}^{(k)}$ $(v_1 + 1 \leq k \leq v_2 = v_1 + o_1)$ and obtain $(s_{v_1+1}, \ldots, s_{v_2})$ by solving a system of o_1 linear equations with o_1 variables $x_{v_1+1}, \ldots, x_{v_2}$ via Gaussian elimination.
   (b) Next, substitute $(s_1, \ldots, s_{v_2})$ into o_2 polynomials $\mathcal{F}^{(k)}$ $(v_2 + 1 \leq k \leq n)$ and obtain $(s_{v_2+1}, \ldots, s_n)$ by solving the linear system of o_2 equations with o_2 variables $x_{v_2+1}, \ldots, x_n$ through Gaussian elimination. Then, $\beta = (s_1, \ldots, s_n)$. If one of the linear systems does not have a solution, choose another vector of Vinegar values and try again.
   (c) Compute $\sigma = \widetilde{T}(\beta)$, where $\widetilde{T} = T^{-1}$. Output σ is a signature of $\mathtt{m}$.

– **Verify**$(PK, \mathtt{m}, \sigma)$

1. Given $(\sigma, \mathtt{m})$, check $\mathcal{P}(\sigma) = \mathcal{H}(\mathtt{m})$. If it holds, accept the signature, otherwise reject it.

2.3 Equivalent Keys in UOV and Rainbow

The existence of numerous different secret keys corresponding to a given public key is a special feature of MQ-schemes [DYC+08, WP05]. The concept of equivalent keys is defined by Definition 1. Let $\mathrm{GL}_m(\mathbb{F}_q)$ be a general linear group of degree m over $\mathbb{F}_q$.

Definition 1. [Equivalent Key] Let $S, S' \in \mathrm{GL}_m(\mathbb{F}_q)$, $T, T' \in \mathrm{GL}_n(\mathbb{F}_q)$, and $\mathcal{F}, \mathcal{F}' \in \mathbb{F}_q[x_1, ..., x_n]^m$. We say that $(\mathcal{F}, S, T)$ is *equivalent* to $(\mathcal{F}', S', T')$ if and only if $S \circ \mathcal{F} \circ T = S' \circ \mathcal{F}' \circ T'$ and $\mathcal{F}|_I = \mathcal{F}'|_I$, i.e., $\mathcal{F}'$ and $\mathcal{F}$ have the same structure when restricted to a fixed index set $I = \{I^{(1)}, \cdots I^{(m)}\}$.

Assume that $\mathcal{P}$ is given by $\mathcal{P} = S \circ \mathcal{F} \circ T$. If $\mathcal{P} = S' \circ \mathcal{F}' \circ T'$ and $\mathcal{F}'$ preserves all zero coefficients of $\mathcal{F}$ then we call (S', T') equivalent keys of $\mathcal{P}$. The concept of equivalent keys plays an important role in breaking MQ-schemes. If an adversary can find any equivalent key then he can forge signatures on any messages for the public key $\mathcal{P}$. KRAs exploit the special structure of the central map, i.e., zero entries at certain known places, to obtain equations with variables in S and T. If one can find simpler equivalent keys, S' and T', then one has to solve a large structured system of multivariate quadratic equations to recover S' and T' by reducing the number of variables. If one can find two invertible linear maps $\Sigma \in \mathrm{GL}_m(\mathbb{F}_q)$ and $\Omega \in \mathrm{GL}_n(\mathbb{F}_q)$ such that

$$\mathcal{P} = S \circ \Sigma^{-1} \circ (\Sigma \circ \mathcal{F} \circ \Omega) \circ \Omega^{-1} \circ T,$$

and $\mathcal{F}'$ and $\mathcal{F}$ have the same structure, then $(S', \mathcal{F}', T')$ is an equivalent key, where $\mathcal{F}'(= \Sigma \circ \mathcal{F} \circ \Omega)$, $S' = S \circ \Sigma^{-1}$, and $T' = \Omega^{-1} \circ T$. Note that, we set the preserving index set as all quadratic terms with zero coefficients in the Oil $\times$ Oil part. Fig. 1 presents the forms of the equivalent keys of Rainbow, where gray parts denote arbitrary entries and white parts denote zero entries and there are ones at the diagonal. For UOV, if one can find $\Omega \in \mathrm{GL}_n(\mathbb{F}_q)$ such that

$$\mathcal{P} = \mathcal{F} \circ T = (\mathcal{F} \circ \Omega) \circ (\Omega^{-1} \circ T),$$

and $\mathcal{F}'$ and $\mathcal{F}$ have the same structure then $(\mathcal{F}', T')$ is an equivalent key, where $\mathcal{F}' = \mathcal{F} \circ \Omega$ and $T' = \Omega^{-1} \circ T$. The form of equivalent keys of UOV is given in Fig. 2.

We will use these special structures of equivalent keys for CPA on UOV and Rainbow, if UOV and Rainbow are implemented with the equivalent keys of the forms in Fig. 1 and Fig. 2, respectively, as in [CHT12].

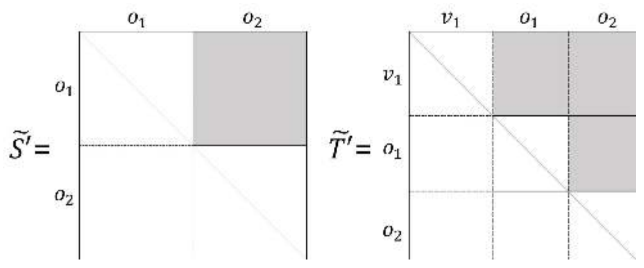


Figure 1: [CHT12], Equivalent keys of Rainbow

2.4 Correlation Power Analysis

PA exploits power consumption, which is measured when a cryptosystem operates on an electronic device. The two main classes of PA are simple power analysis (SPA) and DPA.

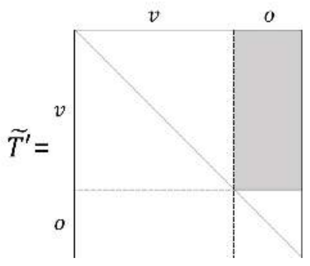


Figure 2: [CHT12], Equivalent key of UOV

Here, we introduce the properties of DPA. Typically, only knowledge of the cryptographic algorithm is sufficient. DPA is based on a divide and conquer scheme. Generally, this approach involves forming a hypothesis value and then comparing the hypothesis against measured power traces. An attacker repeats this process for all sub-key candidates and determines the value of each sub-key to recover the full key.

Power consumption is typically modeled by estimating the number of $1'$s in a register using a Hamming weight or Hamming distance power model. If Pearson's correlation coefficient is used in DPA, it is referred to as CPA. DPA is summarized as follows [MOP07].

1. **Choosing an intermediate result of the executed cryptographic algorithm.**
   This intermediate result is computed by a known non-constant data value d_i, a small part of the secret value k_j, and a target function $f\left(d_i, k_j\right)$. In most attack scenarios, d_i is either plaintext or ciphertext.

2. **Measuring the power consumption.**
   The second step is to measure the power consumption of the target device while it encrypts (or decrypts). First, the attacker finds the position of the target function chosen in Step 1, and then, measures the power consumption at that position.

3. **Calculating hypothetical intermediate values.**
   This step calculates hypothetical intermediate values for every possible k_j.

4. **Mapping hypothetical intermediate values to hypothetical power consumption values.**
   Typically, hypothetical power consumption values are modeled using the Hamming weight or Hamming distance power model.

5. **Statistically comparing the hypothetical power values with the measured power traces.**
   This step compares the hypothetical power consumption values of each key hypothesis with the recorded traces at every position.

The correlation coefficient is an excellent choice when DPA is performed as it is the most common way to determine linear relationships between data. The correlation coefficient used in **Step 4** is expressed as follows:

$$\rho\left(X, Y\right) = \frac{Cov\left(X, Y\right)}{\sqrt{Var\left(X\right) \cdot Var\left(Y\right)}},$$

where $Cov\left(X, Y\right)$ and $Var(X)$ denote the covariance between random variables X and Y and the variance of random variable X, respectively.

3 CPA on Rainbow Implementation Using the Equivalent Keys in the Form of Fig. 1

3.1 General Leakage Model

Here, we present a general leakage model against CPA on Rainbow implementation that operates on embedded devices.[1] As seen in §2.1 and §2.2, both UOV and Rainbow perform matrix-vector product operations for their signature generations. In Rainbow, the matrix-vector product is computed at the second step of signature generation to compute α with $\widetilde{S}$ and the hash value of message m. The matrix-vector product is also calculated at the final step to compute a signature in both UOV($\sigma = \widetilde{T}(\alpha)$) and Rainbow($\sigma = \widetilde{T}(\beta)$).

To recover the secret affine map, we target the location where the matrix-vector product operates. It is easy to reveal the secret affine map $\widetilde{S}$ using CPA as we can control (or know) the vector multiplied by $\widetilde{S}$. Unfortunately, a technical hurdle must be overcome to recover the other secret affine map $\widetilde{T}$, as we cannot find intermediate values, which are multiplied by $\widetilde{T}$. However, if Rainbow and UOV are implemented with their equivalent keys in the forms of Fig. 1 and Fig. 2, respectively, for efficiency, we can also retrieve $\widetilde{T}$ using CPA, although we cannot find all intermediate values.

Let a matrix-vector product with matrix $A \in \mathbb{F}_q^{n\times n}$ and $\mathbf{x} \in \mathbb{F}_q^n$ be $\mathbf{x}' = A\mathbf{x}^T = (x'_1, x'_2, \dots, x'_n) \in \mathbb{F}_q^n$. Each element of $\mathbf{x}'$ is calculated as follows:

$$\mathbf{x}' = \begin{bmatrix} x'_1 \\ x'_2 \\ \vdots \\ x'_n \end{bmatrix} = A\mathbf{x}^T = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \dots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix},$$

$$x'_i = \sum_{j=1}^{n} a_{ij} \cdot x_j = a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \cdots + a_{in} \cdot x_n, \quad 1 \le i \le n. \tag{1}$$

First, if we can control all elements of the vector $\mathbf{x}$ to recover affine map $\widetilde{S}$, then we use the intermediate results in Table 1 to obtain all elements of the i^{th} row of matrix A.

Table 1: Intermediate results to recover $\widetilde{S}$

Target element	Intermediate result
a_{i1}	$\textbf{Guess} \cdot x_1$
a_{i2}	$a_{i1} \cdot x_1 + \textbf{Guess} \cdot x_2$
$\vdots$	$\vdots$
a_{in}	$\sum_{k=1}^{n-1} a_{ik} \cdot x_k + \textbf{Guess} \cdot x_n$

In Table 1, **Guess** represents a hypothetical key. The intermediate results can always be used regardless of using the equivalent keys in the form of Fig. 1.

Second, if the equivalent keys in the form of Fig. 1 are used at the final step of signature generation, we can find some information regarding the intermediate values. Here, let the matrix A be an equivalent key in the form of Fig. 1 which consists of two submatrices A_1 and A_2 as shown Fig. 3. Assume that the equivalent key A in the form of Fig. 1 is used in Rainbow implementation.

As can be seen in Fig. 3, v_1 vinegar variables of $\mathbf{x}'$ are affected by A_1 and A_2, and the first o_1 oil variables of $\mathbf{x}'$ are only affected by A_2. The last o_2 oil variables, which are used

[1]As implementations in the SIMD environment increases, the study of SCA characteristics for this environment is needed. We will study SCA characteristic in the SIMD environment later.

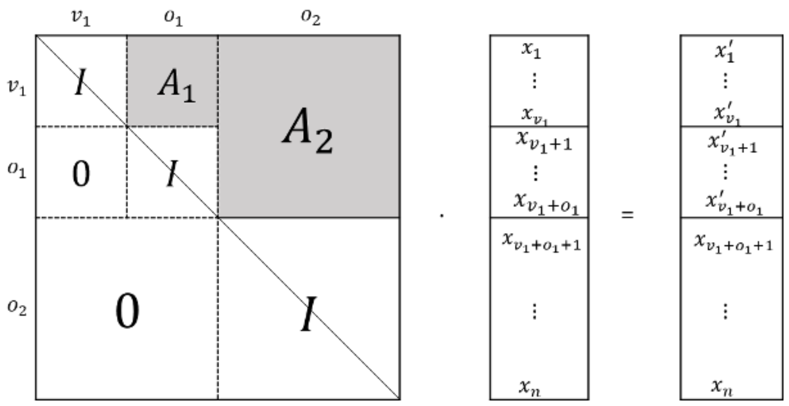


Figure 3: Matrix-vector product using the equivalent key in the form of Fig. 1 in Rainbow with two layers

in the second layer of Rainbow, do not change. Therefore, it is possible to recover A_2 because the last o_2 elements of $\mathbf{x}$ that were computed using the submatrix A_2 are identical to the last o_2 elements of $\mathbf{x}'$. In other words,

$$x'_{v_1+o_1+1} = x_{v_1+o_1+1},\ \ x'_{v_1+o_1+2} = x_{v_1+o_1+2},\ \ \dots\ \ ,\ \ x'_n = x_n.$$

Thus, we can reveal all elements of A_2, i.e., the elements from the $(v_1+o_1+1)^{\text{th}}$ to the n^{th} column. Here, we use the following intermediate result to recover A_2:

$$\mathbf{Guess} \cdot x_k,\ \ v_1+o_1+1 \le k \le n.$$

After recovering A_2, we compute x_{v_1+1} to $x_{v_1+o_1}$ using the following equation:

$$x'_{v_1+t} = 1 \cdot x_{v_1+t} + \sum_{k=1}^{o_2} A^{(2)}_{v_1+t,k} \cdot x_{v_1+o_1+k},\ \ 1 \le t \le o_1,$$

where $A^{(2)}_{i,j}$ $(1 \le i \le v_1+o_1,\ 1 \le j \le o_2)$ represents the $(i,j)^{\text{th}}$ element of the submatrix A_2. As $x_{v_1+1}, x_{v_1+2}, \dots, x_{v_1+o_1}$ is multiplied with A_1, we can reveal all elements of A_1 using the following intermediate result:

$$\mathbf{Guess} \cdot x_k,\ \ v_1+1 \le k \le v_1+o_1.$$

Algorithm 1 Matrix-vector product

Input: matrix $A \in \mathbb{F}_q^{n\times n}$, vector $\mathbf{x} \in \mathbb{F}_q^n$
Output: vector $\mathbf{x}' \left(= A\mathbf{x}^T\right) \in \mathbb{F}_q^n$

```
1: for $i = 1$ to $n$ do
2:     $x'_i = 0$
3: end for
4: for $i = 1$ to $n$ do
5:     for $j = 1$ to $n$ do
6:         $x'_j = x'_j + a_{ji} \cdot x_i$        // +: field addition, ·: field multiplication
7:     end for
8: end for
9: return $\mathbf{x}'$
```

■ Experimental Setup

Algorithm 1 is the most commonly used efficient matrix-vector product method because it can reduce the number of load operations. We analyze the C code of Rainbow at an 80-bit level, which also uses the matrix-vector product in Algorithm 1, over $\mathbb{F}_q$ with $q = 2^8$ [COD]. Clearly, the addition operation of Eq. (1) can be replaced by the exclusive OR operation. To explain simply, we assume that the matrix-vector product is implemented as shown in Algorithm 1 with $m = 6$, $n = 8$, and two layers in the Rainbow signature scheme with $o_1 = 2$ and $o_2 = 4$. Note that, even though the parameters do not match the currently accepted security parameters, if CPA succeeds against this parameter set, the result can be easily extended to larger parameters. Moreover, we can retrieve secret key even if the size of o_2 is small.

We port the matrix-vector product code on the `ChipWhisperer-Lite` evaluation platform [New]. `ChipWhisperer-Lite` was developed to support embedded hardware security research. It is comprised of two main parts, a multi-purpose PA capture instrument and a target board. The target board is an Atmel XMEGA128 programmable chip with a fixed clock frequency of 7.37 MHz. The signal is amplified up to 55dB gain, and the power traces are sampled at a rate of 96 MS/s. To obtain power consumption traces of $\widetilde{S}$, we first generate random hashed messages $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \ldots, \mathbf{h}^{(N)} \in \mathbb{F}_{2^8}^8$, and then perform the matrix-vector product operations. Here, $\mathbf{h}^{(i)}$ represents the hash value of ith message which is used in the ith signature generation. To recover $\widetilde{T}$, we use outputs of signature generation, i.e., $\sigma^{(1)}, \sigma^{(2)}, \ldots, \sigma^{(N)}$. For convenience, we drop the exponent of $\mathbf{h}$ and σ in the remaining paper.

3.2 Case Study: Recovering S and T

Now, we present a case study on the recovery of $\widetilde{S}$ and $\widetilde{T}$ in Rainbow. The second step of the Rainbow signature algorithm is to multiply $\widetilde{S} \in \mathbb{F}_{2^8}^{8\times 8}$ by the hashed message $\mathbf{h} \in \mathbb{F}_{2^8}^8$.

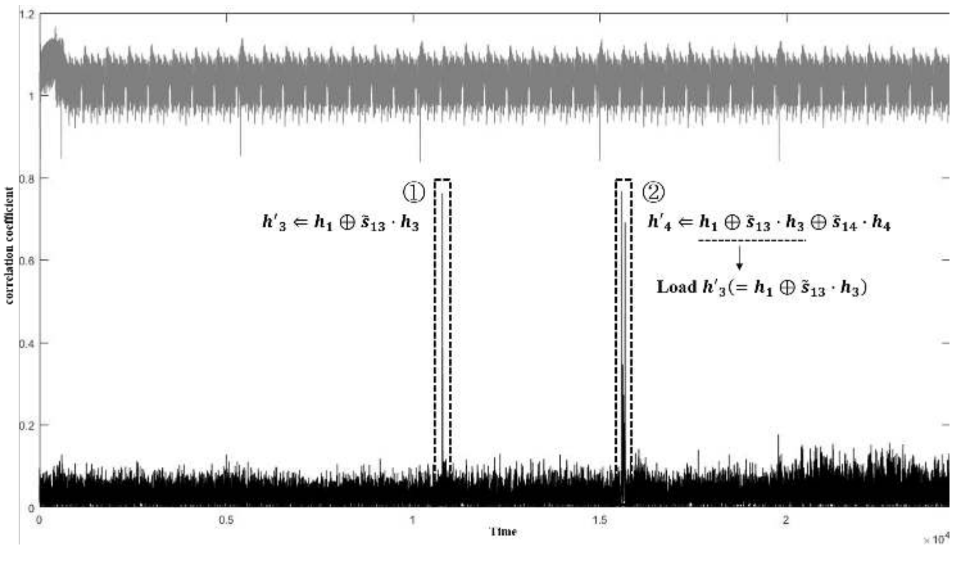


Figure 4: Possible attack positions to reveal $\widetilde{s}_{13}$

$$\widetilde{S}\mathbf{h} = \begin{bmatrix} 01 & 00 & \widetilde{s}_{13} & \widetilde{s}_{14} & \widetilde{s}_{15} & \widetilde{s}_{16} & \widetilde{s}_{17} & \widetilde{s}_{18} \\ 00 & 01 & \widetilde{s}_{23} & \widetilde{s}_{24} & \widetilde{s}_{25} & \widetilde{s}_{26} & \widetilde{s}_{27} & \widetilde{s}_{28} \\ 00 & 00 & 01 & 00 & \widetilde{s}_{35} & \widetilde{s}_{36} & \widetilde{s}_{37} & \widetilde{s}_{38} \\ 00 & 00 & 00 & 01 & \widetilde{s}_{45} & \widetilde{s}_{46} & \widetilde{s}_{47} & \widetilde{s}_{48} \\ 00 & 00 & 00 & 00 & 01 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 01 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 01 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \end{bmatrix}$$

When the hashed message $\mathbf{h}$ and matrix $\widetilde{S}$ are multiplied, $\mathbf{h}$ contains known values, and each element of $\widetilde{S}$ is a secret value that the attacker wants to find. If we perform CPA using the intermediate result discussed in §3.1, it is easy to find all elements of $\widetilde{S}$. However, the obtained traces must be split into partial matrix-vector product traces owing to Algorithm 1.

There are two attack positions to find for each $\widetilde{s}_{ij}$. For example, if we target $\widetilde{s}_{13}$, then the first position is where the value of $h_1 \oplus (\widetilde{s}_{13} \cdot h_3)$ is calculated and stored. The second position is where the value of $h_1 \oplus (\widetilde{s}_{13} \cdot h_3)$ is loaded to calculate $h_1 \oplus (\widetilde{s}_{13} \cdot h_3) \oplus (\widetilde{s}_{14} \cdot h_4)$. Fig. 4 shows the correlation coefficient for the intermediate result $h_1 \oplus (\widetilde{s}_{13} \cdot h_3)$ with the correct value. As can be seen in Fig. 4, there are two positions with high peaks.

When we attack the first position, loading of the message h_1 to calculate $h_1 \oplus (\widetilde{s}_{13} \cdot h_3)$ would encumber us. CPA results for $\widetilde{s}_{13}$ at the first attack position are represented in Fig. 9a and 9b (Appendix A). The correlation coefficient for h_1, i.e., the correlation coefficient of $\widetilde{s}_{13} = 0$, is higher than that for a right key. We have compared the locations where h_1 is loaded and $h_1 \oplus (\widetilde{s}_{13} \cdot h_3)$ is calculated. Peaks occurred in almost similar positions, however, one point difference was observed between loading h_1 and calculating $h_1 \oplus (\widetilde{s}_{13} \cdot h_3)$. The right key should be distinguished by substituting the key candidates in the intermediate result of $\widetilde{s}_{14}$. However, we use the second position for simplicity as it is unaffected by the loading of h_1.

To recover the other secret map $\widetilde{T}$, we consider the computation of the product $\widetilde{T}\beta$ for $\beta \in \mathbb{F}_{2^8}^8$, where β is the intermediate value obtained from $\widetilde{\mathcal{F}}(\alpha)$ for $\alpha = \widetilde{S}(\mathbf{h})$. Here, let $\widetilde{T}$ be of the same form as $\widetilde{S}$.[2]

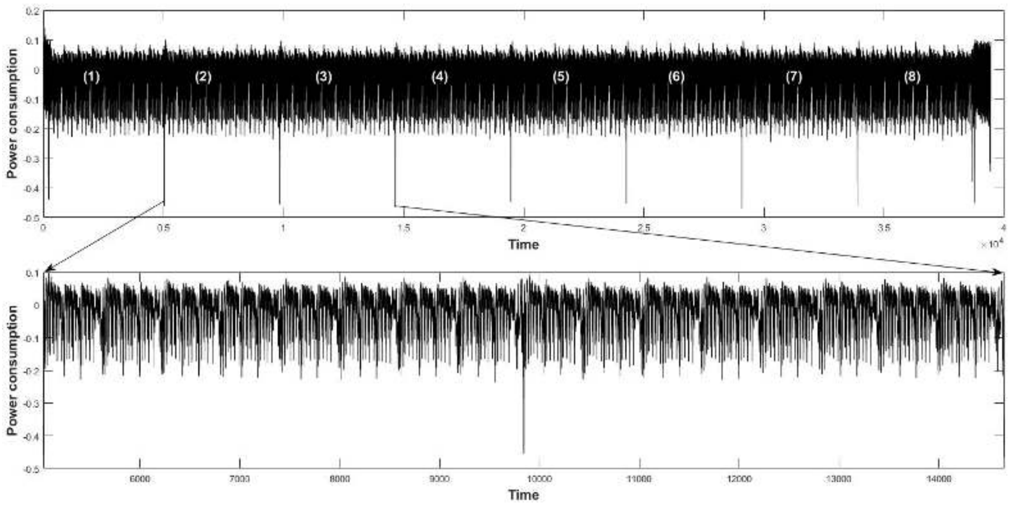


Figure 5: Top: Power consumption trace of the matrix-vector product with 8×8 matrix and vector; Bottom: Expansion of the top trace

[2]We express the elements of $\widetilde{T}$ as $\widetilde{t}_{ij}$.

$$\widetilde{T}\beta = \begin{bmatrix} 01 & 00 & \widetilde{t}_{13} & \widetilde{t}_{14} & \widetilde{t}_{15} & \widetilde{t}_{16} & \widetilde{t}_{17} & \widetilde{t}_{18} \\ 00 & 01 & \widetilde{t}_{23} & \widetilde{t}_{24} & \widetilde{t}_{25} & \widetilde{t}_{26} & \widetilde{t}_{27} & \widetilde{t}_{28} \\ 00 & 00 & 01 & 00 & \widetilde{t}_{35} & \widetilde{t}_{36} & \widetilde{t}_{37} & \widetilde{t}_{38} \\ 00 & 00 & 00 & 01 & \widetilde{t}_{45} & \widetilde{t}_{46} & \widetilde{t}_{47} & \widetilde{t}_{48} \\ 00 & 00 & 00 & 00 & 01 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 01 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 01 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \\ \beta_6 \\ \beta_7 \\ \beta_8 \end{bmatrix} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \\ \sigma_7 \\ \sigma_8 \end{bmatrix}$$

Note that $\sigma_5 = \beta_5$, $\sigma_6 = \beta_6$, $\sigma_7 = \beta_7$, and $\sigma_8 = \beta_8$. Thus, we can reveal the elements of $\widetilde{T}$ from columns 5 to 8. In fact, we still do not have information regarding β_3 and β_4, which will encumber us when we perform CPA to find $\widetilde{t}_{3j}$ and $\widetilde{t}_{4j}$. However, we can reveal $\widetilde{t}_{4j}$(and $\widetilde{t}_{3j}$) because there are positions where $\widetilde{t}_{ij} \cdot \beta_j$ is computed. After these positions are found, we perform CPA with the intermediate result **Guess** $\cdot\, \beta_j$. If we find the elements of $\widetilde{T}$ from columns 5 to 8, we can also calculate β_3 and β_4 as follows:

$$\beta_3 = \sigma_3 \oplus \widetilde{t}_{35} \cdot \beta_5 \oplus \widetilde{t}_{36} \cdot \beta_6 \oplus \widetilde{t}_{37} \cdot \beta_7 \oplus \widetilde{t}_{38} \cdot \beta_8,$$
$$\beta_4 = \sigma_4 \oplus \widetilde{t}_{45} \cdot \beta_5 \oplus \widetilde{t}_{46} \cdot \beta_6 \oplus \widetilde{t}_{47} \cdot \beta_7 \oplus \widetilde{t}_{48} \cdot \beta_8.$$

Simialrly, we can recover the remaining elements of $\widetilde{T}$ from $\widetilde{t}_{j3}$ to $\widetilde{t}_{j4}$ for all $1 \leq j \leq 2$.

3.3 Experimental Results

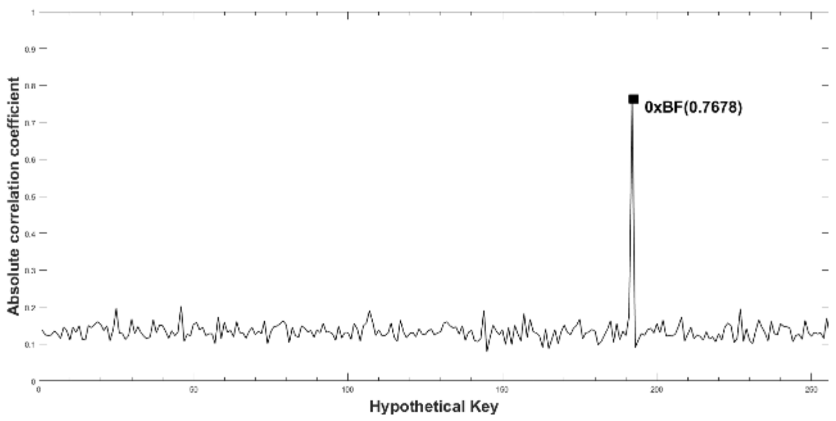
The top of Fig. 5 shows a power consumption trace during the multiplication of the 8×8 matrix $\widetilde{S}$ and vector $\mathbf{h}$. As can be seen, eight operations are performed, i.e., the first iteration after initialization in Algorithm 1 (steps 4?8), and, if we expand part of the top of Fig. 5, we will find that eight similar operations are repeated. The bottom of Fig. 5 enlarges positions 2 and 3 in the top of Fig. 5, i.e., the second loop of Algorithm 1 (steps 5?7).

Since the same message is used consecutively, similar to Algorithm 1, we must find an appropriate target position that effectively reveals all elements of the secret matrices prior to performing CPA. We then carry out CPA with the collected 500 traces. Fig. 9 (Appendix A) and Fig. 6 show the results of CPA for $\widetilde{s}_{13}$. Fig. 9a (Appendix A) shows the maximum correlation coefficients of all hypothetical keys and Fig. 9b (Appendix A) shows the maximum correlation coefficients according to an increased number of traces at the first position. We experiment with traces that increase by 10. Here, even if the number of traces is increased, the appropriate key cannot be found. Figs. 6a and 6b show the maximum correlation coefficients of all hypothetical keys and the maximum correlation coefficients according to an increased number of traces at the second position, respectively. As can be seen in Fig. 6b, we can find the appropriate key using only 30 traces.

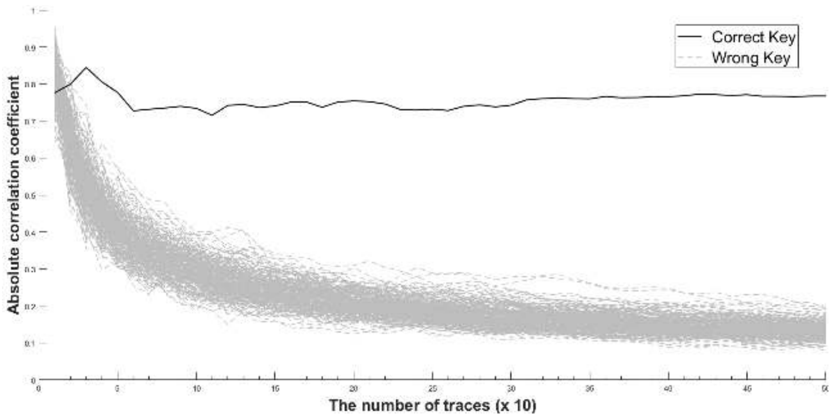
As mentioned previously, we use the signature values to reveal $\widetilde{T}$ and only target the place where $\widetilde{t}_{ij} \cdot \beta_j$ is computed because we have unknown values. Fig. 7a and Fig. 7b show the CPA results for $\widetilde{t}_{45}$ and $\widetilde{t}_{46}$, respectively.

4 Hybrid Attack on Rainbow Implementation with Random Affine Maps

In the previous section, we recovered the affine map T using CPA when Rainbow is implemented with its special equivalent keys of the forms shown in Fig. 1. Now, we show that full secret key recovery is possible via a hybrid attack when Rainbow is implemented using random affine maps instead of the equivalent keys in the form of Fig. 1. The hybrid



(a) Maximum correlation coefficients



(b) Maximum correlation coefficients according to increased number of traces

Figure 6: CPA results for $\widetilde{s}_{13}$ at second position

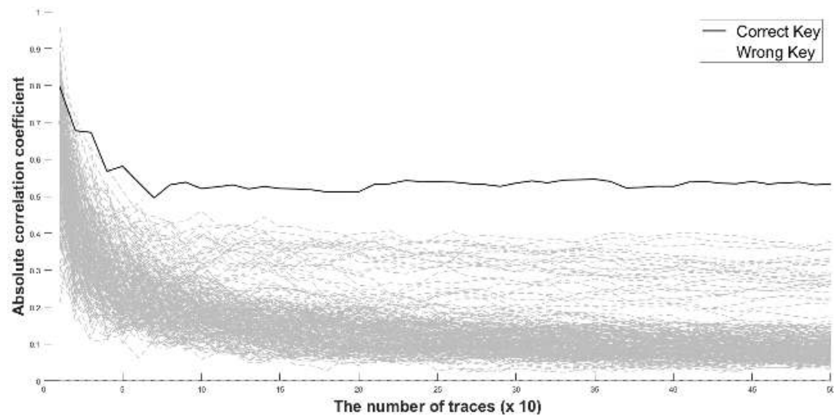
attack is a combination of CPA and algebraic KRAs: we first recover a random affine secret map S by performing CPA, and then, another affine secret map T is recovered by mounting algebraic KRAs.

4.1 Recovery of Random Affine Map S via CPA

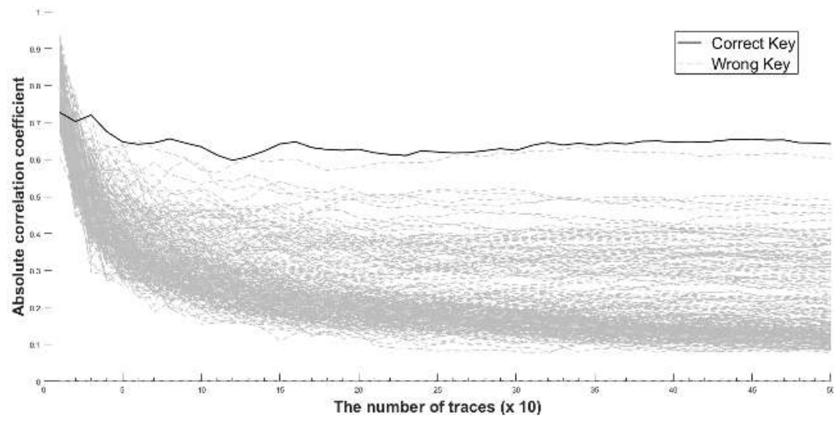
The recovery of the random affine map $\widetilde{S}$ on Rainbow with random affine maps using CPA is identical to that of Rainbow with the special form of equivalent secret keys. Fig. 10 (Appendix B) shows CPA results for the random secret key $\widetilde{S}$ with 500 traces when we target the position where the $\widetilde{s}_{11} \cdot h_1$ value is loaded. As seen in Fig. 10b (Appendix B), only 30 traces are required to reveal $\widetilde{s}_{11}$. Thus, the secret affine map S is recovered in this manner.

4.2 Recovery of Random Affine Map T via KRAs

Suppose that S is recovered from the public key $\mathcal{P} = S \circ \mathcal{F} \circ T$. Then we can easily recover T via the algebraic KRAs. In fact, the role of S is to mix the polynomials in the first and the second layers since the polynomials have different structures in multi-layered MQ-signature schemes such as Rainbow. Thus, in layered MQ-signature schemes, the removal of S means the extraction of each layer that leads to the breaking of the scheme



(a) Maximum correlation coefficients according to increased number of traces for $\widetilde{t}_{45}$



(b) Maximum correlation coefficients according to increased number of traces for $\widetilde{t}_{46}$

Figure 7: CPA results for $\widetilde{t}_{45}$ and $\widetilde{t}_{46}$

through several attacks, including rank-based attacks, direct attacks, and KRAs. Here, we show that T can be retrieved by KRAs using good keys.

Because S and $\mathcal{P}$ are known, we can compute $S^{-1} \circ \mathcal{P}$. Hence, without loss of generality, we begin with the structure $\mathcal{P} = \mathcal{F} \circ T$, where T is a random invertible affine map. We denote $F^{(k)}$ $(1 \leq k \leq m)$ by symmetric matrices corresponding to the homogeneous quadratic part of the k-th component of the central map $\mathcal{F}$. We also denote $P^{(k)}$ $(1 \leq k \leq m)$ by symmetric matrices associated to the quadratic part of the k-th component of the public key $\mathcal{P}$. As $\mathcal{P} = \mathcal{F} \circ T$, $\mathcal{F} = \mathcal{P} \circ \widetilde{T}$, where $\widetilde{T} = T^{-1}$ and certain places with zero coefficients in $F^{(k)}$ are known, we obtain the following equality:

$$F^{(k)} = \widetilde{T}^T \cdot P^{(k)} \cdot \widetilde{T}, \quad \forall 1 \leq k \leq m.$$

The corresponding system of equations is:

$$f_{ij}^{(k)} = \sum_{y=1}^{n} \sum_{z=1}^{n} P_{yz}^{(k)} \widetilde{t}_{yi} \widetilde{t}_{zj}, \tag{2}$$

where $P_{yz}^{(k)}$ the coefficient of $x_y x_z$ in $P^{(k)}$, because we already know that $f_{ij}^{(k)} = 0$ for some

i, j, k by the construction of $\mathcal{F}$. For Rainbow($\mathbb{F}_q, v_1, o_1, o_2$), we obtain a system of

$$\frac{o_1 m(n + v_1 + 1) + o_2^2(o_2 + 1)}{2} - o_1^2 v_1$$

quadratic equations with n^2 variables. The complexity of solving such a system using HF5 is very large, where HF5 is an efficient Gröbner basis algorithm for solving the MQ-problem [BFP09].

To improve this complexity, we can find an equivalent key $(\mathcal{F}', T')$ such that $P = \mathcal{F}' \circ T'$, where $\mathcal{F}'^{(k)}_{ij} = 0$ if and only if $\mathcal{F}^{(k)}_{ij} = 0$. The equivalent key $\mathcal{F}'$ and $\widetilde{T'}$ have the form shown in Fig. 8a and Fig. 8b, respectively. Then, we obtain a system of

$$\frac{o_1 m(n + v_1 + 1) + o_2^2(o_2 + 1)}{2} - o_1^2 v_1$$

quadratic equations with $v_1 m + o_1 o_2$ variables. For the dotted part of Fig. 8, the equations determined from the part are linear because the first v_1 columns of $\widetilde{T'}$ do not have any variables of the form $\widetilde{t_{yi}}$. Thus, we obtain a linear system of $v_1 o_1 o_2$ equations with $o_2(v_1 + o_1)$ variables. From (2), these equations are of the form

$$f^{(k)}_{ij} = \sum_{z=1}^{v_1+o_1} P^{(k)}_{iz} \widetilde{t}_{zj} + P^{(k)}_{ij} \tag{3}$$

for $1 \le i \le v_1$ and $v_1 + o_1 + 1 \le j \le n$. Note that all variables of each equation in (3) are in a column of $\widetilde{T'}$. For each j with $v_1 + o_1 + 1 \le j \le n$, we obtain a smaller linear system of $v_1 o_1$ equations with $v_1 + o_1$ variables in the j-th column of $\widetilde{T'}$. Hence, we obtain o_2 in such linear systems with $v_1 + o_1$ variables (observe that $v_1 o_1 \ge v_1 + o_1$) that are easily solvable. Note that solving each linear system is eventually equivalent to the KRAs using good keys on Rainbow in [Tho13]. Let $\widetilde{T''_j}$ be a good key, where it preserves the j-th column of $\widetilde{T'}$ and the other parts are the same as the identity map. Then, it is enough to solve each linear system from (3) to find unknown variables in $\widetilde{T''_j}$. In our attack, we need to find such good keys as $\widetilde{T''_j}$ for o_2; this is slightly different from the approach of KRAs using good keys in [Tho13].

After substituting the obtained variables into the remaining equations, we obtain a linear system of $o_1^2 o_2$ equations with the remaining $v_1 o_1$ variables in the following form:

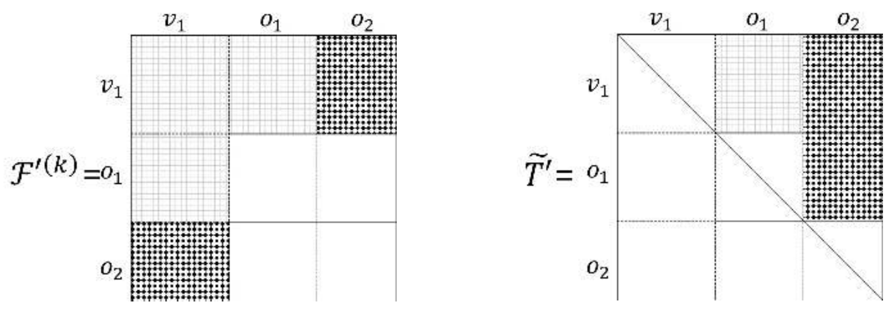
$$f^{(k)}_{ij} = \sum_{y=1}^{v_1} \widetilde{t}_{yi} \left(\sum_{z=1}^{v_1+o_1} P^{(k)}_{yz} t_{zj} + P^{(k)}_{yj} \right) + \sum_{z=1}^{n} P^{(k)}_{iz} t_{zj}. \tag{4}$$

As in (3), all variables in each equation are in a column of $\widetilde{T'}$. For each i with $v_1 + 1 \le i \le v_1 + o_1$, we obtain a smaller linear system of $o_1 o_2$ equations with v_1 variables in the i-th column of $\widetilde{T'}$. Hence, we obtain for o_1 such linear systems with v_1 variables that are easily solvable (observe that $o_1 o_2 \ge v_1$ for most of the suggested parameters of Rainbow).

Finally, we can find $\widetilde{T'}$ in polynomial time by solving o_2 linear systems of $v_1 o_1$ variables and then solving o_1 linear systems of v_1 variables. After recovering T, $\mathcal{F}'$ is also easily computable as $\mathcal{F}' = \mathcal{P} \circ \widetilde{T'}$. Practically, we consider a specific parameter, Rainbow($\mathbb{F}_{2^8}, 36, 21, 22$), which achieves a 128-bit security level. We are able to recover its equivalent key in less than 0.46 milliseconds on Intel Xeon E5-2687W CPU 3.1 GHz with 256GB RAM.

5 CPA on Other MQ-Signature Schemes

We have shown that full secret key recovery on Rainbow-like multi-layered signature schemes via CPA is possible regardless of using the equivalent keys in the form of Fig. 1.



(a) $\mathcal{F}'^{(k)}$ in the first layer ($1 \leq k \leq o_1$) (b) Easily computable part in $\widetilde{T'}$

Figure 8: The forms of the equivalent key $\mathcal{F}'^{(k)}$ and $\widetilde{T'}$

Our CPA can also be applied to the single layer MQ-scheme, UOV, if it is implemented with the equivalent key in the form of Fig. 2. Unlike Rainbow, as all forms of secret central polynomials in UOV are the same, S is unnecessary, i.e., $\mathcal{P} = \mathcal{F} \circ T$. Furthermore, our attack on Rainbow can be applied to UOV because the structure of the equivalent key T' in the form of Fig. 2 is similar to that of Rainbow. However, if the random invertible affine map T is used in UOV implementation, we cannot recover T using the same attack.

Now, we discuss the applicability of our attacks to MQ-schemes submitted to NIST. Seven MQ-signature schemes, LUOV, Rainbow, HiMQ-3, MQDSS, DualModeMS, Gui, and GeMSS, have been submitted to NIST's Post-Quantum Cryptography Standardization [NIS16b]. Our CPA can be applied to the UOV-like single layer scheme LUOV as its design and implementation use the equivalent key in the form of Fig. 2. Our hybrid attack can also be applied to Rainbow-like multi-layered schemes, Rainbow and HiMQ-3. However, our attacks cannot be applied to MQDSS because it does not use the ASA structure.

Finally, by choosing input messages, our CPA can be applied to the other three MQ-signature schemes, DualModeMS, Gui, and GeMSS, based on the ASA structure. More precisely, partial information of the first affine secret map corresponding to S for the other three MQ-signature schemes can be retrieved by the chosen message 1 bit CPA. We cannot control all bits of $\mathbf{h}$ as they use HFEv- schemes. Therefore, we can only recover partial information of the first affine secret map. We believe that the recovery of partial information of S would weaken the security of the schemes. However, for each scheme, depending on additional structures of the central map or the other affine map T, exact analysis should be conducted on the effect of the recovery of S on the recovery of T or some forgery attacks.

6 Countermeasures Against the Proposed CPA

The algebraic KRAs in our attack can only be used when the secret affine map $\widetilde{S}$ is retrieved by CPA. Here, we discuss countermeasures against our CPA to protect the secret affine map $\widetilde{S}$.

■ UOV-like single layer schemes

UOV-like single layer schemes are vulnerable to our attack only when it is implemented using the equivalent key in the form of Fig. 2. If they use random affine maps T instead of the equivalent keys in their implementations, they are secure against our CPA attack.

■ Rainbow-like multi-layered schemes

Unfortunately, Rainbow-like multi-layered schemes are vulnerable to our attack re-

Algorithm 2 Matrix-vector product using shuffling

```
Input: matrix $A \in \mathbb{F}_q^{n\times n}$, vector $\mathbf{x} \in \mathbb{F}_q^n$
Output: vector $\mathbf{x}' \left(= A\mathbf{x}^T\right) \in \mathbb{F}_q^n$
1: for $i = 1$ to $n$ do
2:     $x'_i = 0$
3: end for
4: $\kappa_1 : \mathbb{Z}^*_{n+1} \xrightarrow{R} \mathbb{Z}^*_{n+1}$ , $\kappa_2 : \mathbb{Z}^*_{n+1} \xrightarrow{R} \mathbb{Z}^*_{n+1}$     // Generate random permutations
5: for $i = 1$ to $n$ do
6:     for $j = 1$ to $n$ do
7:         $x'_{\kappa_2(j)} = x'_{\kappa_2(j)} + a_{\kappa_2(j)\kappa_1(i)} \cdot x_{\kappa_1(i)}$
8:     end for
9: end for
10: return $\mathbf{x}'$
```

gardless of the use of equivalent keys in the form of Fig. 1. Therefore, we must focus on implementing a secure algorithm against PA. PA exploits the fact that the power consumption of cryptographic devices depends on intermediate values of the operated cryptographic algorithms. There are some types of countermeasures to eliminate or reduce these dependencies. A good overview of DPA countermeasures is available in [MOP07]. The most commonly used countermeasures are hiding and masking techniques at the algorithmic level. Random insertion of dummy operations and shuffling of operations are commonly used as a hiding technique because of flexibility in software. For example, each time the algorithm runs, the order of the loading rows or columns of the matrix A in Algorithm 1 can be changed.

Algorithm 2 shows the matrix-vector product using the shuffling countermeasure. Here, κ_1 and κ_2 are random permutations of a set of length n. The classical algorithm for random permutation generation can be found in [Knu81], which has been known as a linear complexity. The statistical effects of shuffling in terms of PA have been studied [CCD00, Man04]. It is generally known that if the probability that an intermediate value occurs at a certain time is p, then the number of traces needed for a successful attack increases by a factor of $\frac{1}{p^2}$ [HOM06]. In Algorithm 2, the probability that an intermediate value in our proposed matrix-vector product using shuffling occurs at a certain time is $\frac{1}{n^2}$. Therefore, the number of traces needed for a successful attack increases by $(n^2)^2$ times.

Another approach is to use a logical masking method with random numbers. For example, message randomization is a widely used method to prevent DPA against RSA, which can be expressed as:

$$A \equiv m^d \bmod N \equiv (mr^e)^d\, r^{-1} \bmod N,$$

where r and m represent a random number and a message, respectively. N is the public modulus, and the public key e and the private key d are linked to each other by the equation $e \cdot d \equiv 1 \bmod \varphi(N)$, where $\varphi(\cdot)$ denotes Euler's function. Similarly, we can use message randomization to prevent our attack when $\widetilde{S}$ and $\mathbf{h}$ are multiplied.

$$\widetilde{S}\mathbf{h} = \left(\widetilde{S}\left(\mathbf{h} * r\right)\right) * r^{-1},$$

where $*$ denotes a vector and scalar product. Algorithm 3 shows the pseudo-code of the matrix-vector product obtained using message randomization, and Table 2 shows the comparison of operation counts for Algorithm 1 and 3. As can be seen in Table 2, the matrix-vector obtained using message randomization uses more $2n$ field multiplications and a field inversion as compared with the general matrix-vector product.

Table 2: Comparing operation counts for Algorithm 1 and 3

	Algorithm 1	Algorithm 3
Field Multiplication	n^2	$n^2 + 2n$
Field Addition	n^2	n^2
Field Inversion	–	1

Algorithm 3 Matrix-vector product using the message randomization

Input: matrix $A \in \mathbb{F}_q^{n\times n}$, vector $\mathbf{x} \in \mathbb{F}_q^n$
Output: vector $\mathbf{x}' \left(= A\mathbf{x}^T\right) \in \mathbb{F}_q^n$

```
1:  for $i = 1$ to $n$ do
2:      $x'_i = 0$
3:  end for
4:  $r \in_R \mathbb{F}_q^*$     // The notation $\in_R$ stands for randomly sampling and $\mathbb{F}_q^*$ means $\mathbb{F}_q\backslash\{0\}$
5:  for $i = 1$ to $n$ do
6:      $x_i = x_i \cdot r$
7:  end for
8:  for $i = 1$ to $n$ do
9:      for $j = 1$ to $n$ do
10:         $x'_j = x'_j + a_{ji} \cdot x_i$
11:     end for
12: end for
13: for $i = 1$ to $n$ do
14:     $x'_i = x'_i \cdot r^{-1}$
15: end for
16: return $\mathbf{x}'$
```

This countermeasure ensures the prevention of our proposed attacks, however, the scheme would still be vulnerable to sophisticated attacks (such as high-order DPA). We do not discuss countermeasures against sophisticated attacks here as they are out of the scope of this paper. Finding the optimal method is not easy, hence, our future work includes designing a masking scheme that adapts to the limitations of each implementation platform.

7 Conclusion

We showed that, only via CPA, we succeeded in recovering a full secret key on Rainbow implemented by the equivalent keys in the form of Fig. 1 due to the special structure of the equivalent keys. We also demonstrated how this leakage on Rainbow can be exploited in practice on an 8-bit AVR microcontroller using CPA. Next, we extended the full secret key recovery to the general case using random affine maps via a hybrid attack: after recovering S by performing CPA, we recovered T by mounting algebraic KRAs. The same attack can be applied to UOV when it is implemented with the equivalent key $\widetilde{T'}$ in the form of Fig. 2. Our attacks can also be applied to Rainbow-like multi-layered signature schemes regardless of using the equivalent keys in the form of Fig. 1 and UOV-like single layer signature schemes with the implementations using the equivalent key $\widetilde{T'}$ in the form of Fig. 2 submitted to NIST for Post-Quantum Cryptography Standardization. It is the first result on the security of MQ-signature schemes only using CPA.

Acknowledgments

The authors would like to thank the anonymous reviewers for their useful comments that improved the quality of the paper. We also thank Yoo-Jin Baek and Bo-Yeon Sim for their valuable feedback and for many helpful discussions.

References

[BERW08] Andrey Bogdanov, Thomas Eisenbarth, Andy Rupp, and Christopher Wolf. Time-area optimized public-key engines: Mq-cryptosystems as replacement for elliptic curves? In *Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings*, pages 45–61, 2008.

[BFP09] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *J. Mathematical Cryptology*, 3(3):177–197, 2009.

[CCC$^+$09] Anna Inn-Tung Chen, Ming-Shing Chen, Tien-Ren Chen, Chen-Mou Cheng, Jintai Ding, Eric Li-Hsiang Kuo, Frost Yu-Shuang Lee, and Bo-Yin Yang. SSE implementation of multivariate pkcs on modern x86 cpus. In *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, pages 33–48, 2009.

[CCD00] Christophe Clavier, Jean-Sébastien Coron, and Nora Dabbous. Differential power analysis in the presence of hardware countermeasures. In *Cryptographic Hardware and Embedded Systems - CHES 2000, Second International Workshop, Worcester, MA, USA, August 17-18, 2000, Proceedings*, pages 252–263, 2000.

[CEvMS15] Cong Chen, Thomas Eisenbarth, Ingo von Maurich, and Rainer Steinwandt. Differential power analysis of a mceliece cryptosystem. In *Applied Cryptography and Network Security - 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers*, pages 538–556, 2015.

[CHT12] Peter Czypek, Stefan Heyse, and Enrico Thomae. Efficient implementations of MQPKS on constrained devices. In *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, pages 374–389, 2012.

[CMP18] Laurent Castelnovi, Ange Martinelli, and Thomas Prest. Grafting trees: A fault attack against the SPHINCS framework. In *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, pages 165–184, 2018.

[COD] `https://github.com/fast-crypto-lab/mpkc-128bit`.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005, Proceedings*, pages 164–175, 2005.

[DYC$^+$08] Jintai Ding, Bo-Yin Yang, Chia-Hsin Owen Chen, Ming-Shing Chen, and Chen-Mou Cheng. New differential-algebraic attacks and reparametrization of rainbow. In *Applied Cryptography and Network Security, 6th International*

Conference, ACNS 2008, New York, NY, USA, June 3-6, 2008. Proceedings, pages 242–257, 2008.

[EFGT17] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongswan and electromagnetic emanations in microcontrollers. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1857–1874, 2017.

[EGHP09] Thomas Eisenbarth, Tim Güneysu, Stefan Heyse, and Christof Paar. Microeliece: Mceliece for embedded devices. In *Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings*, pages 49–64, 2009.

[GMO01] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic analysis: Concrete results. In *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, number Generators, pages 251–261, 2001.

[HMP10] Stefan Heyse, Amir Moradi, and Christof Paar. Practical power analysis attacks on software implementations of mceliece. In *Post-Quantum Cryptography, Third International Workshop, PQCrypto 2010, Darmstadt, Germany, May 25-28, 2010. Proceedings*, pages 108–125, 2010.

[HOM06] Christoph Herbst, Elisabeth Oswald, and Stefan Mangard. An AES smart card implementation resistant to power analysis attacks. In *Applied Cryptography and Network Security, 4th International Conference, ACNS 2006, Singapore, June 6-9, 2006, Proceedings*, pages 239–252, 2006.

[HTS11] Yasufumi Hashimoto, Tsuyoshi Takagi, and Kouichi Sakurai. General fault attacks on multivariate public key cryptosystems. In *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*, pages 1–18, 2011.

[KGB$^+$18] Matthias J. Kannwischer, Aymeric Genêt, Denis Butin, Juliane Krämer, and Johannes Buchmann. Differential power analysis of XMSS and SPHINCS. In *Constructive Side-Channel Analysis and Secure Design - 9th International Workshop, COSADE 2018, Singapore, April 23-24, 2018, Proceedings*, pages 168–188, 2018.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 388–397, 1999.

[Knu81] Donald E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms, 2nd Edition.* Addison-Wesley, 1981.

[Koc96] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, pages 104–113, 1996.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, pages 206–222, 1999.

[KY11] Abdel Alim Kamal and Amr M. Youssef. Fault analysis of the ntruencrypt cryptosystem. *IEICE Transactions*, 94-A(4):1156–1158, 2011.

[LSCH10] Mun-Kyu Lee, Jeong Eun Song, Dooho Choi, and Dong-Guk Han. Countermeasures against power analysis attacks for the NTRU public key cryptosystem. *IEICE Transactions*, 93-A(1):153–163, 2010.

[Man04] Stefan Mangard. Hardware countermeasures against DPA ? A statistical analysis of their effectiveness. In *Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings*, pages 222–235, 2004.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynominal-tuples for efficient signature-verification and message-encryption. In *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*, pages 419–453, 1988.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks - revealing the secrets of smart cards.* Springer, 2007.

[MSSS11] H. Gregor Molter, Marc Stöttinger, Abdulhadi Shoufan, and Falko Strenzke. A simple power analysis attack on a mceliece cryptoprocessor. *J. Cryptographic Engineering*, 1(1):29–36, 2011.

[New] NewAE. ChipWhisperer. https://wiki.newae.com/CW1173_ChipWhisperer-Lite.

[NIS16a] NIST. Post-quantum cryptography: NIST's plan for the future. http://csrc.nist.gov/groups/ST/post-quantum-crypto/documents/pqcrypto-2016-presentation.pdf, 2016.

[NIS16b] NIST. Post-Quantum Cryptography, Round 1 Submissions, NIST Computer Security Resource Center. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions, 2016.

[NSA15] NSA. NSA suite B cryptography. https://www.nsa.gov/ia/programs/suiteb_cryptography/, 2015. [Online; Updated on August 19].

[Pat96] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 33–48, 1996.

[PBY17] Peter Pessl, Leon Groot Bruinderink, and Yuval Yarom. To BLISS-B or not to be: Attacking strongswan's implementation of post-quantum signatures. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1843–1855, 2017.

[PCG01] Jacques Patarin, Nicolas Courtois, and Louis Goubin. Quartz, 128-bit long digital signatures. In *Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001, San Francisco, CA, USA, April 8-12, 2001, Proceedings*, pages 282–297, 2001.

[PCY^{+}15] Albrecht Petzoldt, Ming-Shing Chen, Bo-Yin Yang, Chengdong Tao, and Jintai Ding. Design principles for hfev- based multivariate signature schemes. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, pages 311–334, 2015.

[SGB01] Rainer Steinwandt, Willi Geiselmann, and Thomas Beth. A theoretical dpa-based cryptanalysis of the NESSIE candidates FLASH and SFLASH. In *Information Security, 4th International Conference, ISC 2001, Malaga, Spain, October 1-3, 2001, Proceedings*, pages 280–293, 2001.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[SW07] Joseph H. Silverman and William Whyte. Timing attacks on ntruencrypt via variation in the number of hash calls. In *Topics in Cryptology - CT-RSA 2007, The Cryptographers' Track at the RSA Conference 2007, San Francisco, CA, USA, February 5-9, 2007, Proceedings*, pages 208–224, 2007.

[Tho13] Enrico Thomae. *About the security of multivariate quadratic public key schemes.* PhD thesis, Ruhr-University Bochum, 2013.

[vMG14a] Ingo von Maurich and Tim Güneysu. Lightweight code-based cryptography: QC-MDPC mceliece encryption on reconfigurable devices. In *Design, Automation & Test in Europe Conference & Exhibition, DATE 2014, Dresden, Germany, March 24-28, 2014*, pages 1–6, 2014.

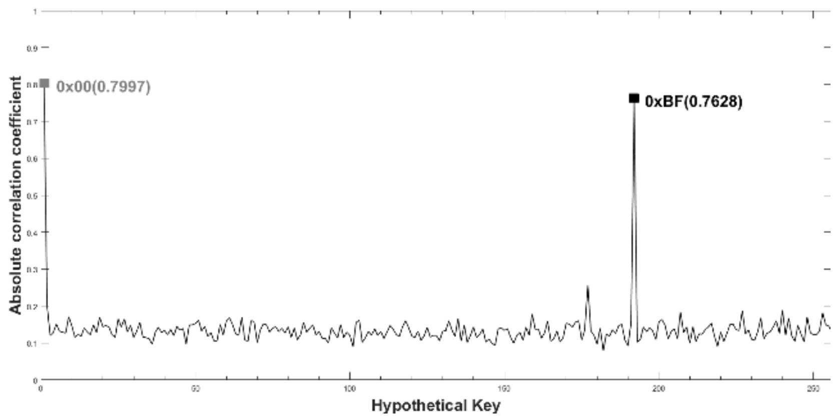
[vMG14b] Ingo von Maurich and Tim Güneysu. Towards side-channel resistant implementations of QC-MDPC mceliece encryption on constrained devices. In *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, pages 266–282, 2014.

[WP05] Christopher Wolf and Bart Preneel. Large superfluous keys in multivariate quadratic asymmetric systems. In *Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings*, pages 275–287, 2005.

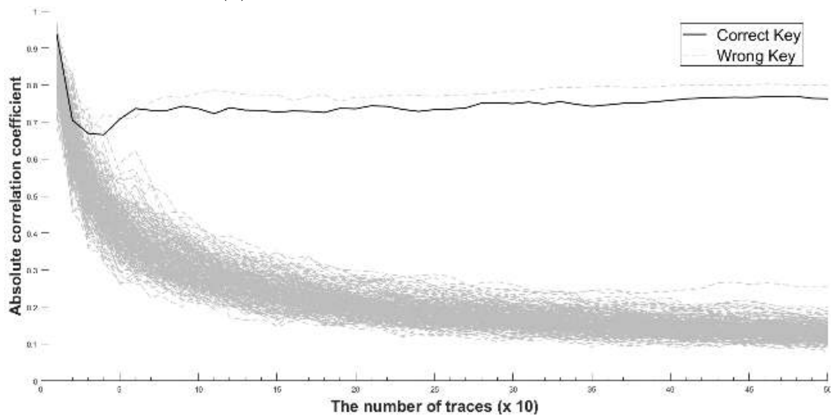
[YL17] Haibo Yi and Weijian Li. On the importance of checking multivariate public key cryptography for side-channel attacks: The case of entts scheme. *Comput. J.*, 60(8):1197–1209, 2017.

[ZWW13] Xuexin Zheng, An Wang, and Wei Wei. First-order collision attack on protected NTRU cryptosystem. *Microprocessors and Microsystems - Embedded Hardware Design*, 37(6-7):601–609, 2013.

Appendix A CPA Results for $\widetilde{s}_{13}$ at First Attack Point



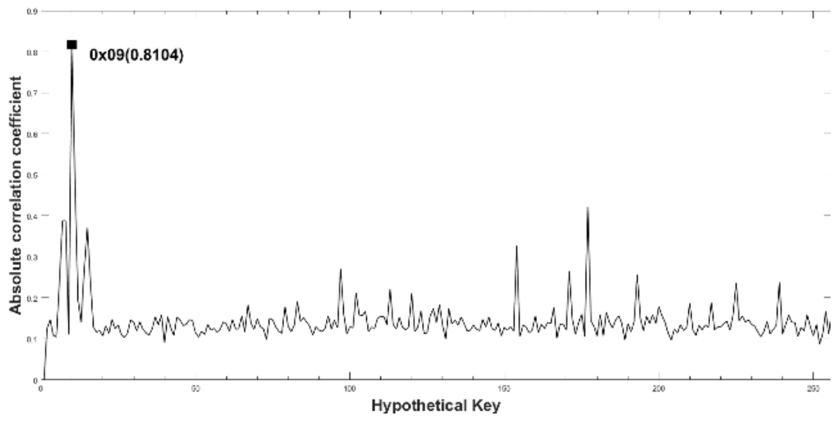
(a) Maximum correlation coefficients



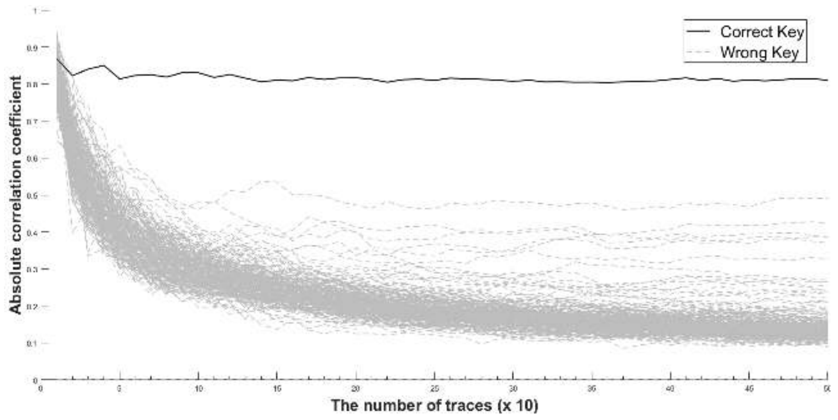
(b) Maximum correlation coefficients according to increased number of traces

Figure 9: CPA results for $\widetilde{s}_{13}$ at first position

Appendix B CPA Results for Random $\widetilde{s}_{11}$



(a) Maximum correlation coefficients for $\widetilde{s}_{11}$



(b) Maximum correlation coefficients according to increased number of traces for $\widetilde{s}_{11}$

Figure 10: CPA results for random $\widetilde{s}_{11}$