國立臺灣大學電機資訊學院電子工程研究所

碩士論文

Graduate Institute of Electronics Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

NewHope 二項式抽樣模板攻擊
A Template Attack on Binomial Sampling in NewHope

彭俊又

CHUN-YU PENG

指導教授：鄭振牟博士

Advisor: Chen-Mou Cheng, Ph.D.

中華民國 108 年 7 月

July, 2019

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

摘要

NewHope 是一個被看好極有可能成為未來後量子密碼系統的演算法。在分析它抵禦量子電腦攻擊的安全性之餘，密碼系統實作的安全性也是一項重要的議題。本論文中，我們首先分析 NewHope 演算法中可能成為旁通道分析目標的模組。接著，我們針對其二項式抽樣的實作進行旁通道模板攻擊。實驗的結果顯示出攻擊者可以從單一次的功率消耗波形中，以 100% 的成功率分析出二項式抽樣出的秘密參數。

關鍵字： NewHope, 後量子密碼學, 旁通道分析, 模板攻擊

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

Abstract

The NewHope cryptosystem is a promising candidate for the future post-quantum cryptography standard. Besides its security against the attacks from quantum and classical computers, the side-channel security is also an important issue to the implementation of a cryptosystem. In this thesis, we first evaluate the potential side-channel vulnerabilities in the NewHope cryptosystem. Then, a template attack is presented, which can reveal the secret information generated by the Binomial Sampling Function and compromise the security of the cryptosystem. The result shows a 100% success rate of recovering the secrets by only using a single side-channel power consumption trace.

Keywords: NewHope, PQC, Side-Channel Analysis, Template Attack.

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

Contents

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

List of Figures

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

List of Tables

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

Chapter 1

Introduction

Cryptographic primitives are the foundation of information security. However, the world of cryptography is constantly under the threat of newer and better cryptanalysis techniques. At the end of the 1990s, the exponential growth of the computing power prompted the National Institute of Standards and Technology (NIST) to migrate the symmetric encryption standard from DES to AES. Later, the discovery of side-channel analysis opens a new area of cryptanalysis and completely revolutionizes the implementation aspect of cryptosystems. Recently, as the development of quantum computers once again threatens the security of the public key cryptosystems, NIST initiated the Post-Quantum Cryptography (PQC) Standardization Project in response and trying to find a new set of public key cryptosystems that can withstand the attacks from quantum computers. The standardization project brings new energy into the community of cryptographic research, which has come up with a large number of proposals. While evaluating the mathematical soundness of the candidates against both classical and quantum computers, it is also important to investigate whether the implementations are secure against the uprising side-channel attacks.

NewHope is one of the lattice-based key-exchange protocol, which was submitted to the NIST PQC project. Because of the simple scheme and good performance, it is considered a promising candidate. In this work, we will discuss the potential risk of the straightforward implementation of the NewHope cryptosystem. We will show a single trace attack on a microprocessor, based solely on the side-channel information. The attack can compromise the security of NewHope by extracting the secret information, including

doi:10.6342/NTU201903050

the secret key, generated by the Binomial Sampling Function.

1.1 NIST PQC Standardization Project

Twenty years ago, Shor discovered a quantum algorithm that can exponentially speed up the integer factorization and discrete logarithm problems, which are the mathematical hard problem behind the RSA and ECC-based cryptosystem. At the time, it was not clear whether or not building a large scale quantum computer is possible. Therefore, these public key cryptosystems were not considered vulnerable in practice.

In the past decade, a large amount of research energy has been put into developing quantum computers. This type of computer can use the quantum mechanical phenomena to solve problems that are unmanageable for classical computers. Large tech companies such as Google, Intel, and IBM are all developing their own quantum computers ranging from 20-qubits to 72-qubits. It is estimated that significantly large-scale quantum computers will come to reality within 20 years. As a result, it will compromise the confidentiality and integrity of the digital world, and shatter the security of the public key cryptosystems that are currently in used, including the RSA, DSA, and ECC-based public key cryptographic schemes. To prepare for the upcoming threat to the public key cryptosystems in the near future, NIST initiated the Post-Quantum Cryptography Standardization Project in February 2016. The goal is to develop new cryptographic systems that are secure against both quantum and classical computers.

The post-quantum cryptosystems can be categorized into several families, including the lattice-based cryptosystem, multivariate cryptosystem, code-based cryptosystem, hash-based signature schemes, and Supersingular isogeny key exchange. Some of the schemes have even been around for several decades, without any known attacks. However, it is stated in the *Report on Post-Quantum Cryptography* by NIST [8], that more research and analysis are required before selecting any of the algorithms as standard.

There are several reasons why NIST decided to start the standardization project. First, the progress of the development of quantum computers has increased drastically in recent years. Second, the transition from the current public key cryptosystems to post-quantum

doi:10.6342/NTU201903050

ones might be tricky, since most of the post-quantum schemes have a significantly larger key size than the currently used systems. Therefore, it might take a few years to transition to the new systems. Third, the evaluation process of the post-quantum schemes is expected to be more complicated than the previous SHA-3 and AES competitions, since the algorithms themselves are more complex. Also, there has not been a sufficient amount of research energy until now. Since the mathematical foundation of the different post-quantum families is quite different, it is hard to directly compare some of the schemes. As a result, it is expected to have several candidate cryptosystems being selected as the standard, and serve as replacements for the current public key cryptosystems, including the digital signature, public key encryption, and key establishment algorithms.

The formal call for proposal was announced on December 20, 2016. Upon the submission deadline on November 30, 2017, there were 82 proposals submitted to the competition. 69 of the submissions were deemed "complete and proper", meaning they have met the minimum acceptance criteria and the submission requirements given by NIST. Three weeks after the submissions were made public, 12 schemes were found insecure or even broken, and some were withdrawn. After a year of evaluation, the second-round candidates were announced on January 30, 2019, and there are 26 candidates left in the competition, including 17 public key encryption and key establishment schemes, and 9 digital signature schemes.

The selection criteria in the first round are security, cost and performance, and algorithm and implementation characteristics, in the order of importance. For the twelve to eighteen months following the second-round candidates announcement, the remaining candidates, with their updated parameters and implementations, will once again be kept under a thorough inspection both by NIST and the cryptographic community. In the later stage of the selection process, the implementation aspect will play a larger role in the evaluation process. NIST is hoping to see the performance data on both software and hardware, such as on the Cortex-M4 microprocessor and the Artix-7 FPGA, as well as the optimized implementation, including using the assembly code and the instruction set extension. The resistance to side-channel attacks is also an important issue in both algorithmic design and

doi:10.6342/NTU201903050

implementations [1].

1.2 Roadmap

This thesis is composed of six chapters.

Chapter 2 introduces some notations and the mathematical hard problem behind the NewHope key establishment scheme. Then, a brief description of the algorithm is given in the following section.

Chapter 3 presents several leakage assessment techniques that can be used to evaluate side-channel characteristics. Then, the framework of the template attack is described, and different data reduction methods are presented.

Chapter 4 gives a detailed evaluation of the NewHope key encapsulation mechanism. Then, we analyze the side-channel properties of the Binomial Sampling Function in NewHope and identify the weakness in the implementation.

Chapter 5 shows the template attacks on an ARM Cortex-M4 microprocessor. The result supports our analysis on the Binomial Sampling Function that the implementation is indeed vulnerable to the template attack.

Chapter 6 concludes the template attack on NewHope.

doi:10.6342/NTU201903050

Chapter 2

NewHope

The NewHope cryptosystem is a key-encapsulation mechanism (KEM) based on the conjectured quantum hardness of the Ring-LWE problem [20]. It was submitted to the NIST post-quantum cryptography standardization project, and remaining a candidate algorithm in the second round evaluation process [1]. The parameter choice of NewHope allows the utilization of Number Theoretic Transform (NTT), making the computation efficient in both time and memory usage. An experiment conducted by Google had shown no problem integrating NewHope into the TLS key agreement in a development version of the Chrome browser, with only a slight increase of the latency by milliseconds, which is caused by the larger message size of the post-quantum cryptography scheme. Another one of the largest security chip company in the world, Infineon, has also claimed to have already deployed a variant of NewHope on their commercially available contactless security chip product.

2.1 Preliminaries

2.1.1 Mathematical background

In the NewHope cryptosystem, all elements are polynomials over the ring structure: $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$, which is the ring of integer polynomials modulo $X^n + 1$, where each coefficient is in the field of modulus q. In other words, each of the elements in the cryp-

doi:10.6342/NTU201903050

tosystem is a polynomial with n coefficients, and all the coefficients are integers within the range $[0, q-1]$. In the following, the bold letters are used to refer to the elements in $\mathcal{R}_q$. There are two parameter settings, $(n, q) = (512, 12289)$ and $(n, q) = (1024, 12289)$, matching the NIST security level 1 and 5 respectfully.

2.1.2 Ring Learning with Errors

The original Learning with Errors (LWE) problem was introduced in [22], and the problem is describe as follows:

There is a secret vector $\boldsymbol{s}$, chosen randomly in $\mathbb{Z}_q$. Given a set of random linear equations on $\boldsymbol{s}$ with 'errors', the goal is to recover the secret vector $\boldsymbol{s}$. For instance, given the following set of 'errored linear equations':

$$\begin{aligned}
8s_1 + 6s_2 + 5s_3 + 10s_4 + e_0 &= 7 \pmod{17} \\
5s_1 + 7s_2 + 4s_3 + 6s_4 + e_1 &= 5 \pmod{17} \\
6s_1 + 3s_2 + 3s_3 + 2s_4 + e_2 &= 15 \pmod{17} \\
4s_1 + 1s_2 + 9s_3 + 7s_4 + e_3 &= 10 \pmod{17} \\
&\vdots \\
3s_1 + 6s_2 + 2s_3 + 7s_4 + e_n &= 15 \pmod{17}
\end{aligned}$$

with the errors e_i within the range ± 1. The objective is to find the secret vector $\boldsymbol{s}$, which in this case is $(s_1, s_2, s_3, s_4) = (1, 9, 8, 4)$.

Without the errors, finding the secret $\boldsymbol{s}$ would be as simple as solving a set of linear equations, which can be done using Gaussian elimination, as long as the number of equations exceeds the number of variables in $\boldsymbol{s}$. However, in the problem of Learning with Errors, with each new equation, a new variable e_i is introduced. Therefore, the number of equations would never exceed the number of variables, making the problem impossible to solve with Gaussian elimination.

In the Ring-LWE problem, all the elements are in the polynomial ring $\mathcal{R}_q$. The problem is to find the secret $\boldsymbol{s}$, given two polynomials $(\boldsymbol{a}, \boldsymbol{b} = \boldsymbol{a} \circ \boldsymbol{s} + \boldsymbol{e})$, where $\boldsymbol{a}$ is uniformly

doi:10.6342/NTU201903050

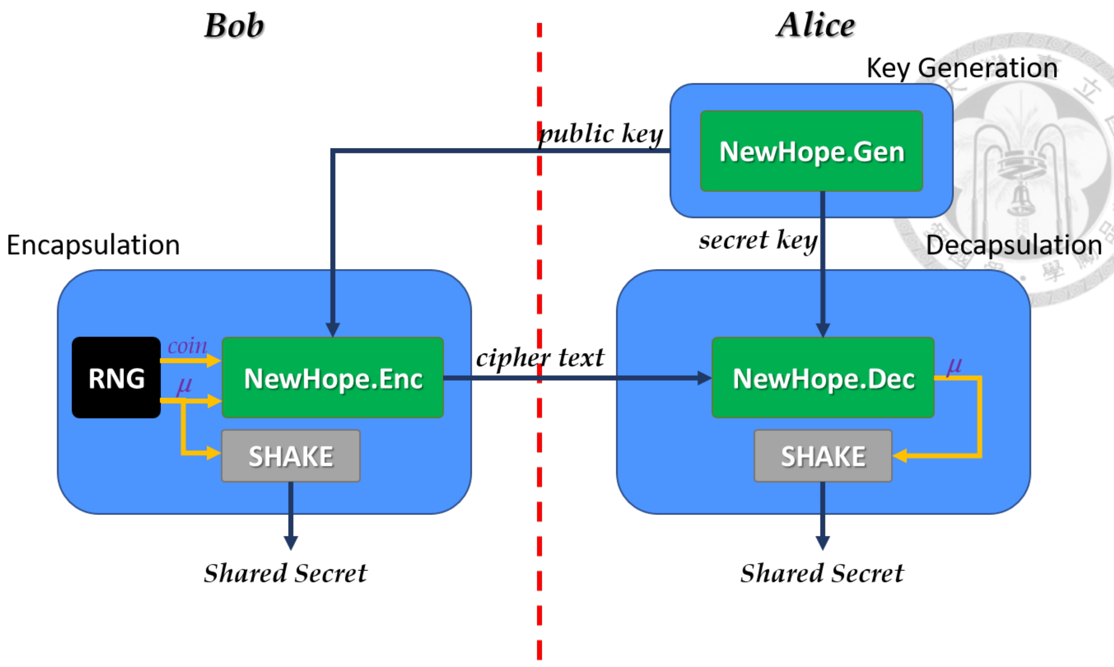


Figure 2.1: NewHope Key Encapsulation Mechanism.

distributed in $\mathcal{R}_q$, and $(\boldsymbol{s}, \boldsymbol{e})$ are randomly sampled following an error distribution. It is shown in [14], that the hardness of Ring-LWE is similar to solving the Shortest Vector Problem (SVP) on an ideal lattice.

2.2 Algorithm Description

NewHope is a Ring-LWE based key encapsulation mechanism and is a variant of the NewHope-Simple scheme [2]. Different from the earlier NewHope-Usenix scheme [3], NewHope uses encryption-based key exchange method rather than the reconciliation method. The concept of the NewHope key encapsulation mechanism is shown in Figure 2.1, where Bob takes Alice's public key to encrypt a randomly generated key material. Alice then uses her secret key to recover the key material. The shared secret is then derived from the key material using SHAKE256. The algorithm can be separated into three steps: Key Generation, Encapsulation, and Decapsulation. In this section, we will describe a simplified version of these three operations, where the details such as the NTT and the message encode/decode are omitted.

doi:10.6342/NTU201903050

2.2.1 Key Generation

A public and secret key pair is generated in the key generation process. The algorithm is shown in Algorithm 1, where a 'large' polynomial $\boldsymbol{a}$ is generated by the GenA function, which generates a polynomial with the coefficients uniformly distributed in $[0, q-1]$. Another two 'small' polynomials $\boldsymbol{s}$ and $\boldsymbol{e}$ are generated by the Sample function, shown in Algorithm 3, where the coefficients are binomial distributed in the range $[-8, 8]$. The public key is the two polynomials $(\boldsymbol{a}, \boldsymbol{b} = \boldsymbol{a} \circ \boldsymbol{s} + \boldsymbol{e})$, and the secret key is $\boldsymbol{s}$.

Algorithm 1 NewHope Key Generation

```
1: function NewHope-KeyGen()
2:     $seed \xleftarrow{\$} \{0, \ldots, 255\}^{32}$
3:     $publicseed || noiseseed \leftarrow$ SHAKE256($64, seed$) $\in \{0, \ldots, 255\}^{32+32}$
4:     $\boldsymbol{a} \leftarrow$ GenA($publicseed$)
5:     $\boldsymbol{s} \leftarrow$ Sample($noiseseed, 0$)
6:     $\boldsymbol{e} \leftarrow$ Sample($noiseseed, 1$)
7:     $\boldsymbol{b} \leftarrow \boldsymbol{a} \circ \boldsymbol{s} + \boldsymbol{e}$
8:     return $(publickey = (\boldsymbol{a}, \boldsymbol{b}), secretkey = \boldsymbol{s})$
```

2.2.2 Encapsulation

A randomly generated key material is encrypted and sent to Alice in the encapsulation process. The function is described in Algorithm 2. A 256-bit random key material μ is encoded into a polynomial $\boldsymbol{K}$ with coefficients either 0 or q/2 depending on the random key bit, shown in Algorithm 4. Another three 'small' polynomials $\boldsymbol{s'}$, $\boldsymbol{e'}$ and $\boldsymbol{e''}$ are again generated by the Sample function. The message is consist of two polynomials $(\boldsymbol{c}_1, \boldsymbol{c}_2)$, where $\boldsymbol{c}_1 = \boldsymbol{a} \circ \boldsymbol{s'} + \boldsymbol{e'}$, and $\boldsymbol{c}_2 = \boldsymbol{b} \circ \boldsymbol{s'} + \boldsymbol{e''} + \boldsymbol{K}$. The message c is then sent to Alice, while the 32-byte shared secret ss is derived from the key material μ with SHAKE256.

2.2.3 Decapsulation

In the decapsulation process, the key material is recovered by decoding $\boldsymbol{c}_2 - \boldsymbol{c}_1 \circ \boldsymbol{s}$, and the shared secret is then derived from the key material with SHAKE 256. The function is shown in Algorithm 5. It is shown below that with the help of the secret polynomial $\boldsymbol{s}$,

doi:10.6342/NTU201903050

Algorithm 2 NewHope Encapsulation

```
1: function NewHope-Encaps($publickey$)
2:     $coin \xleftarrow{\$} \{0, \ldots, 255\}^{32}$
3:     $\mu||coin' \leftarrow$ SHAKE256($64, coin$) $\in \{0, \ldots, 255\}^{32+32}$
4:     $\boldsymbol{a}, \boldsymbol{b} \leftarrow publickey$
5:     $\boldsymbol{s'} \leftarrow$ Sample($coin', 0$)
6:     $\boldsymbol{e'} \leftarrow$ Sample($coin', 1$)
7:     $\boldsymbol{e''} \leftarrow$ Sample($coin', 2$)
8:     $\boldsymbol{c}_1 \leftarrow \boldsymbol{a} \circ \boldsymbol{s'} + \boldsymbol{e'}$
9:     $\boldsymbol{K} \leftarrow$ Encode($\mu$)
10:    $\boldsymbol{c}_2 \leftarrow \boldsymbol{b} \circ \boldsymbol{s'} + \boldsymbol{e''} + \boldsymbol{K}$
11:    $ss \leftarrow$ SHAKE256($32, \mu$)
12:    return $(ss, c = (\boldsymbol{c}_1, \boldsymbol{c}_2))$
```

Algorithm 3 NewHope Binomial Sampling Function

```
1: function NewHope-Sample($seed \in \{0, \ldots, 255\}^{32}, nonce$)
2:     $\boldsymbol{r} \leftarrow \mathcal{R}_q$
3:     $extseed \leftarrow \{0, \ldots, 255\}^{34}$
4:     $extseed[0:32] \leftarrow seed||nonce$
5:     for $i$ from $0$ to $(n/64) - 1$ do
6:         $extseed[33] \leftarrow i$
7:         $buf \leftarrow$ SHAKE256($128, extseed$)
8:         for $j$ from $0$ to $63$ do
9:             $a \leftarrow buf[2 * j]$
10:            $b \leftarrow buf[2 * j + 1]$
11:            $r_{64*i+j} \leftarrow$ HW($a$) − HW($b$) $\mod q$
12:    return $\boldsymbol{r} \in \mathcal{R}_q$
```

Algorithm 4 NewHope Message Encoding Function

```
1: function NewHope-Encode($\mu \in \{0, \ldots, 255\}^{32}$)
2:     $\boldsymbol{r} \leftarrow \mathcal{R}_q$
3:     for $i$ from $0$ to $31$ do
4:         for $j$ from $0$ to $7$ do
5:             $mask \leftarrow -((\mu[i] \gg j)\&1)$
6:             $r_{8*i+j+0} \leftarrow mask\&(q/2)$
7:             $r_{8*i+j+256} \leftarrow mask\&(q/2)$
8:             if $n$ equals $1024$ then
9:                 $r_{8*i+j+512} \leftarrow mask\&(q/2)$
10:                $r_{8*i+j+768} \leftarrow mask\&(q/2)$
11:    return $\boldsymbol{r} \in \mathcal{R}_q$
```

doi:10.6342/NTU201903050

Alice can recover $\boldsymbol{K}$ with a relatively small error $\boldsymbol{e}'''$.

$$\begin{aligned}
&\boldsymbol{c}_2 - \boldsymbol{c}_1 \circ \boldsymbol{s} \\
=&(\boldsymbol{b} \circ \boldsymbol{s}' + \boldsymbol{e}'' + \boldsymbol{K}) - (\boldsymbol{a} \circ \boldsymbol{s}' + \boldsymbol{e}') \circ \boldsymbol{s} \\
=&((\boldsymbol{a} \circ \boldsymbol{s} + \boldsymbol{e}) \circ \boldsymbol{s}' + \boldsymbol{e}'' + \boldsymbol{K}) - (\boldsymbol{a} \circ \boldsymbol{s}' + \boldsymbol{e}') \circ \boldsymbol{s} \\
=&(\boldsymbol{a} \circ \boldsymbol{s} \circ \boldsymbol{s}' + \boldsymbol{e} \circ \boldsymbol{s}' + \boldsymbol{e}'' + \boldsymbol{K}) - (\boldsymbol{a} \circ \boldsymbol{s}' \circ \boldsymbol{s} + \boldsymbol{e}' \circ \boldsymbol{s}) \\
=&\boldsymbol{e} \circ \boldsymbol{s}' + \boldsymbol{e}'' + \boldsymbol{K} + \boldsymbol{e}' \circ \boldsymbol{s} = \boldsymbol{K} + \boldsymbol{e}'''
\end{aligned}$$

The decoding step can then remove the remaining errors $\boldsymbol{e}'''$ and recover the encapsulated key material μ as shown in Algorithm 6.

Algorithm 5 NewHope Decapsulation

```
1: function NewHope-Decaps($c, secretkey$)
2:     $(\boldsymbol{c}_1, \boldsymbol{c}_2), \boldsymbol{s} \leftarrow c, secretkey$
3:     $\mu \leftarrow$ Decode($\boldsymbol{c}_2 - \boldsymbol{c}_1 \circ \boldsymbol{s}$)
4:     $ss \leftarrow$ SHAKE256($32, \mu$)
5:     return $ss$
```

Algorithm 6 NewHope Message Decoding Function

```
1: function NewHope-Decode($\boldsymbol{r} \in \mathcal{R}_q$)
2:     $\mu \leftarrow \{0, \ldots, 255\}^{32}$
3:     for $i$ from $0$ to $255$ do
4:         $t \leftarrow |(r_{i+0} \mod q) - (q-1)/2|$
5:         $t \leftarrow t + |(r_{i+256} \mod q) - (q-1)/2|$
6:         if $n$ equals $1024$ then
7:             $t \leftarrow t + |(r_{i+512} \mod q) - (q-1)/2|$
8:             $t \leftarrow t + |(r_{i+768} \mod q) - (q-1)/2|$
9:             $t \leftarrow t - q$
10:        else
11:            $t \leftarrow t - q/2$
12:        $t \leftarrow t \gg 15$
13:        $\mu[i \gg 3] \leftarrow \mu[i \gg 3] | (t \ll (i \& 7))$
14:    return $\mu \in \{0, \ldots, 255\}^{32}$
```

doi:10.6342/NTU201903050

Chapter 3

Side-Channel Analysis and Template Attack

3.1 Side-Channel Analysis

Side-channel analysis is the type of attack that exploits the physical information acquired from an operating cryptographic device to reveal the secrets within the device. This type of attack does not require a deep mathematical understanding of cryptographic systems, but rather a wide range of knowledge from hardware designs to signal processing and statistical analysis techniques. Throughout the years, different types of side channels have been found to leak the information that can weaken the security of a cryptosystem, including computational time, power consumption, EM radiation, acoustic wave, or even the heat signal generated by the device. The compromising side-channel information is also called leakages.

Since the cryptographic algorithms need to be run on an electrical device, such as a microprocessor or a specialized hardware circuit, it is common to have information related to the cryptosystem leaked through different physical channels. Different types of countermeasures have been proposed to minimize or avoid side-channel leakages. For instance, using a constant time computation algorithm to avoid timing leakage, or applying the masking and hiding countermeasures to break the relation between power consump-

doi:10.6342/NTU201903050

tion and the processing data. To better understand how a countermeasure affects the side-channel characteristics and to gain more confidence in its effectiveness, it is common to utilize statistical tools to quantify the leakages. This process is also known as leakage assessment.

3.2 Leakage Assessment Techniques

Leakage assessment techniques are used to assess whether a device leaks exploitable side-channel information while performing cryptographic operations. There are several different assessment techniques utilize different statistical tools to quantify the leakages, such as the Test Vector Leakage Assessment (TVLA) methodology [11], Signal-to-Noise Ratio (SNR), Normalized Inter-Class Variance (NICV) [6], and Mutual Information Analysis (MIA). These techniques can be useful to both the attackers and the designers of cryptographic systems. For an attacker, performing a standard side-channel attack is often computationally heavy and time-consuming. Leakage assessment tools can help confirm possible leakages and identify the location of the leakage information. By only focusing on the part where the device leaks the secret information, it is useful to lower the cost of an attack. For the designers, leakage assessment techniques can be used to test the effectiveness of different countermeasures and finding the desired trade-off between the side-channel resistance and performance efficiency. These assessment techniques can also provide a certain level of confidence that no detectable information is leaked through the side-channels during the cryptographic operations.

3.2.1 Test Vector Leakage Assessment

The Test Vector Leakage Assessment methodology is a popular leakage detection tool in both academic research and the industry. TVLA uses the Welch t-test to assess whether the changes in the sensitive data can be detected in the measurement data. There are several settings designed to detect different specific types of data leakage. The most common setting is the general test, which tests whether the fixed and random input data cause

doi:10.6342/NTU201903050

differences in the side-channel measurement.

The t-test statistics can be calculated by the following formula:

$$t = \frac{\bar{L}_A - \bar{L}_B}{\sqrt{\frac{S_A^2}{N_A} + \frac{S_B^2}{N_B}}}$$

where $\bar{L}_A$ and $\bar{X}_B$ are the leakage sample means, the S_A and S_B are the standard deviations, and N_A and N_B are the number of measurement in each set. The threshold of ± 4.5 is often set as the pass criteria. If two separate tests exceed the threshold in the same direction, then the implementation is considered leaking side-channel information, and fail the TVLA test.

3.2.2 Signal-to-Noise Ratio

Signal-to-noise ratio is a popular measure used in electrical engineering, that shows the ratio of the desired signal to the unwanted background noise. In power analysis, SNR quantifies the amount of information that is leaking through the side-channel, where the information is the variation in the side-channel signal caused by the different processing data. A high SNR value indicates a large leakage information [15]. The SNR value can be calculated by the following formula:

$$\text{SNR} = \frac{\text{Var}(signal)}{\text{Var}(noise)} = \frac{\text{Var}(\text{E}[\boldsymbol{L}|X])}{\text{E}[\text{Var}(\boldsymbol{L}|X)]}$$

where the leakage traces $\boldsymbol{L}$ are first separated into groups by the intermediate data value. Then, the signal is the variance of the means from each group, and the noise is the variance within the same group.

3.2.3 Normalized Inter-Class Variance

The NICV leakage detection tool was proposed in [6], which has the advantage of relying only on the publicly known information, such as the plaintext or the ciphertext information. The idea of the NICV is to quantify how much of the total variance in the measurement is

doi:10.6342/NTU201903050

contributed by the variation of the input data. The calculation of the NICV value can be done by the following:

$$\text{NICV} = \frac{\text{Var}(\text{E}[\boldsymbol{L}|X])}{\text{Var}(\boldsymbol{L})} = \frac{1}{1 + \frac{1}{\text{SNR}}}$$

where the numerator denotes the variation from the input data, and the denominator denotes the total variation of the data set. If the NICV value reaches 1, it means that all the variation of the measurement is caused by the data-dependent power consumption, which indicates a significant leakage was found in the side-channel measurement. Because of not having an assumption on the power model, it is shown to be an upper bound to the correlation-based leakage detection method and can be directly related to the SNR. One detail of the calculation of NICV is that the number of traces in each group is assumed to be roughly the same, otherwise the computation may result in the NICV value exceeds 1. In the case where the number of traces in each group is highly unbalanced, one can simply calculate the SNR, then transform the result into NICV.

3.3 Template Attack

Among the different kinds of side-channel analysis techniques, template attack, which was first introduced in [7], is believed to be the most powerful form of an attack in an information-theoretic sense. The commonly used Differential-based Power Analysis (DPA) [13] techniques focus on the instantaneous power consumption at a certain point during the cryptographic operation, while in reality, the leakage often lies in several parts of the recorded trace. Template attack, on the other hand, utilizes all the information available in the leakage traces and builds a profile linking the traces to the secret information. As a result, the number of traces required during an attack can be significantly lower than the differential-based power analysis techniques.

It is shown in [15], that the instantaneous power consumption can be modeled as the combination of the operation dependent signal and the random noise. Where we can use a power model to map the processing data to the instantaneous power consumption, and

doi:10.6342/NTU201903050

the noise can be seen as a Gaussian distribution. As a result the instantaneous power consumption can also be modeled as a Gaussian distribution, where the means depends on the processing data and the variance is caused by the noise.

$$\begin{aligned} P_{total} &= P_{signal} + P_{noise} \\ &= PowerModel(data) + \mathcal{N}(0,\, \sigma^2) \\ &= \mathcal{N}(PowerModel(data),\, \sigma^2) \end{aligned}$$

The power trace recorded from a cryptographic device is a time series of instantaneous power consumption samples. In order to describe the correlation between each sample, the multivariate Gaussian model is commonly used to characterize the power trace. The multivariate Gaussian model can be seen as the extension of the Gaussian model, where the mean is now a mean vector, and the variance is now a covariance matrix denoting the correlation between each sample. For example, a power trace consist of 10 sample points, can be modeled as a 10-dimensional multivariate Gaussian model, with a mean vector of size 10 and a covariance matrix of size 10x10.

In the profiling phase, the attacker is assumed to have access to an identical device and have full control over it. For example, being able to change the internal secret information or sending different input data. To build the templates using the multivariate Gaussian model, the following steps are required:

1. Record a large amount of traces, with different input data or secret values.
2. Separate the traces into different groups according to the secret values.
3. Calculate the mean vector $\boldsymbol{t}_k$ and the covariance matrix $\boldsymbol{C}_k$ of each group k.

$$\boldsymbol{t}_k = \frac{1}{n_p} \sum_{i=1}^{n_p} \boldsymbol{t}_{k,i}$$

$$\boldsymbol{C}_k = \frac{1}{n_p - 1} \sum_{i=1}^{n_p} (\boldsymbol{t}_{k,i} - \boldsymbol{t}_k)(\boldsymbol{t}_{k,i} - \boldsymbol{t}_k)^T$$

To estimate the mean vector of the model for each group, it is common to use the average of the traces in the same group. The estimated covariance matrix comes from the

doi:10.6342/NTU201903050

pairwise covariance between each sampling point within the group. The more data are collected to build the model, the more accurate the model is to represent the real distribution of the traces.

In the attack phase, the goal is to infer the secret value being processed by the device. Using the multivariate Gaussian models that are built during the profiling phase, the probability density function of the leakage trace $\boldsymbol{t}$, given the mean and covariance matrix of the secret value k is as follows

$$\Pr(\boldsymbol{t}|\boldsymbol{t}_k, \boldsymbol{C}_k) = \frac{1}{\sqrt{(2\pi)^n|\boldsymbol{C}_k|}} \exp\left(-\frac{1}{2}(\boldsymbol{t}-\boldsymbol{t}_k)^T \boldsymbol{C}_k{}^{-1}(\boldsymbol{t}-\boldsymbol{t}_k)\right).$$

During the attack, the adversary would collect a set of traces $\mathcal{L}_a$ from the target device with the unknown secrets. Then, the secret value is extracted by selecting the model that maximize the product of the posterior probabilities

$$k^* = \underset{k}{\operatorname{argmax}} \prod_{\boldsymbol{t}_i \in \mathcal{L}_a} \frac{\Pr\ (\boldsymbol{t}_i|\boldsymbol{t}_k, \boldsymbol{C}_k) \Pr\ (k)}{\Pr\ (\boldsymbol{t}_i)}.$$

When performing the template attack using the multivariate Gaussian model, several problems can arise during the implementation. First, the required memory of the covariance matrix and the computational time grows quadratically to the number of samples in each trace. As the side-channel leakages usually contain thousands, sometimes millions of samples per trace, it is impractical to build the template based on the raw side-channel trace directly acquired from the device. Second, the number of traces used for profiling is necessary to be larger than the number of samples in each power trace, otherwise, the covariance matrix $\boldsymbol{C}_k$ could become singular [9], and causing problems when calculating the probability density function. Third, as the number of traces and samples grows, the computation may reach the floating-point limits, and increase the error in the templates. To fix these problems, in practice, it is common to reduce the number of samples in the traces either by only selecting some point of interest or apply some data-dimension reduction techniques, such as the Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA).

doi:10.6342/NTU201903050

3.3.1 Point of Interest Selection

The goal of selecting the point of interest is to reduce the number so that building the templates and calculating the probability density function can be practical and efficient.

There are two main reasons why we can build the templates by only using part of the traces. First, during the whole cryptographic operation, usually, there are only a few parts of the operation are related to the secret information. Take the power trace of an 8-bit microprocessor running an AES encryption for example, the information directly related to the secret key can only be found in the first round of the encryption process because of the diffusion property of the permutation layer. When taking a closer look at the power trace during the first round operation, the power consumption related to the byte 0 sub-key will only appear at the beginning of the Add Round Key, Byte Substitution, and the Mix Columns operations. There will not be any information related to the byte 0 sub-key in the Shift Rows operation since the byte 0 does not take part in the operation. Second, the samples that are close together in the power trace usually contain similar information, especially when they are in the same clock cycle.

Based on the above observations, there are several proposals for selecting the point of interest. The most common one is to use the pairwise difference between the average trace of each group [7]. Sometimes an additional condition of not picking the samples within the same clock cycle may be used to reduce the samples.

3.3.2 Principal Component Analysis

Principal Component Analysis is a popular dimension reduction tool, which projects a high-dimensional data into a low-dimensional subspace while preserving the maximum information within the data [4]. The PCA-based template attack first projects the traces onto the lower-dimensional principal subspace, then perform the template attack on the principal subspace.

doi:10.6342/NTU201903050

To find the principal components, we first calculate the empirical covariance matrix.

$$\boldsymbol{S} = \frac{1}{K}\sum_{k=1}^{K}(\boldsymbol{t}_k - \bar{\boldsymbol{t}})(\boldsymbol{t}_k - \bar{\boldsymbol{t}})^T$$

Where $\boldsymbol{t}_k$ is the mean trace for each class, and $\bar{\boldsymbol{t}}$ is the average of the mean traces. The principal directions can then be found by calculating the eigenvectors of the empirical covariance matrix. However, the size of the covariance matrix $\boldsymbol{S}$ is normally quite large, and difficult to compute. It is shown in [4], that the computation can be done with the following alternative method. Let $\boldsymbol{T} = (\boldsymbol{t}_1 - \bar{\boldsymbol{t}}, \boldsymbol{t}_2 - \bar{\boldsymbol{t}}, ..., \boldsymbol{t}_K - \bar{\boldsymbol{t}})$, and the covariance matrix can be rewritten as $\boldsymbol{S} = \frac{1}{K}\boldsymbol{T}\boldsymbol{T}^T$. First, let us perform the eigendecomposition of $\frac{1}{K}\boldsymbol{T}^T\boldsymbol{T}$, and we will have$(\frac{1}{K}\boldsymbol{T}^T\boldsymbol{T})\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Delta}$, where $\boldsymbol{U}$ and $\boldsymbol{\Delta}$ are the eigenvectors and the eigenvalues. Then, left multiply $\boldsymbol{T}$ on both sides, and we now have $\boldsymbol{S}(\boldsymbol{T}\boldsymbol{U}) = (\boldsymbol{T}\boldsymbol{U})\boldsymbol{\Delta}$, where $\boldsymbol{T}\boldsymbol{U}$ is the eigenvectors of $\boldsymbol{S}$. Finally, the projection matrix $\boldsymbol{W}$ is built by selecting the M eigenvectors with the largest eigenvalues. After projecting all the traces onto the principal subspace, we can continue the template attack with only M samples in each trace.

3.3.3 Linear Discriminant Analysis

Similar to the PCA-based template attack, the LDA-based template attack also projects the traces into a lower-dimensional subspace and performs the template attack with a lower number of samples. The objective of the Linear Discriminant Analysis is to find the axes where the ratio of the between-class variance and the within-class variance is maximized:

$$J(w) = \frac{Var(\boldsymbol{w}^T\boldsymbol{T}_{between})}{Var(\boldsymbol{w}^T\boldsymbol{T}_{within})} = \frac{\boldsymbol{w}^T\boldsymbol{S}_B\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{S}_W\boldsymbol{w}}.$$

The $\boldsymbol{S}_B$ and $\boldsymbol{S}_W$ denotes the between-class scatter matrix and within-class scatter matrix. This objective function matches the calculation of the leakage assessment tool SNR, where the between-class variance can be related to the data-dependent signal in the traces, and the within-class variance indicates the noise. In other words, LDA projects the traces onto a lower-dimensional space where the SNR is maximized.

doi:10.6342/NTU201903050

To calculate the LDA directions, the first step is to calculate the between-class scatter matrix $\boldsymbol{S}_B$ and the within-class scatter matrix $\boldsymbol{S}_W$.

$$\boldsymbol{S}_B = \sum_{k=1}^{K} N_k(\boldsymbol{t}_k - \bar{\boldsymbol{t}})(\boldsymbol{t}_k - \bar{\boldsymbol{t}})^T$$

$$\boldsymbol{S}_W = \sum_{k-1}^{K} \sum_{i=1}^{N_k} (\boldsymbol{t}_{k,i} - \boldsymbol{t}_k)(\boldsymbol{t}_{k,i} - \boldsymbol{t}_k)^T$$

Then, the LDA directions can be found by performing the eigendecomposition $(\boldsymbol{S}_W^{-1}\boldsymbol{S}_B)$. Similar to the PCA-based template attack, the last step is to build the projection matrix $\boldsymbol{W}$ by selecting the M eigenvectors with the largest eigenvalues, and project the traces into the subspace with only M samples in each trace.

3.3.4 PCA vs. LDA

In the case of side-channel analysis, where the number of samples in each trace is much larger than the number of classes, PCA is excellent at reducing the number of samples. PCA provides us with a way to systematically lower the number of samples in each trace. It can achieve a small sample size even when the raw data trace contains a large number of samples. However, the PCA-based template attack only takes the between-class differences into account, thus, it may not be optimal since the noise within each class is not considered.

The LDA-based template attack, on the other hand, takes the noise distribution into account and tries to find the projection that maximizes the class separability. The advantage of LDA over PCA can be illustrated in Figure 3.1, where the red LDA direction shows a better separability between the blue group and the orange group. However, the limitation of Linear Discriminant Analysis arises when the sample size is large in the side-channel traces. The within-class scatter matrix $\boldsymbol{S}_W$ becomes singular and not invertible when the number of traces N_k is fewer than the number of time samples in each trace. In other words, LDA is not suitable when a large number of samples are in the measurement traces.

doi:10.6342/NTU201903050

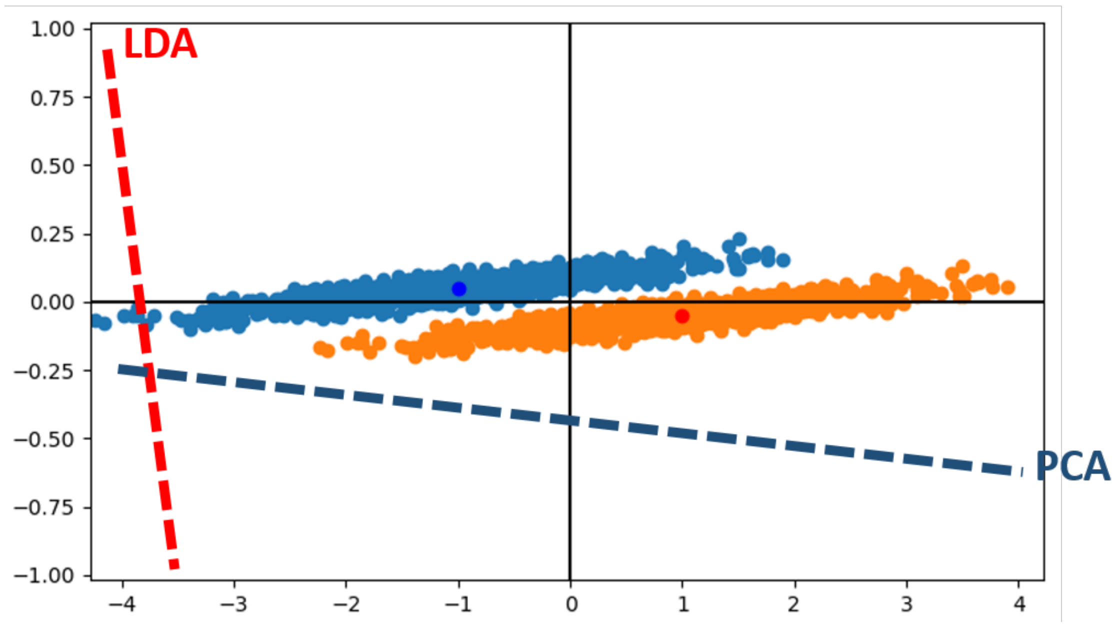


Figure 3.1: PCA vs. LDA

doi:10.6342/NTU201903050

Chapter 4

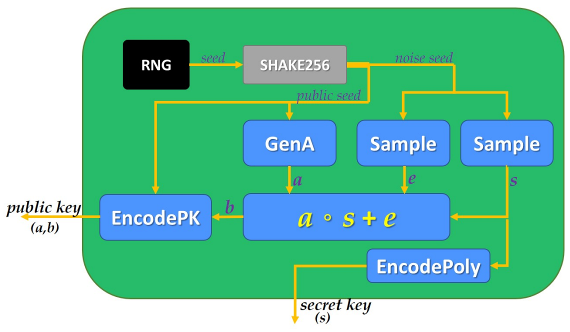
Side-Channel Evaluation on NewHope

4.1 Side-Channel Evaluation on the modules

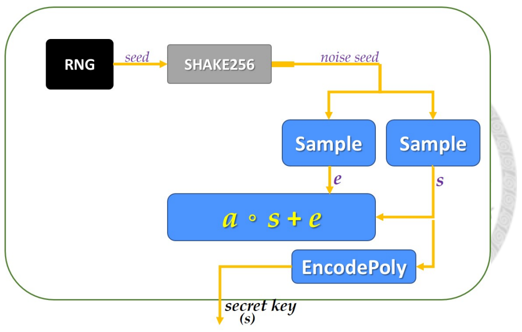
From the aspect of side-channel analysis, while implementing a cryptographic system, it is important to not leak any information related to the secret key or some other intermediate values that could lead to the secrets and compromise the security of the system. For example, in the symmetric block cipher, such as AES, the encryption key is usually considered the information that should be kept a secret, and only known by the authorized party. If the Hamming weight of the first-round S-box output is leaked through the power consumption, which is common in the case of an unprotected AES software implementation, an attacker can easily recover the secret key using Differential Power Analysis.

In the case of public key cryptosystems, or more specifically the NewHope key encapsulation mechanism, the secret information includes both the secret key of NewHope and the 256-bit key material, which could be derived into the shared secret on both sides. Therefore, it is important to implement the functions processing these secrets in a way that doesn't leak information through the side-channels. To further evaluate the design of the NewHope cryptosystem, we now look into the three main functions of the public key encryption scheme, including the key generation, encryption, and decryption process in NewHope, and point out the functions that may be a security risk of the side-channel attacks.

doi:10.6342/NTU201903050

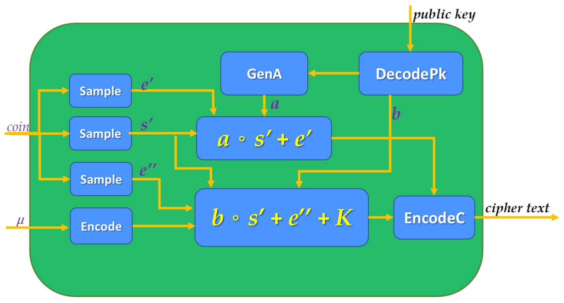


(a) NewHope Key Generation

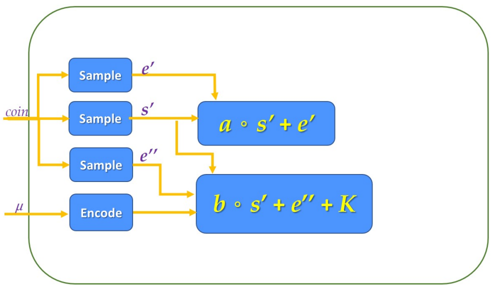


(b) Secrets in Key Generation

Figure 4.1: NewHope Key Generation Side-Channel Evaluation



(a) NewHope Encryption



(b) Secrets in Encryption

Figure 4.2: NewHope Encryption Side-Channel Evaluation

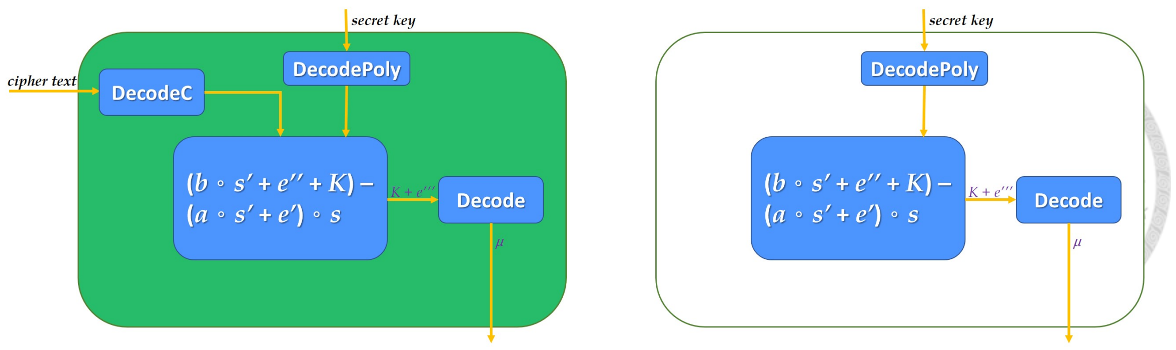
Key Generation

A simplified version of the key generation process is shown in the Figure 4.1(a). If we remove the modules that only process the publicly known information, the remaining modules, shown in Figure 4.1(b), are the ones that should be carefully implemented such that they won't leak side-channel information and compromise the security of the system. The modules are the True Random Number Generator, SHAKE256, Binomial Sampling Function, Polynomial Encoding, and the polynomial arithmetic modules, such as the Number Theoretic Transform (NTT), and polynomial addition and subtraction.

Encryption

A simplified version of the encryption process is shown in Figure 4.2(a), and the modules processing the secret information are shown in Figure 4.2(b). These modules include Message Encoding, Binomial Sampling Function, and polynomial arithmetic modules.

doi:10.6342/NTU201903050


secret key
cipher text
DecodeC
DecodePoly
$(b \circ s' + e'' + K) - (a \circ s' + e') \circ s$
$K + e'''$
Decode
μ
secret key
DecodePoly
$(b \circ s' + e'' + K) - (a \circ s' + e') \circ s$
$K + e'''$
Decode
μ

(a) NewHope Decryption (b) Secrets in Decryption

Figure 4.3: NewHope Decryption Side-Channel Evaluation

Decryption

A simplified version of the decryption in NewHope is shown in Figure 4.3(a), and the modules that are processing the secret information are the polynomial arithmetic modules, and the Message Decoding, shown in Figure 4.3(b).

To summarize, the following modules in NewHope may require more attention and a careful evaluation of their implementation to prevent side-channel attacks: True Random Number Generator, SHAKE256, Binomial Sampling Function, Polynomial Encode, Polynomial Decode, Message Encode, Message Decode, and the polynomial arithmetic modules.

4.2 Related Works

Several works have already discussed the side-channel properties of the lattice-based primitives. Simple power analyses targeting the modular addition operation and the NTT operation was proposed in [19] and [21]. To protect against side-channel attacks, masking schemes on the NTT operation and the other Ring-LWE primitives were introduced in [23] and [16]. While some of the primitives in NewHope have been discussed in the previous work, other subroutines, such as the encoding process and the Binomial Sampling Function, are less explored. Although these operations have already been implemented in a constant-time fashion, the leakage information in the power consumption may still compromise the security of the cryptosystem.

doi:10.6342/NTU201903050

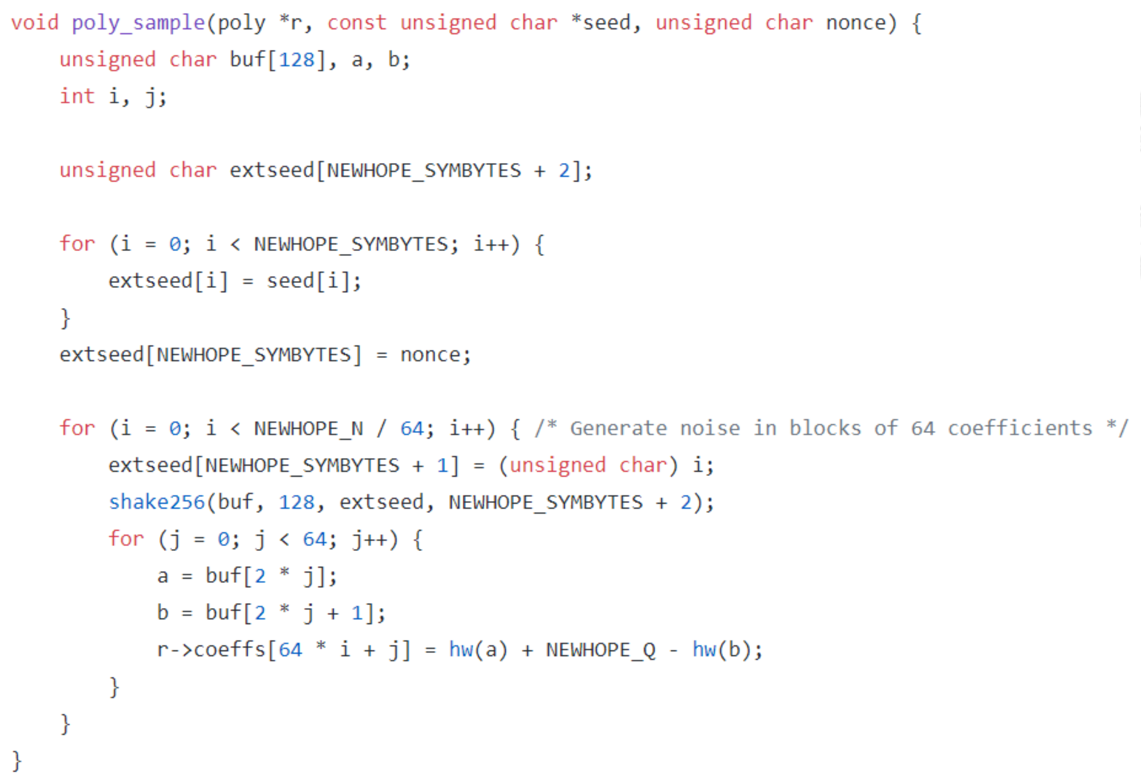
Depending on the threat model and the situation where the NewHope scheme is used, the definition of side-channel vulnerability may vary in different scenarios. For example, in the case of the NewHope-CPA secure mode, the generated key pair is meant to be ephemeral. Therefore, the differential-based side-channel attacks and the masking countermeasures are irrelevant in the scenario. And the reuse of the key pair may lead to other types of attacks [5]. On the other hand, if the attacker is powerful enough to recover the secret information within a single trace, such as mentioned in [21], the masking countermeasure may also fail, since the attacker can simply recover the different shares of the secret information and combine the results.

4.3 Evaluation on the Binomial Sampling Function

Of all the modules processing the secret information, the Binomial Sampling Function is found to be the most important subroutine for several reasons. First, this function is used multiple times in the whole scheme, including the key generation and the encryption process in NewHope. Second, the sampling function, which generates a 'small' polynomial with coefficients that are binomially distributed within the range of $[-8, 8]$, is the sole provider of the confidentiality for the whole encryption scheme. Third, during the key generation process, the secret key is also generated by the Binomial Sampling Function. Therefore, if an attacker is able to recover the output of this function, the NewHope key encapsulation mechanism will then be completely compromised and provide no protection to the secret information at all.

To further evaluate the Binomial Sampling process, we now take a closer look at the implementation of NewHope in the pqm4 crypto library [12]. In Figure 4.4, we can see that the Sampling function first takes in a 32-byte seed and a nonce. Then, it uses SHAKE256 to generate a series of random bytes. In the last step, the binomial distributed random numbers are generated by subtracting the Hamming weight of two random bytes. If an attacker can gather the Hamming weight information of the two random bytes through the side-channel information, then recovering the output of this function would be as simple as subtracting the two Hamming weight values.

doi:10.6342/NTU201903050



```
void poly_sample(poly *r, const unsigned char *seed, unsigned char nonce) {
    unsigned char buf[128], a, b;
    int i, j;

    unsigned char extseed[NEWHOPE_SYMBYTES + 2];

    for (i = 0; i < NEWHOPE_SYMBYTES; i++) {
        extseed[i] = seed[i];
    }
    extseed[NEWHOPE_SYMBYTES] = nonce;

    for (i = 0; i < NEWHOPE_N / 64; i++) { /* Generate noise in blocks of 64 coefficients */
        extseed[NEWHOPE_SYMBYTES + 1] = (unsigned char) i;
        shake256(buf, 128, extseed, NEWHOPE_SYMBYTES + 2);
        for (j = 0; j < 64; j++) {
            a = buf[2 * j];
            b = buf[2 * j + 1];
            r->coeffs[64 * i + j] = hw(a) + NEWHOPE_Q - hw(b);
        }
    }
}
```

Figure 4.4: Implementation of the NewHope Sampling Function in pqm4 crypto library.

Although getting the Hamming weight information of processing data is a commonly used assumption in literature [10], it may not be easy to attain on a real device. First, the leakage model of a device may be very different from the Hamming weight model. Second, the noise in the side-channel measurement may cause errors in the template attack. In the case of the sampling process, we only get a single shot at recovering the information, since the sampling function generates a new set of outputs each time it is used. Also, the success rate of the identification needs to be high enough to ensure the correctness of recovering the full 1024 coefficients in a secret polynomial. To achieve a single trace side-channel attack, it is important to minimize the noise during the acquisition phase and try to maximize the difference in the signal during the template building phase. In our experiment, the synchronized sampling equipment is used to reduce the noise during the acquisition, and the POI selection criteria and the LDA-based template are used to maximize the separability of the side-channel measurements.

doi:10.6342/NTU201903050

Chapter 5

Experiment

5.1 Experiment Setup

In the experiment, the NewHope submitted reference C program and the pqm4 crypto library [12] are used as the target implementation for the side-channel analysis. The pqm4 crypto library contains implementations of the post-quantum cryptographic schemes targeting the ARM Cortex-M4 family of microprocessors. The implementation includes the Cortex-M4 specific optimization using the assembly code. However, the binomial sampling function in NewHope is still implemented in the C language. The target platform in our experiment is the STM32F3 microprocessor, which is also in the ARM Cortex-M4 family.

The ChipWhisperer toolchain is used as the acquisition device, including the Chipwhisperer-Pro hardware capture device and the ChipWhisperer Capture 4.0.2 software program. The ADC in the ChipWhisperer-Pro has a 10-bit resolution, and the analog amplifier is of a maximum 60dB gain. ChipWhisperer is an open-source toolchain for side-channel analysis [18]. One of the main attractions, other than being low-cost equipment, is the synchronized sampling feature, which lowers the samples required during the acquisition and decreases the noise introduced by the phase shift [17]. In this setup, the clock rate of the microprocessor is 7.38 MHz, and the sampling rate is 29.52MS/s, four times the microprocessor's clock frequency.

doi:10.6342/NTU201903050

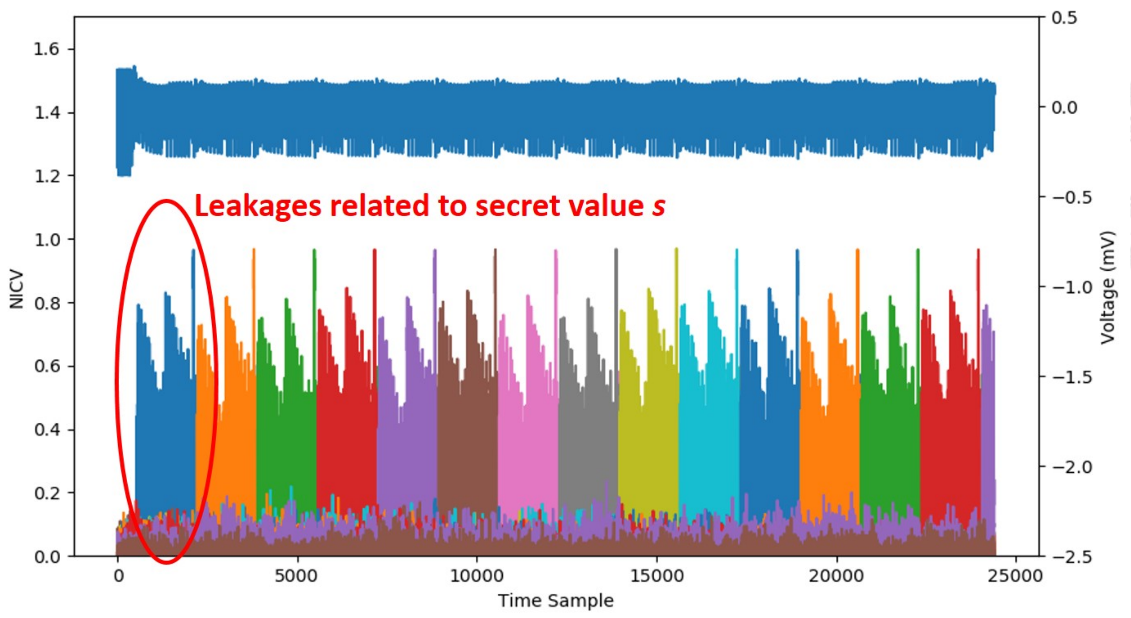


Figure 5.1: The NICV of the Binomial Sampling Function for each secret value s.

5.2 Leakage Assessment on the Binomial Sampling Function

To confirm our suspicion that the Binomial Sampling Function is indeed leaking the information related to its output, we use the NICV leakage detection tool to test the power traces, and the result is shown in Figure 5.1. The power trace is shown in the upper half of the figure with a blue line, and the lower half shows the result of the NICV value for each sampled output s. We can see that the leakage information for each output s can be detected in the repeated pattern in the captured power trace.

If we take a closer look at a single sampling operation, shown in Figure 5.2. The power trace can be roughly separated into three parts. With the help of the NICV leakage assessment, we can confirm that the first and second parts correspond to the calculation of the Hamming weight of the two random bytes. The last part is the operation where the two Hamming weight values are subtracted into the output s. At the end of the sampling operation, the NICV value reaches 9.5, which is high enough to possibly extract the secret with a single trace in our experience.

doi:10.6342/NTU201903050

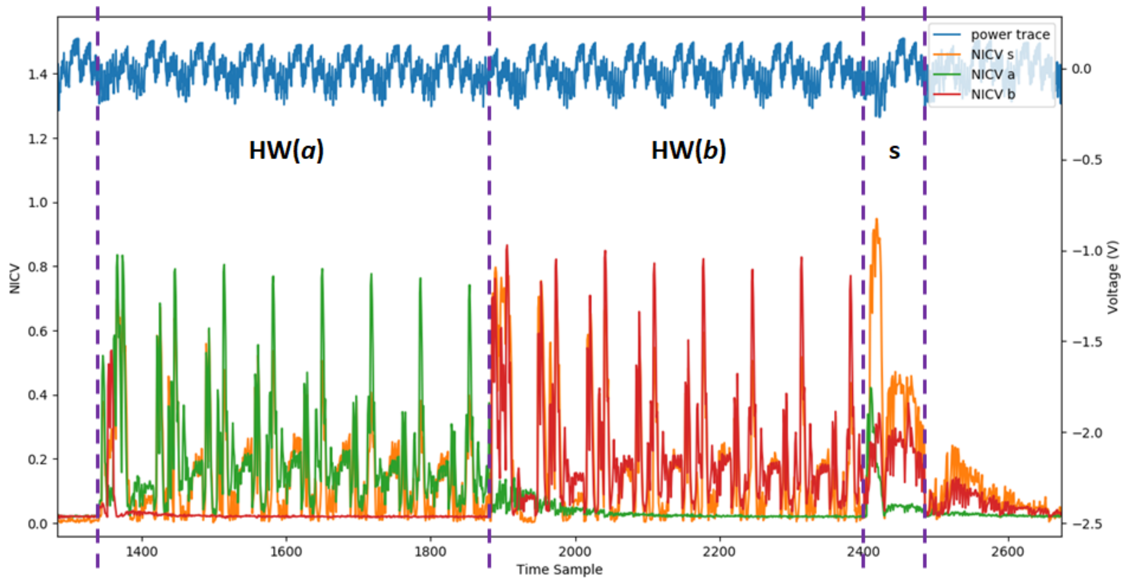


Figure 5.2: NICV of the random bytes a, b, and the output s of a single sampling operation.

5.3 Template Attack on the Binomial Sampling Function

In this section, we are going to present the result of the two types of template attacks, the POI selection and the LDA-based template attack. In the profiling stage, we recorded 25,600 traces of the Binomial Sampling Function with random input seeds, and another 200 traces were recorded for testing. Each trace contains 64 sampling operations, which covers one inner for loop in the Sampling Function. To increase the number of traces for profiling, we separate each trace into 64 single sampling operation. As a result, we have a total number of 1,628,400 traces for profiling.

5.3.1 POI-based Template Attack

In our experiment, it is found that the following selection process yields a better result. First, use the NICV leakage detection technique to find the samples that contain the most information related to the secrets in the power trace. Then, find the correlation coefficients between the selected samples, and remove the ones that are highly related to others. Because the highly correlated samples do not bring much extra information, removing those samples can reduce the computational complexity without losing much information. The thresholds for NICV and the correlation coefficient were set to 0.5 and 0.99, and the selected POIs of the random bytes a and b are shown as the blue dots in Figure 5.3 and

doi:10.6342/NTU201903050

	a	b	$HW(a)$	$HW(b)$	s
Success Rate	96.72%	95.31%	90.43%	81.66%	67.16%

Table 5.1: The success rate of the POI-based template attack of 12800 sampling operations.

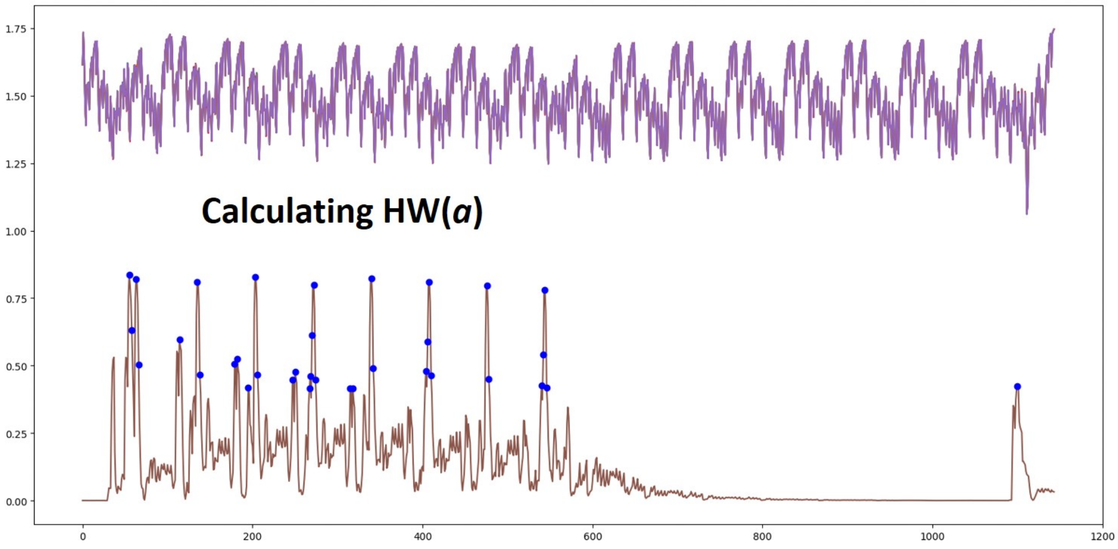


Figure 5.3: The selected POIs of random byte a.

Figure 5.4.

200 traces, each contains 64 sampling operations, are used in the evaluation process. The success rate is calculated by the correct identification of the secret value with a total of 12,800 trials. The result of the POI-based template attack is shown in Table 5.1.

Even though the success rate of identifying the random number a and b is quite high, the 96% success rate is not good enough to threaten the security of NewHope, since there are 1024 coefficients in the secret polynomial $\boldsymbol{s}$.

5.3.2 LDA-Based Template Attack

Figure 5.5 shows the projection of the profiling traces using the first two LDA-directions. Each dot in Figure 5.5 represents a trace in the profiling data set. Traces with different s value were projected onto a different location in the subspace. The 17 groups of dots were well separated in the LDA subspace. And we can observe the binomial distributed s by the fact that the number of dots in the group -8 and 8 are both significantly lower than the others. Figure 5.6(a) and Figure 5.6(b) also shows a highly separated traces by the

doi:10.6342/NTU201903050

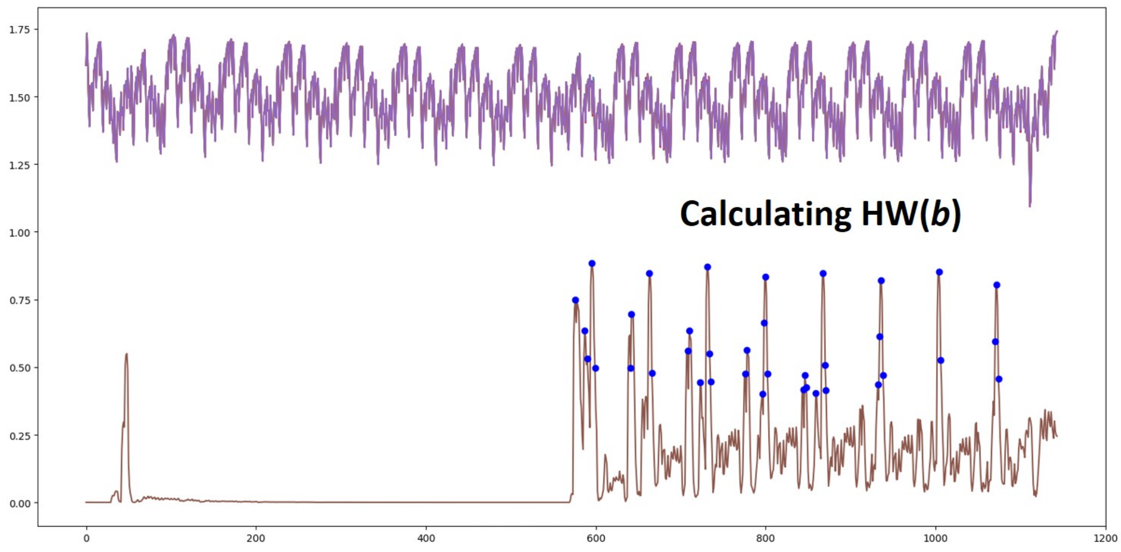


Figure 5.4: The selected POIs of random byte b.

	a	b	$HW(a)$	$HW(b)$	s
Success Rate	100%	100%	100%	100%	99.96%

Table 5.2: The success rate of the LDA-based template attack of 12800 sampling operations.

values of HW(a) and HW(b) in the LDA subspace. The good performance of the LDA-based template attack is shown in Table 5.2, where the attack achieves a 100% success rate with 12,800 trials when extracting the Hamming weight information of the random bytes a and b. The result provides the evidence for our analysis in Section 4.3, that an attacker can recover the Hamming weight information of the two random bytes and calculate the secret information s. With the success rate reaching 100%, there should be no problem recovering the full 1024 coefficients of the secret polynomial $\boldsymbol{s}$.

5.3.3 Discussion

In the experiment, we showed that not only is it possible to extract the Hamming weight information directly from the side-channel information, in the case of Binomial Sampling Function, we can also extract the exact value of the processed data. From the result of the two types of template attacks, we can conclude that the LDA-based template attack performs better than the POI-based method. The main difference between the two approaches is that the POI-based template only utilizes part of the leakage sample while the

doi:10.6342/NTU201903050

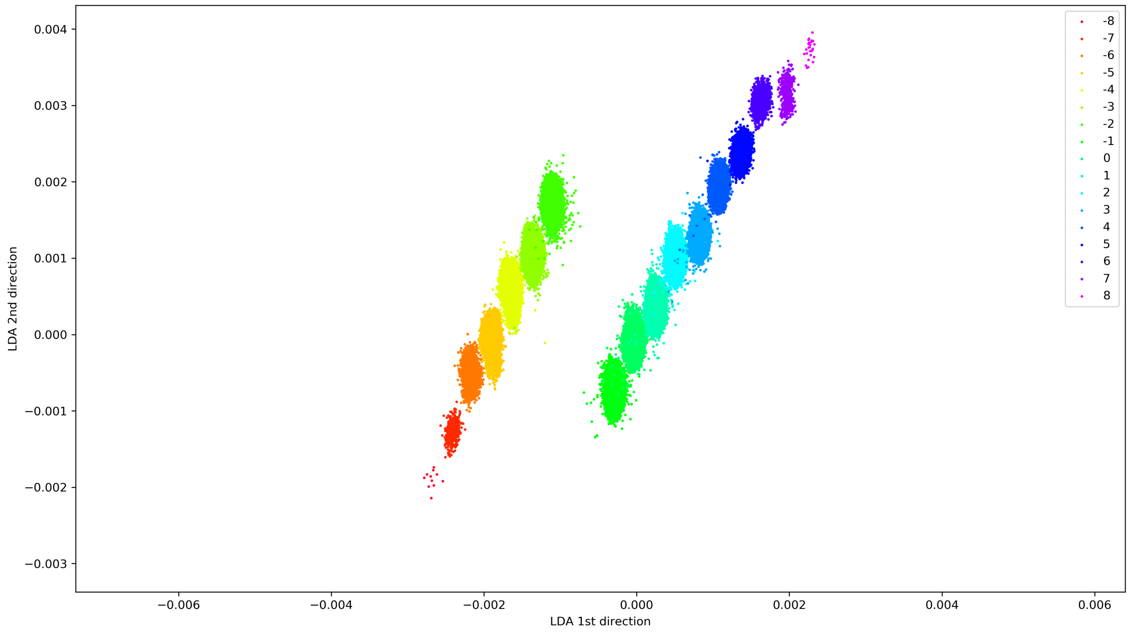
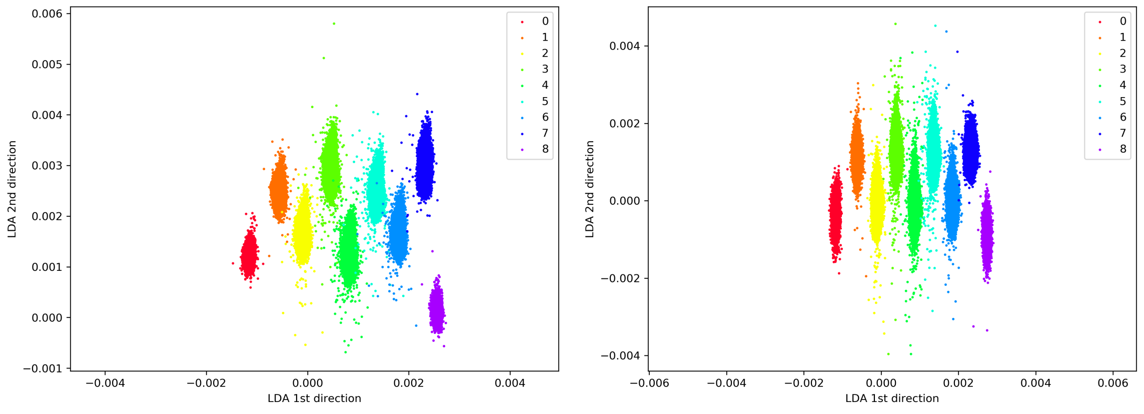


Figure 5.5: The projection of the profiling traces with the first two LDA-direction of s.



(a) Hamming Weight of a (b) Hamming Weight of b

Figure 5.6: The projection of the profiling traces with the first two LDA-directions.

doi:10.6342/NTU201903050

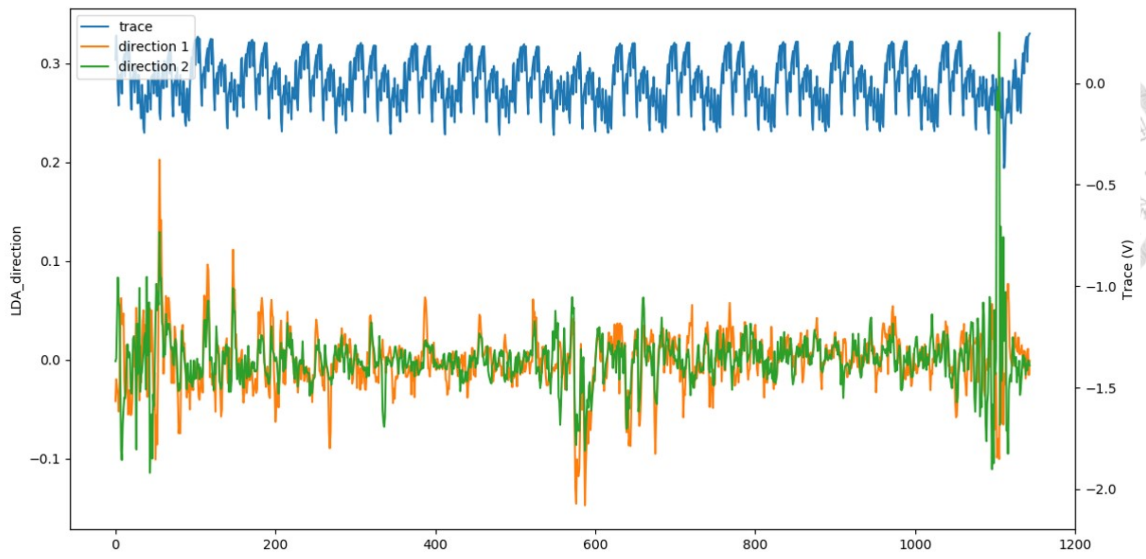


Figure 5.7: The 1st and 2nd LDA-direction of s.

LDA method combines all the available information to maximize the differences between each group.

One interesting observation we made in our experiment is that the projected traces of the value -8 to 8 on the first direction are arranged in order from left to right in Figure 5.5. If we take a look at the first LDA-direction, which was shown as the orange line in Figure 5.7, we can see that there is a positive peak in the beginning of the trace, and a negative peak in the middle of the traces. It is shown in Figure 5.2 that these two locations are the beginning of the two Hamming weight calculations, which makes it highly possible to leak the Hamming weight information of the 2 random bytes a and b. The 'projection' of traces is essentially performing some linear combination of the samples in each trace, such that the traces from the different groups are best separated. What the first LDA direction does is to find the two Hamming weight information in the power trace and subtract the two values to get the maximum separability of the traces, which matches the actual calculation of the secret value s.

5.3.4 Mitigation

A parallel implementation of the Hamming weight function, calculating four Hamming weight at once, can not only speed up the performance but also increase the noise in the

doi:10.6342/NTU201903050

power leakage, which making the ‘single trace’ attack more difficult. However, with a higher resolution oscilloscope and a localized EM measurement, an attacker may still be able to identify the secrets and perform the attack. A masking countermeasure is proposed in [16]. However, if an attacker can identify the two shares with a single trace, the masking scheme may fail as a countermeasure. A better way is to apply the shuffling countermeasure by randomizing the sequence of the sampling operation. The shuffling countermeasure should be secure against the single-trace attack as long as the sequence is unknown to the attacker.

doi:10.6342/NTU201903050

Chapter 6

Concolusion

The side-channel characteristics of the NIST PQC candidates are becoming more and more important since the report form NIST stated that the performance will play more of a role in the later selection process. It is important to investigate whether the implementations are secure against side-channel attacks.

In this work, we analyzed the design of the NewHope cryptosystem, and identified the modules that may be the targets of side-channel analysis. We then showed that it is possible to extract the secret information generated by the Binomial Sampling Function. First, the NICV leakage assessment tool is used to confirm and identify leakages during the sampling operation. Then, we implemented different types of template attacks, and achieve a single trace attack on an ARM Cortex-M4 microprocessor with a 100% success rate. The result shows that the straightforward implementation of the NewHope cryptosystem is vulnerable to side-channel analysis. While this work focuses on the implementation of NewHope, some other Ring-LWE based encryption schemes also use the same method to sample from the binomial distribution. Therefore, the attack may also apply to other Ring-LWE based candidates with similar implementation.

doi:10.6342/NTU201903050

doi:10.6342/NTU201903050

References

[1] G. Alagic, G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, Y.-K. Liu, C. Miller, D. Moody, R. Peralta, et al. *Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process*. US Department of Commerce, National Institute of Standards and Technology, 2019.

[2] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Newhope without reconciliation. *IACR Cryptology ePrint Archive*, 2016:1157, 2016.

[3] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange—a new hope. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 327–343, 2016.

[4] C. Archambeau, E. Peeters, F.-X. Standaert, and J.-J. Quisquater. Template attacks in principal subspaces. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 1–14. Springer, 2006.

[5] A. Bauer, H. Gilbert, G. Renault, and M. Rossi. Assessment of the key-reuse resilience of newhope. In *Cryptographers' Track at the RSA Conference*, pages 272–292. Springer, 2019.

[6] S. Bhasin, J.-L. Danger, S. Guilley, and Z. Najm. Nicv: normalized inter-class variance for detection of side-channel leakage. In *2014 International Symposium on Electromagnetic Compatibility, Tokyo*, pages 310–313. IEEE, 2014.

[7] S. Chari, J. R. Rao, and P. Rohatgi. Template attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 13–28. Springer, 2002.

doi:10.6342/NTU201903050

[8] L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone. *Report on post-quantum cryptography*. US Department of Commerce, National Institute of Standards and Technology, 2016.

[9] O. Choudary and M. G. Kuhn. Efficient template attacks. In *International Conference on Smart Card Research and Advanced Applications*, pages 253–270. Springer, 2013.

[10] C. Clavier, D. Marion, and A. Wurcker. Simple power analysis on aes key expansion revisited. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 279–297. Springer, 2014.

[11] B. J. Gilbert Goodwill, J. Jaffe, P. Rohatgi, et al. A testing methodology for side-channel resistance validation. In *NIST non-invasive attack testing workshop*, volume 7, pages 115–136, 2011.

[12] M. J. Kannwischer, J. Rijneveld, P. Schwabe, and K. Stoffelen. PQM4: Post-quantum crypto library for the ARM Cortex-M4. https://github.com/mupq/pqm4.

[13] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Annual International Cryptology Conference*, pages 388–397. Springer, 1999.

[14] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer, 2010.

[15] S. Mangard, E. Oswald, and T. Popp. *Power analysis attacks: Revealing the secrets of smart cards*, volume 31. Springer Science & Business Media, 2008.

[16] T. Oder, T. Schneider, T. Pöppelmann, and T. Güneysu. Practical cca2-secure and masked ring-lwe implementation. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 142–174, 2018.

doi:10.6342/NTU201903050

[17] C. O' Flynn and Z. Chen. Synchronous sampling and clock recovery of internal oscillators for side channel analysis and fault injection. *Journal of Cryptographic Engineering*, 5(1):53–69, 2015.

[18] C. O' Flynn and Z. D. Chen. Chipwhisperer: An open-source platform for hardware embedded security research. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 243–260. Springer, 2014.

[19] A. Park and D.-G. Han. Chosen ciphertext simple power analysis on software 8-bit implementation of ring-lwe encryption. In *2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST)*, pages 1–6. IEEE, 2016.

[20] T. Poppelmann, E. Alkim, R. Avanzi, J. Bos, L. Ducas, A. de la Piedra, P. Schwabe, and D. Stebila. Newhope. *NIST submissions*, 2017.

[21] R. Primas, P. Pessl, and S. Mangard. Single-trace side-channel attacks on masked lattice-based encryption. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 513–533. Springer, 2017.

[22] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[23] M.-J. O. Saarinen. Arithmetic coding and blinding countermeasures for ring-lwe. *IACR Cryptology ePrint Archive*, 2016:276, 2016.

doi:10.6342/NTU201903050