



PADERBORN  
UNIVERSITY



Faculty of Computer Science,  
Electrical Engineering and Mathematics  
Research Group Codes and Cryptography

# BUFF Transform

Bachelor's Thesis

in Partial Fulfillment of the Requirements for the  
Degree of  
Bachelor of Science

by

MARLENA MÜLLER

submitted to:

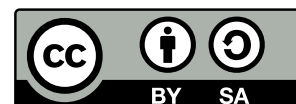
Prof. Dr. Johannes Blömer

and

Prof. Dr. Christian Scheideler

Paderborn, September 12, 2025

This work is licensed under a [Creative Commons](https://creativecommons.org/licenses/by-sa/4.0/)  
“Attribution-ShareAlike 4.0 International” license.





“Cryptography is the ultimate form of non-violent direct action.”

— Julian Assange



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Contents of this Thesis . . . . .	1
<b>2</b>	<b>Definitions and Notations</b>	<b>3</b>
2.1	Asymptotic Notation . . . . .	3
2.1.1	$\mathcal{O}$ Notation . . . . .	3
2.1.2	$\omega$ Notation . . . . .	3
2.1.3	Negligible and Super-Poly Functions . . . . .	3
2.2	Linear Algebra . . . . .	3
2.2.1	Rings and Polynomials . . . . .	4
2.2.2	Matrices and Vectors . . . . .	4
2.2.3	Inner Product and Norm . . . . .	4
2.3	Probability Theory . . . . .	5
2.4	Plain Model and the Random Oracle Model . . . . .	5
2.4.1	Plain Model . . . . .	5
2.4.2	Random Oracle Model (ROM) . . . . .	6
2.4.3	Differences between the Plain Model and the ROM . . . . .	6
2.5	Digital Signature Schemes . . . . .	6
2.6	Hash Functions . . . . .	7
2.6.1	Basic Properties of Secure Hash Functions . . . . .	7
2.6.2	Non-Malleability in Hash Functions . . . . .	8
2.7	Non-Resignability . . . . .	9
2.7.1	Entropy Measures . . . . .	9
2.7.2	Non-Resignability in the Plain Model . . . . .	11
2.7.3	Non-Resignability in the ROM . . . . .	11
2.8	Exclusive Ownership . . . . .	12
<b>3</b>	<b>BUFF- and BUFF-Lite-Transform</b>	<b>15</b>
3.1	BUFF-Lite-Transform . . . . .	15
3.1.1	Security of the BUFF-Lite-Transform . . . . .	16
3.2	BUFF-Transform . . . . .	17
3.2.1	Security of the BUFF-Transform . . . . .	18
3.2.2	Resignability of the BUFF-Transform . . . . .	19
3.3	Salted-BUFF-Transform . . . . .	22
3.3.1	Weak Non-Resignability . . . . .	22
<b>4</b>	<b>Applying the Salted-BUFF-Transform</b>	<b>31</b>
4.1	Motivation for Post-Quantum Signature Schemes . . . . .	31
4.2	Introduction to Lattice Cryptography . . . . .	31
4.2.1	apprCVP based Lattice Cryptography . . . . .	32
4.2.2	LWE based Lattice Cryptography . . . . .	35

4.3	FALCON	36
4.3.1	GPV Framework	36
4.3.2	Simplified FALCON Signature Scheme	37
4.3.3	FALCON-PS-3	38
4.4	CRYSTALS-DILITHIUM	42
4.4.1	DILITHIUM and Non-Resignability	43
<b>5</b>	<b>Conclusion</b>	<b>45</b>
5.1	Future Work	45
	<b>Bibliography</b>	<b>47</b>

# 1 Introduction

Cryptography has always been more than a technical discipline; it is a deeply political technology. From its earliest uses in securing diplomatic correspondence to its modern role in protecting digital communication, cryptography has been at the center of struggles over power, secrecy, and control. The significant role of cryptanalysis during the Second World War is a stark reminder of its importance: the breaking of the German Enigma machine by Allied efforts not only shortened the war but also demonstrated how access to secure or insecure communication could alter the course of history ([HS93]).

In the following decades, cryptography increasingly became a matter of state regulation. During the Cold War, encryption algorithms were classified as military technology in the United States and strong cryptographic tools were subject to strict export restrictions ([DL01]). The ability to use and distribute secure communication was thus not only a technical but also a political question, shaped by fears of adversaries gaining access to strong encryption.

Today, similar tensions continue to exist in new forms. Modern debates around legislation such as the proposed EU “Chat Control” initiative highlights how governments still seek ways to limit or weaken cryptographic protections in the name of security and law enforcement. Such measures risk undermining the very foundations of private and secure communication in the digital age. Secure communication is not a luxury—it is a human right ([ECH24]). It enables freedom of speech, privacy, and the protection of sensitive personal and professional information in a world where digital interactions are omnipresent ([Sno15]).

Cryptography lies at the heart of this right. By ensuring confidentiality, authenticity, and integrity, cryptographic mechanisms enable citizens to protect their digital lives against surveillance, manipulation, and abuse. The development of advanced cryptographic primitives, is part of the ongoing effort to strengthen these guarantees. Therefore research in this field is not only of technical relevance but also carries significant societal and political weight, as it contributes to safeguarding the basic freedoms upon which democratic societies depend on.

## 1.1 Contents of this Thesis

The overarching goal of this thesis is to study the BUFF-Transform and its variants as methods for strengthening the security of digital signature schemes. While signature schemes traditionally aim at providing existential unforgeability, research has shown the necessity of additional properties such as non-resignability and exclusive ownership. The analysis of the BUFF-Transform, presented in this thesis, is a processing and synthesis of the current state of research. This thesis examines to what extent the BUFF-Transform and its modifications can provide these stronger guarantees and how they can be applied to the post-quantum signature schemes FALCON and CRYSTALS-DILITHIUM.

The thesis is structured as follows:

- **Chapter 2: Definitions and Notations**

This chapter introduces the mathematical preliminaries and cryptographic concepts that will be used throughout the thesis. It covers asymptotic notation, basic tools from linear algebra, and the formal models of security analysis, including the plain model and the random oracle model. Furthermore, it provides definitions for digital signature schemes, cryptographic hash functions, non-resignability, and exclusive ownership, establishing the theoretical foundation for the subsequent analysis.

- **Chapter 3: BUFF- and BUFF-Lite-Transform**

This chapter presents the BUFF-Transform and the BUFF-Lite-Transform, two constructions that extend classical signature schemes with additional security properties. Their security is formally analyzed, with particular attention to the claims of non-resignability. Furthermore, the salted-BUFF-Transform is introduced as a modification designed to overcome weaknesses regarding non-Resignability of the original BUFF-Transform.

- **Chapter 4: Applying the Salted-BUFF-Transform**

This chapter investigates the application of the salted-BUFF-Transform in the context of post-quantum cryptography. After motivating the need for quantum-resistant signature schemes, it introduces lattice-based cryptography and examines two leading candidates from the NIST standardization process: FALCON and CRYSTALS-DILITHIUM. The analysis demonstrates how the idea of salting the BUFF-Transform interacts with these schemes and evaluates its effect on their security properties.

- **Chapter 5: Conclusion**

In this chapter we will sum up our findings of this thesis and discuss what further topics should be studied in the context of the BUFF-Transform and its derivatives.

## 2 Definitions and Notations

This chapter lists the notations and definitions that will be used throughout this thesis. They are introduced here to keep the presentation consistent and to make later sections easier to follow.

### 2.1 Asymptotic Notation

Asymptotic notation describes the growth of functions as the input size  $n$  tends toward infinity, allowing us to abstract away constant factors and lower-order terms. It is central in algorithm analysis and complexity theory.

#### 2.1.1 $\mathcal{O}$ Notation

The notation  $f(n) \in \mathcal{O}(g(n))$  (read “ $f$  is big- $\mathcal{O}$  of  $g$ ”) means that  $f$  grows *at most* as fast as  $g$ , up to a constant factor, for sufficiently large  $n$ . Formally:

$$\exists c > 0, \exists n_0 \text{ s.t. } \forall n \geq n_0 : |f(n)| \leq c \cdot g(n).$$

A function  $f$  is poly-bounded, if  $f(n) \in \mathcal{O}(n^c)$  for some constant  $c \in \mathbb{N}^+$ .

#### 2.1.2 $\omega$ Notation

The notation  $f(n) \in \omega(g(n))$  means that  $f$  grows faster as  $g$ , up to a constant factor, for sufficiently large  $n$ . Formally:

$$\exists c > 0, \exists n_0 \text{ s.t. } \forall n \geq n_0 : f(n) > c \cdot g(n).$$

#### 2.1.3 Negligible and Super-Poly Functions

In cryptography, a function  $\mu(n)$  is negligible if it decreases faster than the inverse of any polynomial. Formally

$$\mu(n) \text{ is negligible} \iff \forall c > 0 : \lim_{n \rightarrow \infty} \mu(n)n^c = 0.$$

Negligible functions quantify security guarantees: an adversary’s success probability should be negligible in the security parameter. We say  $f(n) \in \text{negl}(n)$  if  $f$  is negligible.

In contrast a function  $f$  is super-poly, if  $\frac{1}{f}$  is negligible. As the name suggests, this definition is the same as:  $f$  grows faster than any polynomial function or formally:  $f(n) \notin \mathcal{O}(n^c)$  for all  $c \in \mathbb{N}^+$ .

## 2.2 Linear Algebra

Linear algebra provides the fundamental tools to study algebraic structures, vector spaces and transformations between them. In this section, we briefly introduce only

some of the most important notations and concepts used throughout this thesis.

### 2.2.1 Rings and Polynomials

A ring  $(R, +, \cdot)$  is an algebraic structure consisting of a set  $R$  together with two binary operations, addition and multiplication, satisfying associativity, distributivity, and the existence of an additive identity  $0 \in R$ . An important example is the ring of polynomials  $R[X]$  with coefficients in a ring  $R$ . In this setting, computations can also be carried out modulo an ideal, e.g. working with polynomials modulo  $X^n - 1$ .

### 2.2.2 Matrices and Vectors

For  $n \in \mathbb{N}^+$ , we denote by

$$1_n = \underbrace{\left( \begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{array} \right)}_{n \text{ columns}} \left. \vphantom{\begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{array}} \right\} n \text{ rows}$$

the  $n \times n$  identity matrix, and by

$$0_n = \underbrace{\left( \begin{array}{cccc} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{array} \right)}_{n \text{ columns}} \left. \vphantom{\begin{array}{cccc} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{array}} \right\} n \text{ rows}$$

the  $n \times n$  zero matrix.

A vector is a column matrix of size  $n \times 1$  or a row matrix of size  $1 \times n$ . Given a matrix  $A \in \mathbb{R}^{m \times n}$ , its transpose is denoted by  $A^T \in \mathbb{R}^{n \times m}$ .

### 2.2.3 Inner Product and Norm

For  $x, y \in \mathbb{R}^n$ , the inner product is defined as

$$\langle x, y \rangle = \sum_{i=1}^n x_i y_i.$$

Most notably, if  $\langle x, y \rangle = 0$  this means  $x$  and  $y$  are orthogonal. The inner product induces the Euclidean norm

$$\|x\|_2 = \sqrt{\langle x, x \rangle}.$$

## 2.3 Probability Theory

Let  $\Omega$  be a finite sample space and let  $\Pr$  be a probability measure assigning to each outcome  $\omega \in \Omega$  a number  $\Pr[\omega]$  such that

$$\Pr[\omega] \geq 0 \quad \text{for all } \omega \in \Omega, \quad \sum_{\omega \in \Omega} \Pr[\omega] = 1.$$

For any event  $A \subseteq \Omega$ , the probability of  $A$  is

$$\Pr[A] = \sum_{\omega \in A} \Pr[\omega].$$

We use  $\bar{A}$  to denote the event ‘ $A$  does not occur’, or formally  $\bar{A} = \Omega \setminus A$ .

**Definition 2.1** (Conditional Probability) *For two events  $A, B \subseteq \Omega$  with  $\Pr[B] > 0$ , the conditional probability of  $A$  given  $B$  is defined as*

$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]}.$$

**Definition 2.2** (Expected Value) *Let  $X : \Omega \rightarrow \mathbb{R}$  be a discrete random variable. The expected value of  $X$  is*

$$\mathbb{E}[X] = \sum_{\omega \in \Omega} X(\omega) \Pr(\omega).$$

Equivalently we can define the expected value as follows: If  $X$  takes values  $x_1, x_2, \dots$  with probabilities  $\Pr[X = x_i]$ , then

$$\mathbb{E}[X] = \sum_i x_i \Pr[X = x_i].$$

**Lemma 2.3** (Difference Lemma) *Let  $Z, W_0, W_1$  be events defined over some probabilistic space and let  $\bar{Z}$  denote the complement of the event  $Z$ . Suppose that  $W_0 \wedge \bar{Z}$  occurs if and only if  $W_1 \wedge \bar{Z}$  occurs. Then we have*

$$|\Pr[W_0] - \Pr[W_1]| \leq \Pr[Z].$$

## 2.4 Plain Model and the Random Oracle Model

In provable security, the model defines the assumptions under which a cryptographic scheme is analyzed. The two models, which are mostly used are the plain model and the Random Oracle Model (ROM). We will now discuss the key aspects of these models as presented in [KL07].

### 2.4.1 Plain Model

The plain model (also called the standard model) assumes that all parties only have access to their own inputs and any public parameters generated during setup. Security proofs in the plain model rely solely on well-established hardness assumptions (e.g., factoring, discrete logarithm, lattice problems), without granting access to idealized oracles. While proofs in this model are considered stronger, they are often more difficult to construct.

### 2.4.2 Random Oracle Model (ROM)

The random oracle model augments the plain model by providing all parties (including adversaries) with access to a publicly available random oracle, i.e. a black-box function, that returns truly random outputs for each new input, and consistently returns the same output for repeated queries. In practice, the random oracle is typically used to model cryptographic hash functions.

### 2.4.3 Differences between the Plain Model and the ROM

When proving the security of a cryptographic scheme, the choice between the plain model and the random oracle model (ROM) has important consequences for both, the interpretation of results and their applicability in practice.

A proof in the plain model is considered stronger, because it does not rely on idealized components. Security in the plain model automatically implies security in the ROM (under the same computational assumptions), but as [CGH04] has shown the reverse is not true: a scheme may be secure in the ROM and insecure when the oracle is instantiated with any real-world function.

## 2.5 Digital Signature Schemes

In this section we will define Digital Signature Schemes and their security. The definitions are based on [BS23].

**Definition 2.4** (Digital Signature Scheme) *A signature scheme  $S = (KGen, Sign, Vrfy)$  is a triple of three efficient algorithms defined over a message space  $\mathcal{M}$  and a signature space  $\mathcal{S}$ .*

- $KGen(1^\lambda) \rightarrow (pk, sk)$ : *The key generation algorithm takes a security parameter  $\lambda \in \mathbb{N}^+$  and outputs a public key  $pk$  and a secret key  $sk$ .*
- $Sign(sk, m) \rightarrow \sigma$ : *The signing algorithm takes a secret key  $sk$  and a message  $m \in \mathcal{M}$  as input and outputs  $\sigma \in \mathcal{S}$ .*
- $Vrfy(pk, m, \sigma) \rightarrow \{\text{accept}, \text{reject}\}$ : *The verification algorithm takes a public key  $pk$ , a message  $m$  and a signature  $Sign$  and outputs either *accept* or *reject*.*

We require that for all  $\lambda \in \mathbb{N}^+$  and for all possible outputs  $(pk, sk)$  of  $KGen(1^\lambda)$  and all messages  $m \in \mathcal{M}$ :

$$\Pr[Vrfy(pk, m, Sign(sk, m)) = \text{accept}] = 1.$$

Using this definition, we can now define the security of Digital Signature Schemes. Intuitively a signature scheme is considered secure, if an attacker, who can query for signatures for chosen messages, cannot (efficiently) produce a valid signature under the same key-pair for a new message. Formally we define this property using a game for modeling the attack and say, a scheme is secure if for any attacker the probability of winning (i.e. forging a valid signature for a new message) is negligible.

**Definition 2.5** (EUF-CMA Security Game) *For a signature scheme  $S = (KGen, Sign, Vrfy)$  we define the Existential Unforgeability under Chosen Message Attack (EUF-CMA) game between a challenger  $\mathcal{C}$  and an adversary  $\mathcal{A}$  as follows:*

- The challenger  $\mathcal{C}$  runs  $KGen(1^\lambda)$  to generate a public key  $pk$  and a private key  $sk$ . The public key  $pk$  is given to the adversary  $\mathcal{A}$ .
- The adversary can query  $\mathcal{C}$ ,  $q \in \mathbb{N}^+$  times for signatures of messages  $m_1, m_2, \dots, m_q \in \mathcal{M}$  and receives the signatures  $\sigma_1, \sigma_2, \dots, \sigma_q$  under corresponding secret key  $sk$ .
- $\mathcal{A}$  outputs a pair  $(m, \sigma)$ .

We say  $\mathcal{A}$  wins in this game, if  $Vrfy(pk, m, \sigma) = \text{accept}$  and  $m \notin \{m_1, m_2, \dots, m_q\}$  (i.e.,  $m$  is a new message).

**Definition 2.6** (EUF-CMA Security) We define the advantage in the EUF-CMA security game for a signature scheme  $S$  and an adversary  $\mathcal{A}$  as follows:

$$\text{Adv}_S^{\text{EUF-CMA}}(\lambda, \mathcal{A}) = \Pr[\mathcal{A} \text{ wins}].$$

We say that  $S$  provides EUF-CMA security if for every probabilistic polynomial time adversary  $\mathcal{A}$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$

$$\text{Adv}_S^{\text{EUF-CMA}}(\lambda, \mathcal{A}) \leq \mu(\lambda).$$

## 2.6 Hash Functions

Hash functions are functions, which map values from a large, possibly infinitely large, input space into a small, finitely sized, output space. Hash functions are used in a variety of ways in computer science, e.g. for calculating checksums, in data structures.

In cryptography we use hash functions for building many cryptographic schemes, like encryption schemes, message authentication codes or signature schemes. For all these applications we require, that finding either a collision (i.e. two different inputs have the same output) or a pre-image is hard.

While in reality we use concrete hash functions, for cryptanalysis of hash functions, we need to use so called keyed hash functions for a theoretical analysis of hash functions. When doing a theoretical analysis of a non keyed hash function, we would get, that hash functions are not secure, as an attacker with a precomputed collision or pre-image would break the security efficiently. Keyed hash-functions circumvent this, by behaving differently for different keys, so collisions or pre-images cannot be precomputed anymore.

### 2.6.1 Basic Properties of Secure Hash Functions

The definitions of secure hash functions we will use are based on [BS23].

**Definition 2.7** (Hash Function) A keyed hash function  $H$  over a key space  $\mathcal{K}$ , a message space  $\mathcal{M}$  and a digest space  $\mathcal{D}$  is a deterministic algorithm that takes  $(k, m) \in \mathcal{K} \times \mathcal{M}$  as input and outputs a digest  $d \in \mathcal{D}$ .

Though it is not required by this definition, in reality the message space is much larger than the digest space.  $\mathcal{M}$  is often defined as all possible bit strings.

**Definition 2.8** (Collision Resistance) For a hash function  $H$  we define the collision resistance game between a challenger  $\mathcal{C}$  and an adversary  $\mathcal{A}$  as follows:

- The challenger randomly chooses  $k \in \mathcal{K}$  and sends it to  $\mathcal{A}$ .

- $\mathcal{A}$  outputs two messages  $m_0, m_1 \in \mathcal{M}$ .

The adversary wins the game, if  $H(k, m_0) = H(k, m_1)$  and  $m_0 \neq m_1$ . We define the advantage  $\mathbf{Adv}_H^{\text{CR}}(\mathcal{A}) = \Pr[\mathcal{A} \text{ wins}]$ .

A hash function  $H$  is collision resistant, if for every probabilistic polynomial time adversary  $\mathcal{A}$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$  it holds that

$$\Pr[\mathbf{Adv}_H^{\text{CR}}(\mathcal{A})] \leq \mu(\lambda)$$

**Definition 2.9** (Pre-image Resistance) For a hash function  $H$  we define the pre-image resistance game between an efficient adversary  $\mathcal{A}$  and a challenger  $\mathcal{C}$  as follows:

- The challenger chooses a random  $m \in \mathcal{M}$  and  $k \in \mathcal{K}$  and sends  $k, H(k, m)$  to  $\mathcal{A}$ .
- The adversary outputs a message  $m'$ .

The adversary wins this game, if  $H(k, m) = H(k, m')$  holds. We define the advantage  $\mathbf{Adv}_H^{\text{PIR}}(\mathcal{A}) = \Pr[\mathcal{A} \text{ wins}]$ .

A hash function  $H$  is pre-image resistant, if for every probabilistic polynomial time adversary  $\mathcal{A}$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$  it holds that

$$\Pr[\mathbf{Adv}_H^{\text{PIR}}(\mathcal{A})] \leq \mu(\lambda).$$

**Definition 2.10** (Second Pre-image Resistance Game) For a hash function  $H$  we define the second pre-image resistance game between an efficient adversary  $\mathcal{A}$  and a challenger  $\mathcal{C}$  as follows:

- The challenger chooses a random  $m \in \mathcal{M}$  and  $k \in \mathcal{K}$  and sends  $k, m$  to  $\mathcal{A}$ .
- The adversary outputs a message  $m'$ .

The adversary wins this game, if  $H(k, m) = H(k, m')$  and  $m \neq m'$ . We define the advantage  $\mathbf{Adv}_H^{2\text{PIR}}(\mathcal{A}) = \Pr[\mathcal{A} \text{ wins}]$ .

A hash function  $H$  is second pre-image resistant, if for every probabilistic polynomial time adversary  $\mathcal{A}$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$  it holds that

$$\Pr[\mathbf{Adv}_H^{2\text{PIR}}(\mathcal{A})] \leq \mu(\lambda).$$

Collision resistance implies second pre-image resistance, which follows intuitively, as an adversary for the second pre-image can easily be transformed into an adversary for the collision resistance game.

**Definition 2.11** (Secure Hash Function) A hash function is considered secure if it is collision resistant and pre-image resistant.

## 2.6.2 Non-Malleability in Hash Functions

For some cryptographic primitives, building upon hash functions, we need additional properties. One of these properties is the so called  $\Phi$ -non-malleability, which was introduced in [BCFW09]. The goal of  $\Phi$ -non-malleability is to formalize a stronger variant of the pre-image resistance, where it is not required to find any pre-image, but a related pre-image.

To formally define the relation, a class of admissible transformations is provided to the adversary, where it can choose a function that describes some arbitrary relation. The following definition is based on [CDF<sup>+</sup>21].

**Definition 2.12** ( $\Phi$ -non-malleability) *For a hash function  $H$  and a class of admissible transformations  $\Phi$  we define the  $\Phi$ -non-malleability game between a challenger and an adversary  $\mathcal{A}$  as follows:*

1. *The challenger generates a hash function key  $k \leftarrow KGen(1^\lambda)$  and sends it to  $\mathcal{A}$ .*
2.  *$\mathcal{A}$  generates a distribution  $\mathcal{X}$  and sends it to the challenger.*
3. *The challenger samples  $x \leftarrow \mathcal{X}$ , calculates  $y \leftarrow H(k, x)$  and sends  $y$  to  $\mathcal{A}$ .*
4.  *$\mathcal{A}$  outputs  $(y', \phi)$ , where  $y'$  is a hash digest and  $\phi \in \Phi$ .*

*The adversary  $\mathcal{A}$  wins, if  $H(k, \phi(x)) = y'$  and  $\phi(x) \neq x$ . We define the advantage as follows:*

$$\mathbf{Adv}_H^{\Phi NM}(\lambda, \mathcal{A}) = \Pr[\mathcal{A} \text{ wins}].$$

*A hash function  $H$  is  $\Phi$ -non-malleable, if for every probabilistic polynomial time adversary  $\mathcal{A}$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$  it holds that*

$$\mathbf{Adv}_H^{\Phi NM}(\lambda, \mathcal{A}) \leq \mu(\lambda).$$

In practical scenarios, this property is useful: an attempt, to attack a hash-based cryptographic system, designed to ensure authenticity by supplying an arbitrary random message, is unlikely to succeed. A human recipient would recognize the message as nonsense, while a computer program would fail to parse it, as it would almost certainly not conform to the expected syntax. However, if the messages used in the attack are related, for example, changing a **Yes** to a **No**, the change might not raise suspicion.

## 2.7 Non-Resignability

In this section we will introduce non-resignability as a security notion for signature schemes.

The basic idea of non-resignability is that an attacker, given a public key  $pk$ , a corresponding signature  $\sigma$  for some message  $m$  and some additional, but restricted, information about  $m$ , cannot efficiently produce a forged signature  $\bar{\sigma}$  and a from  $pk$  different public key  $\bar{pk}$  such that  $\mathbf{Vrfy}(\bar{pk}, m, \bar{\sigma}) = \mathbf{accept}$  with non negligible probability.

### 2.7.1 Entropy Measures

For introducing non-resignability we need to define ways to restrict the information the attacker gets about  $m$ . If the attacker can guess  $m$  with high probability, a feasible attack against non-resignability would be, to generate a new key-pair and sign the guessed  $m$  with this new key-pair. This would directly lead to no signature scheme being able to provide non-resignability.

For restricting the information the attacker will get, we will use entropy measures and require a minimal entropy of the message conditioned on the additional information.

Shannon introduced the idea of entropy measures in [Sha48]. His definition was later called Shannon Entropy and is defined as follows:

$$H(X) := \sum_{x \in X} p(x) \log p(x)$$

where  $X$  is a discrete random variable and  $p : X \rightarrow [0, 1]$  is its distribution. When using this definition with an log base of 2, we have that  $H(X)$  describes the average number of bits required to describe a value of  $X$ .

We will not use this entropy measure for our definition, as other entropy measures will yield much stronger definitions.

### Min-Entropy

Among alternative entropy notions, statistical min-entropy is of interest in our context. Unlike Shannon entropy, which reflects the average uncertainty, min-entropy captures the worst-case scenario by focusing solely on the probability of the most likely outcome. This makes it well-suited for conservative security guarantees, where even a single highly probable guess could enable an attacker to break the scheme. Our definitions are based on [KRS09] and [DFHS24].

**Definition 2.13** (Guessing Probability) *The guessing probability  $guess$  for two discrete random variables  $X$  and  $Y$  is defined as follows:*

$$guess(X | Y) := \left( \sum_{z \in Z} Pr[Y = z] \cdot \max\{Pr[X = x | Y = y] \mid x \in X\} \right)$$

Informally  $guess(X | Z)$  is the probability to guess  $X$  correctly, knowing  $Y$ .

**Definition 2.14** (Statistical Min-Entropy) *The min-entropy  $H_\infty$  for a discrete random variable  $X$  is defined as follows:*

$$H_\infty(X) := \log \frac{1}{\max\{Pr(X = x) \mid x \in X\}}.$$

*The conditional min-entropy for two discrete random variables  $X$  and  $Y$  is defined as follows:*

$$H_\infty(X | Y) := -\log(guess(X | Y)).$$

By definition  $H_\infty(X) \leq H(X)$  holds for all discrete random variables  $X$ .

### Computational HILL-Entropy

In contrast to the statistical min-entropy  $H_\infty$  we can also define a computational counterpart. While there are different computational entropy measures, we will only use the HILL-entropy. The HILL-entropy was introduced in [HILL99]. The definitions we will give are based on [BSW03].

To define the HILL-entropy, we will first need to define computational distance of distributions.

**Definition 2.15** (Computational Distance of Distributions) *Let  $C$  be some class of efficiently computable functions and  $X, Y$  some distributions. The computational distance  $\delta_C$  of the distributions is defined as follows*

$$\delta_C(X, Y) = \max\{Pr[c(X) = 1] - Pr[c(Y) = 1] \mid c \in C\}.$$

The canonical choice for  $C$  is the set of polynomial sized circuits. We will assume for the rest of this thesis, that  $C$  is always a polynomial circuit, if not defined otherwise.

Using this, we can now define the HILL-entropy:

**Definition 2.16** (HILL-Entropy) *Let  $X$  be a random variable. The HILL-entropy is defined as follows:*

$$\varepsilon \text{HILL}_\infty(X) = \max\{H_\infty(Y) \mid \delta_C(X, Y) \leq \varepsilon \wedge Y \text{ is distribution}\}.$$

The conditional HILL-entropy for two random variables  $X, Y$  is defined as follows:

$$\varepsilon \text{HILL}_\infty(X|Z) = \max\{H_\infty(Y|Z) \mid \delta_C((X, Z), (Y, Z)) \leq \varepsilon\}.$$

Here  $(X, Z)$  and  $(Y, Z)$  denote the joint distributions of the respective random variables.

When using the HILL-entropy, the  $\varepsilon$  can be omitted, if it suffices, that  $\varepsilon$  is some negligible value.

### 2.7.2 Non-Resignability in the Plain Model

After introducing the different entropy measures, we can now define non-resignability.

**Definition 2.17** (Non-Resignability-Game (Plain Model)) *Given a signature scheme  $S = (KGen, Sign, Vrfy)$  and an explicit hint function  $\mathbf{aux}$  we define the security game  $NR_S$  for an adversary  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$  as follows:*

```

    (sk, pk) ← KGen(1λ)
    m ← A0(pk)
    h := aux(m, pk)
    σ ← Sign(sk, m)
    (σ̄, p̄k) ← A1(pk, σ, h)
    v := Vrfy(p̄k, m, σ̄)
    return (v = 1 ∧ p̄k ≠ pk)
    
```

**Definition 2.18** (Non-Resignability (Plain Model)) *A signature scheme  $S = (KGen, Sign, Vrfy)$  is called non-resignable in the plain model if for any probabilistic efficient adversary  $\mathcal{A}$  and any probabilistic efficient function  $\mathbf{aux}$  that satisfy the computational entropy condition:*

$$\mathbb{E}_{\substack{(sk, pk) \leftarrow KGen(1^\lambda) \\ m \leftarrow \mathcal{A}_0(pk)}} \text{HILL}_\infty(m \mid pk, \mathbf{aux}(m, pk)) \geq \omega(\log \lambda),$$

it holds that  $\text{Adv}_S^{\text{NR}}(\lambda, \mathcal{A}, \mathbf{aux}) \leq \text{negl}(\lambda)$ .

$\text{Adv}_S^{\text{NR}}(\lambda, \mathcal{A}, \mathbf{aux})$  is defined as the probability of adversary  $\mathcal{A}$  outputting 1 in the  $NR_S$ -Game when run with security parameter  $\lambda$  and signature scheme  $S$ .

### 2.7.3 Non-Resignability in the ROM

When defining Non-Resignability in the ROM, we not only switch from using a concrete hash function to a random oracle, but we also need to decide which algorithms are given access to the oracle. As in the plain model and in the reality, the used hash function is known, it is the natural choice to give access to the random oracle to all algorithms, so in our case, to  $\mathcal{A}_0, \mathcal{A}_1$  and  $\mathbf{aux}$ .

**Definition 2.19** (Non-Resignability-Game (ROM)) *Given a random oracle  $H$ , a signature scheme  $S^H = (KGen^H, Sign^H, Vrfy^H)$  and an explicit hint function  $\mathbf{aux}^H$  we define the security game  $NR_S^H$  for an adversary  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  as follows:*

$$\begin{aligned} & (sk, pk) \leftarrow KGen^H(1^\lambda) \\ & m \leftarrow \mathcal{A}_0^H(pk) \\ & h := \mathbf{aux}^H(m, pk) \\ & \sigma \leftarrow Sign^H(sk, m) \\ & (\bar{\sigma}, \bar{pk}) \leftarrow \mathcal{A}_1^H(pk, \sigma, h) \\ & v := Vrfy^H(\bar{pk}, m, \bar{\sigma}) \\ & \mathbf{return} (v = 1 \wedge \bar{pk} \neq pk) \end{aligned}$$

When we look at this new game definition, and would reuse the same entropy requirement as in definition 2.18, the following attack against every signature schemes would be suitable:

- $\mathcal{A}_0$  outputs  $H(1)$  as  $m$ .
- $\mathbf{aux}$  outputs some static hint (e.g. does not give any hint).
- $\mathcal{A}_1$  outputs  $(\text{Sign}(sk, H(1)), \bar{pk})$  for a new generated key pair  $\bar{sk}, \bar{pk}$ .

To solve this problem we need to change the entropy requirement. For this we will use the statistical min-entropy  $H_\infty$  which we will additionally condition on the oracle  $H$ .

Using the computational HILL entropy measure for the new definition would lead to problems when conditioning on the random oracle, as the representation of the random oracle is already exponential in the security parameter.

**Definition 2.20** (Non-Resignability (ROM)) *A signature scheme  $S^H = (KGen^H, Sign^H, Vrfy^H)$  is called non-resignable in the ROM if for any probabilistic efficient adversary  $\mathcal{A}^H$  and any probabilistic efficient function  $\mathbf{aux}^H$  that satisfies the statistical entropy condition:*

$$H_\infty_{\substack{(sk, pk) \leftarrow KGen^H(1^\lambda) \\ m \leftarrow \mathcal{A}_0^H(pk)}} (m \mid pk, \mathbf{aux}^H(m, pk), H) \geq \omega(\log \lambda)$$

it holds that  $\mathbf{Adv}_{S^H}^{NR^H}(\lambda, \mathcal{A}^H, \mathbf{aux}^H) \leq \text{negl}(\lambda)$ .

Again,  $\mathbf{Adv}_{S^H}^{NR^H}(\lambda, \mathcal{A}^H, \mathbf{aux}^H)$  is defined as the probability of  $\mathcal{A}^H$  outputting 1 in the  $NR_S^H$  game, when run with security parameter  $\lambda$  and auxiliary function  $\mathbf{aux}$ .

## 2.8 Exclusive Ownership

In [PS05] the notion of exclusive ownership was introduced. The idea of exclusive ownership is, that given a triple  $(pk, m, s)$  consisting of a public key  $pk$ , a message  $m$  and a signature  $s$  of  $m$  valid under  $pk$  one cannot efficiently calculate either a different public key  $pk'$  and different message  $m'$  such that  $Vrfy(pk', m', s) = \text{accept}$  or only a different message  $m'$  such that  $Vrfy(pk, m', s) = \text{accept}$ . If the former is fulfilled, we speak of Malicious-Strong Destructive Exclusive Ownership (M-S-DEO), if the later is fulfilled, we speak of Malicious-Strong Conservative Exclusive Ownership. If both are fulfilled, we call it Malicious-Strong Universal Exclusive Ownership (M-S-UEO).

In this thesis, we will only use M-S-UEO, so we will only define M-S-UEO formally.

**Definition 2.21** (Malicious-Strong Universal Exclusive Ownership (M-S-UEO) Experiment) *Let  $S$  be a digital signature scheme and  $\mathcal{A}$  an adversary that takes no input. We define  $\mathbf{Exp}_{S,\mathcal{A}}^{M-S-UEO}$  on input  $\lambda$  as follows:*

$(m_1, m_2, \sigma, pk_1, pk_2) \leftarrow \mathcal{A}()$   
 $d_1 \leftarrow \mathbf{Vrfy}(pk_1, m_1, \sigma)$   
 $d_2 \leftarrow \mathbf{Vrfy}(pk_2, m_2, \sigma)$   
**return**  $d_1 = \mathbf{accept} \wedge d_2 = \mathbf{accept} \wedge pk_1 \neq pk_2$

**Definition 2.22** (Malicious-Strong Universal Exclusive Ownership (M-S-UEO)) *Let  $S$  be a digital signature scheme. We say that  $S$  provides malicious-strong universal exclusive ownership (M-S-UEO) if for every probabilistic polynomial time adversary  $\mathcal{A}$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$*

$$\Pr[\mathbf{Exp}_{S,\mathcal{A}}^{M-S-UEO}(\lambda)] \leq \mu(\lambda).$$



## 3 BUFF- and BUFF-Lite-Transform

In [CDF<sup>+</sup>21], two new methods are introduced for transforming an EUF-CMA secure signature scheme into new schemes with enhanced security properties, utilizing a collision-resistant hash function. These constructions are referred to as the BUFF-Transform (beyond unforgeability features) and the BUFF-Lite-Transform.

In this chapter, we examine both the BUFF-Lite-Transform and the BUFF-Transform, discussing the security notions they aim to achieve and providing formal proofs for them. Furthermore, we analyze the findings of [DFHS24] and demonstrate why the BUFF-Transform, contrary to the original claims made in [CDF<sup>+</sup>21], does not satisfy the property of non-resignability. Also we will discuss the Salted-BUFF-Transform which was presented in [DFHS24], to fix the problems of the BUFF-Transform and see, that it does not provide non-resignability, but a slightly weaker notion.

	EUF-CMA	M-S-UEO	MBS	NR
BUFF-Lite-Transform	✓	✓	✓	✗
BUFF-Transform	✓	✓	✓	✗
Salted-BUFF-Transform	✓	✓	✓	✓

Table 3.1: Overview of security properties of the BUFF-Transform and its related transforms

### 3.1 BUFF-Lite-Transform

The first signature scheme transform presented in [CDF<sup>+</sup>21] is the BUFF-Lite-Transform. This is a simplified version of the BUFF-Transform, which we will discuss in section 3.2.

The BUFF-Lite-Transform is based on the prior work in [PS05] and combines the proposed transforms for providing M-S-UEO and MBS into the BUFF-Lite-Transform.

**Definition 3.1** (BUFF-Lite-Transform) *Given a signature scheme  $S = (KGen, Sign, Vrfy)$  and a hash function  $H$ , we define a transformed signature scheme  $S^* = (KGen^*, Sign^*, Vrfy^*)$ . The algorithms  $KGen^*$ ,  $Sign^*$ , and  $Vrfy^*$  are specified in fig. 3.1. This transform is called the BUFF-Lite-Transform.*

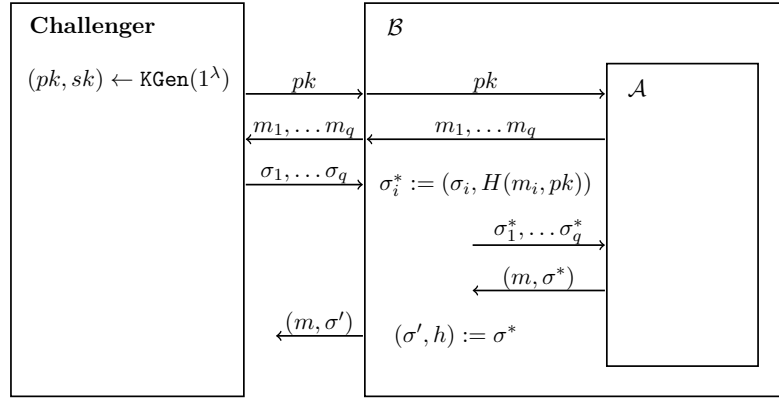
1: $KGen^*(1^\lambda)$ :	1: $Sign^*(sk, m)$ :	1: $Vrfy^*(pk, m, \sigma^*)$ :
2: $(sk, pk) \leftarrow KGen(1^\lambda)$	2: $h \leftarrow H(m, pk)$	2: $(\hat{\sigma}, \hat{h}) \leftarrow \sigma^*$
3: $\mathbf{return} (sk, pk)$	3: $\sigma \leftarrow Sign(sk, m)$	3: $h \leftarrow H(m, pk)$
	4: $\sigma^* \leftarrow (\sigma, h)$	4: $d \leftarrow Vrfy(pk, m, \hat{\sigma})$
	5: $\mathbf{return} \sigma^*$	5: $\mathbf{return} d = 1 \wedge \hat{h} = h$

Figure 3.1: BUFF-Lite-Transform as defined in fig. 5 in [CDF<sup>+</sup>21]

### 3.1.1 Security of the BUFF-Lite-Transform

**Theorem 3.2** (EUF-CMA conservation of the BUFF-Lite-Transform) *Let  $S$  be an EUF-CMA-secure signature scheme. Then the application of the BUFF-Lite-Transform in fig. 3.1 produces an EUF-CMA-secure signature scheme  $S^*$ .*

*Proof.* Given an adversary  $\mathcal{A}$  against the EUF-CMA-security of  $S^*$  we can construct an adversary  $\mathcal{B}$  which breaks  $S$  as follows:



We will now analyze the advantage of  $\mathcal{B}$  depending on the advantage of  $\mathcal{A}$ .

Our adversary  $\mathcal{B}$  simulates the challenger for  $\mathcal{A}$  correctly. For every signing query of  $m_i$ ,  $\mathcal{A}$  expects the answer  $\text{Sign}^*(sk, m) = (\text{Sign}(sk, m_i), H(m_i, pk))$ . As  $\mathcal{B}$  forwards  $m_i$  to the challenger and it answers with  $\text{Sign}(sk, m_i)$  and as  $\mathcal{B}$  knows  $pk$  and can calculate  $H(pk, m)$  on its own,  $\mathcal{B}$  can send the expected answer  $(\text{Sign}(sk, m_i), H(m_i, pk))$  to  $\mathcal{A}$ .

When  $\mathcal{A}$  eventually outputs its forgery  $(m, \sigma^*)$ , where  $\sigma^* = (\sigma, h)$ ,  $\mathcal{B}$  will extract the signature  $\sigma$  from it and outputs  $(m, \sigma)$  as its own forgery. We have that  $\text{Vrfy}^*(pk, m, \sigma^*) = \text{accept}$  holds only if  $\text{Vrfy}(pk, m, \sigma) = \text{accept}$ , due to the construction of  $\text{Vrfy}^*$ . So if  $\mathcal{A}$  wins its simulated game,  $\mathcal{B}$  wins as well.

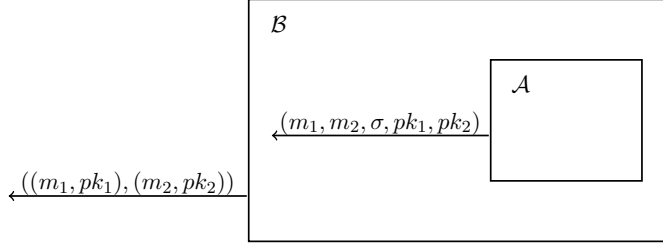
Using this, we can conclude:

$$\text{Adv}_{S^*}^{\text{EUF-CMA}}(\lambda, \mathcal{A}) \leq \text{Adv}_S^{\text{EUF-CMA}}(\lambda, \mathcal{B}).$$

If  $S$  is an EUF-CMA-secure signature scheme,  $\text{Adv}_S^{\text{EUF-CMA}}$  is negligible for all efficient adversaries. Thus  $\text{Adv}_{S^*}^{\text{EUF-CMA}}$  must be negligible for all efficient adversaries as well, because otherwise, using our specified  $\mathcal{B}$ , we could construct an efficient adversary breaking  $S$  EUF-CMA-security.  $\square$

**Theorem 3.3** (M-S-UEO property of the BUFF-Lite-Transform) *Let  $S$  be a signature scheme. Then the application of the BUFF-Lite-Transform given in fig. 3.1 produces a signature scheme  $S^*$  that provides M-S-UEO assuming that the hash function  $H$  is collision resistant.*

*Proof.* Given a probabilistic polynomial time adversary  $\mathcal{A}$  against the M-S-UEO-security of  $S^*$  we can construct a probabilistic polynomial time adversary  $\mathcal{B}$  which breaks  $H$  as follows:



If  $\mathcal{A}$  is polynomial time,  $\mathcal{B}$  is polynomial time as well, as  $\mathcal{B}$  only runs  $\mathcal{A}$  and extracts  $m_1, m_2$  from its output.

We will now analyze the advantage of  $\mathcal{B}$  depending on the advantage of  $\mathcal{A}$ . If we assume  $\mathcal{A}$  wins in the M-S-UEO game, by definition of the game the following assumptions must hold:  $\text{Vrfy}(pk_1, m_1, \sigma) = \text{accept}$ ,  $\text{Vrfy}(pk_2, m_2, \sigma) = \text{accept}$  and  $pk_1 \neq pk_2$ . We have by construction of  $S^*$ , that  $\sigma$  consists of  $\hat{\sigma}$  and  $\hat{h}$  and  $\text{Vrfy}(pk_i, m_i, \sigma) = \text{accept}$  implies  $H(m_i, pk_i) = \hat{h}$  must hold. This leads to  $H(m_1, pk_1) = \hat{h} = H(m_2, pk_2)$ . As  $pk_1 \neq pk_2$  we have found a collision on  $H$  and  $\mathcal{B}$  wins.

This leads to the following inequality:

$$\text{Adv}_{H, \mathcal{B}}^{CR}(\lambda) \geq \text{Exp}_{S, \mathcal{A}}^{\text{M-S-UEO}}(\lambda).$$

As we assumed that  $H$  is collision resistant,  $\text{Adv}_{H, \mathcal{B}}^{Col}(\lambda)$  must be negligible for all efficient probabilistic adversaries, and thus  $\text{Exp}_{S, \mathcal{A}}^{\text{M-S-UEO}}$  must be negligible for all efficient probabilistic adversaries as well.  $\square$

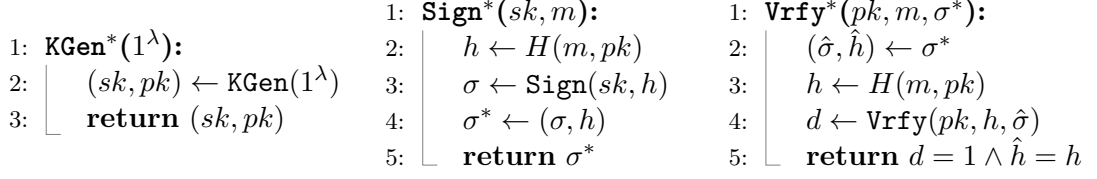
**Theorem 3.4** (MBS property of the BUFF-Transform) *Let  $S$  be a signature scheme. Then the application of the BUFF-Lite-Transform given in fig. 3.1 produces a signature scheme  $S^*$  that provides MBS assuming that the hash function  $H$  is collision resistant.*

*Proof.* The proof of this theorem follows the same structure as the proof of definition 3.3. The only difference is that the adversary  $\mathcal{A}$  outputs a tuple  $(m_1, m_2, \sigma, pk)$  such that  $H(m_1, pk) = H(m_2, pk)$  holds if  $\mathcal{A}$  wins in the MBR game, due to  $\text{Vrfy}(pk, m_i, \sigma) = \text{accept}$  for  $i \in \{1, 2\}$ , thereby once again violating the collision resistance of the hash function.  $\square$

## 3.2 BUFF-Transform

In addition to the BUFF-Lite-Transform [CDF<sup>+</sup>21] also introduced the BUFF-Transform, a more sophisticated variant of the BUFF-Lite-Transform. Originally it was thought, that the BUFF-Transform yields the same security properties as the BUFF-Lite-Transform and additionally also non-resignability. [DFHS24] showed, that the non-resignability does not hold in the ROM.

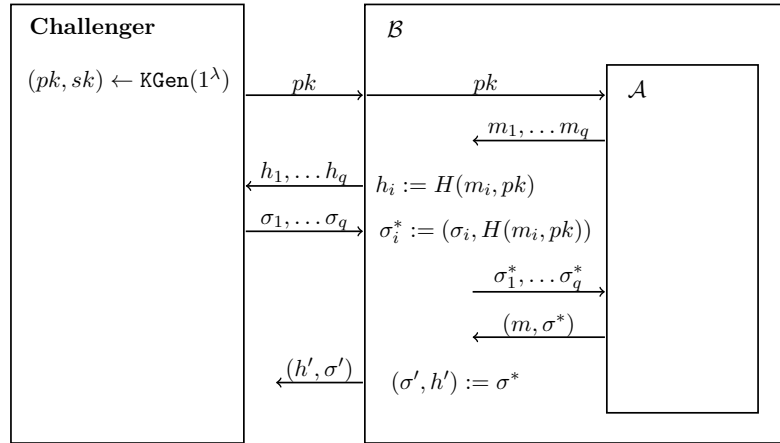
**Definition 3.5** (BUFF-Transform) *Given a signature scheme  $S = (\text{KGen}, \text{Sign}, \text{Vrfy})$  and a hash function  $H$ , we define a transformed signature scheme  $S^* = (\text{KGen}^*, \text{Sign}^*, \text{Vrfy}^*)$ . The algorithms  $\text{KGen}^*$ ,  $\text{Sign}^*$ , and  $\text{Vrfy}^*$  are specified in fig. 3.2. This transform is called the BUFF-Transform.*


 Figure 3.2: BUFF-Transform as defined in fig. 6 in [CDF<sup>+</sup>21]

### 3.2.1 Security of the BUFF-Transform

**Theorem 3.6** (EUF-CMA conservation of the BUFF-Transform) *Let  $S$  be an EUF-CMA-secure signature scheme. Then the application of the BUFF-Transform in fig. 3.2 produces an EUF-CMA-secure signature scheme  $S^*$ , assuming that the hash function  $H$  is collision resistant.*

*Proof.* Given an adversary  $\mathcal{A}$  against the EUF-CMA-security of  $S^*$  we can construct an adversary  $\mathcal{B}$  which breaks  $S$  as follows:



We will now analyze the advantage of  $\mathcal{B}$  depending on the advantage of  $\mathcal{A}$ .

Our adversary  $\mathcal{B}$  simulates the challenger for  $\mathcal{A}$  correctly. For every signing query of  $m_i$ ,  $\mathcal{A}$  expects the answer  $\text{Sign}^*(sk, m) = (\text{Sign}(sk, H(m_i, pk)), H(m_i, pk))$ . As  $\mathcal{B}$  takes  $m_i$  and uses the known  $pk$  to calculate  $H(m_i, pk)$  and sends it to the challenger. Then the challenger answers the query with  $\text{Sign}(sk, H(m_i, pk))$  and as  $\mathcal{B}$  knows  $pk$  and it can thus calculate  $H(m_i, pk)$  on its own, allowing  $\mathcal{B}$  to send the expected answer  $(\text{Sign}(sk, H(m_i, pk)), H(m_i, pk))$  to  $\mathcal{A}$ .

When  $\mathcal{A}$  eventually outputs its forgery  $(m, \sigma^*)$ , where  $\sigma^* = (\sigma, h)$ ,  $\mathcal{B}$  will extract the signature  $\sigma$  and the hash  $h$  from it and outputs  $(h, \sigma)$  as its own forgery. The forgery  $(h, \sigma)$  is a valid forgery, if and only if  $\text{Vrfy}(pk, h, \sigma) = \text{accept}$  and there was no query from  $\mathcal{B}$  for  $h$  (i.e.  $\mathcal{A}$  has not queried for a  $m_i$  s.t.  $H(m_i) = h$ ).

If  $\mathcal{A}$  wins in its corresponding game,  $\text{Vrfy}(pk, h, \sigma)$  must hold by construction of  $S^*$ . If  $H$  is collision resistant, the probability, that  $h = H(m_i)$  for any  $m_i$  is negligible, but at most  $\#\text{Queries} \cdot \text{Adv}_H^{CR}$ , as  $h = H(m)$  for an  $m \neq m_i$  for all  $i$ .

Using this, we can conclude:

$$\text{Adv}_{S^*}^{EUF-CMA}(\lambda, \mathcal{A}) \leq \text{Adv}_S^{EUF-CMA}(\lambda, \mathcal{B}) + \#\text{Queries} \cdot \text{Adv}_H^{CR}.$$

If  $S$  is an EUF-CMA-secure signature scheme,  $\mathbf{Adv}_S^{\text{EUF-CMA}}$  is negligible for all efficient probabilistic adversaries. If  $H$  is collision resistant  $\mathbf{Adv}_H^{\text{CR}}$  is negligible and as  $\mathcal{A}$  is efficient, the number of queries is polynomial bound.

This leads to  $\mathbf{Adv}_{S^*}^{\text{EUF-CMA}}$  being negligible for all efficient probabilistic adversaries as well, because otherwise, using our specified  $\mathcal{B}$  we could construct an efficient adversary breaking the EUF-CMA-security of  $S$ .  $\square$

**Theorem 3.7** (M-S-UEO property of the BUFF-Transform) *Let  $S$  be a signature scheme. Then the application of the BUFF-Transform given in fig. 3.2 produces a signature scheme  $S^*$  that provides M-S-UEO assuming that the hash function  $H$  is collision resistant.*

*Proof.* Analogous to proof 3.3 we can conclude here that the BUFF-Transform provides M-S-UEO as well. This is because for the proof it is mostly irrelevant, if we sign the message directly or sign its hash, as long as the hash function is collision resistant.  $\square$

**Theorem 3.8** (MBS property of the BUFF-Transform) *Let  $S$  be a signature scheme. Then the application of the BUFF-Transform given in fig. 3.2 produces a signature scheme  $S^*$  that provides MBS assuming that the hash function  $H$  is collision resistant.*

*Proof.* The proof of this theorem follows the same structure as the proof of definition 3.3. The only difference is that the adversary  $\mathcal{A}$  outputs a tuple  $(m_1, m_2, \sigma, pk)$  such that  $H(m_1, pk) = H(m_2, pk)$  holds if  $\mathcal{A}$  wins in the MBR game, due to  $\mathbf{Vrfy}(pk, m_i, \sigma) = \text{accept}$  for  $i \in \{1, 2\}$ , thereby once again violating the collision resistance of the hash function.  $\square$

### 3.2.2 Resignability of the BUFF-Transform

[CDF<sup>+</sup>21] claimed, that the BUFF-Transform additionally provides non-resignability, both in the ROM and the Plain Model. In an analysis of the BUFF transform in [DFHS24] it was shown, that the BUFF-Transform does not provide non-resignability, neither in the ROM nor in the plain model. In this section we will discuss the findings of [DFHS24].

#### Plain Model

**Theorem 3.9** *Let  $S$  be an EUF-CMA secure signature scheme over message space  $\mathcal{M}$  that satisfies the following property:*

$$\Pr_{m \leftarrow \mathcal{M}, (sk, pk) \leftarrow \text{KGen}(1^\lambda)} [\mathbf{HILL}_\infty(m \mid pk, \text{Sign}(sk, m)) \geq \omega(\log \lambda)] = 1.$$

*Then there exists in the non-resignability game a probabilistic polynomial time adversary  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$  and a probabilistic polynomial time function  $\text{aux}$ , that satisfies the computational entropy requirement in definition 2.18, such that:*

$$\mathbf{Adv}_S^{\text{NR}}(\lambda, \mathcal{A}, \text{aux}) \geq 1 - \text{negl}(\lambda).$$

*Proof.* We will now proof the theorem 3.9 by using the adversary  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$  and auxiliary function  $\text{aux}$  defined in fig. 3.3.

1: $\mathcal{A}_0$ : 2: $\left[ \begin{array}{l} m \leftarrow \mathcal{M} \\ \mathbf{return} m \end{array} \right.$	1: $\mathcal{A}_1(pk, \sigma, h)$ : 2: $\left[ \begin{array}{l} (\bar{\sigma}, \bar{pk}) \leftarrow h \\ \mathbf{return} (\bar{\sigma}, \bar{pk}) \end{array} \right.$	1: $\mathbf{aux}(m)$ : 2: $\left[ \begin{array}{l} (\bar{sk}, \bar{pk}) \leftarrow \text{KGen}(1^\lambda) \\ \sigma \leftarrow \text{Sign}(\bar{sk}, m) \\ h \leftarrow (\text{sigma}, \bar{pk}) \\ \mathbf{return} h \end{array} \right.$
------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 3.3: The adversary and auxiliary function to prove theorem 3.9

In order to prove this, we need to show, that the output of  $\mathcal{A}_1$  is valid i.e.  $pk \neq \bar{pk}$  and  $\text{Vrfy}(\bar{pk}, m, \bar{\sigma}) = \text{accept}$  with probability of at least  $1 - \text{negl}(\lambda)$ . Additionally we need to show, that the entropy requirement on the output of  $\mathbf{aux}$  is fulfilled, i.e. the HILL entropy is at least  $\omega(\log \lambda)$ .

We will first show, that the output of  $\mathcal{A}_1$  is valid. By construction of  $\mathcal{A}$  the output  $\bar{pk}$  is generated by  $\mathbf{aux}$ .  $\mathbf{aux}$  uses the key generation algorithm  $\text{KGen}$  to do this and as  $S$  is EUF-CMA secure it holds that  $\Pr[pk = \bar{pk}] \leq \text{negl}(\lambda)$  and we get by transformation  $\Pr[pk \neq \bar{pk}] \geq 1 - \text{negl}(\lambda)$ . Additionally  $\bar{\sigma}$  is generated by  $\mathbf{aux}$  and by construction  $\bar{\sigma} = \text{Sign}(\bar{sk}, m)$ . As  $S$  is a signature scheme, it follows:  $\text{Vrfy}(\bar{pk}, m, \bar{\sigma}) = \text{accept}$  with probability 1. Combining these, we get that the output of  $\mathcal{A}_1$  is valid, so it remains to show, that  $\mathbf{aux}$  fulfills the entropy requirement:

$$\text{HILL}_\infty(m \mid pk, \mathbf{aux}(m, pk)) = \text{HILL}_\infty(m \mid \mathbf{aux}(m, pk)) \quad (3.1)$$

$$= \text{HILL}_\infty(m \mid \bar{pk}, \text{Sign}(\bar{sk}, m)) \quad (3.2)$$

$$\geq \omega(\log \lambda) \quad (3.3)$$

Equation (3.1) holds, as  $m$  and  $pk$  are independent and thus the entropy of  $m$  cannot be dependent of  $pk$ , eq. (3.2) holds due to the construction of  $\mathbf{aux}$  and eq. (3.3) holds due to the entropy requirement made in theorem 3.9.

In conclusion, the adversary  $\mathcal{A}$  together with  $\mathbf{aux}$  plays out a valid attack such that

$$\text{Adv}_S^{\text{NR}}(\lambda, \mathcal{A}, \mathbf{aux}) \geq 1 - \text{negl}(\lambda)$$

holds. □

**Corollary 3.10** *Let  $S$  be an EUF-CMA signature scheme and  $H$  a (keyed) hash function that compresses at least by  $|pk| + \omega(\log \lambda)$ . Then the signature scheme  $S^*$  produced by the application of the BUFF-Transform in fig. 3.2 does not provide non-resignability.*

*Proof.* We will now show, that the compression requirement, translates into the entropy requirement of theorem 3.9, after which the statement follows directly. This means, we need to show that for the auxiliary function  $\mathbf{aux}$  from fig. 3.3  $\text{HILL}_\infty(m \mid \bar{pk}, \text{Sign}(\bar{sk}, m)) \geq \omega(\log \lambda)$  is fulfilled.

First we observe, that  $\bar{pk}$  gets chosen independently of  $m$  so  $\text{HILL}_\infty(m \mid \bar{pk}, \text{Sign}(\bar{sk}, m)) = \text{HILL}_\infty(m \mid \text{Sign}(\bar{sk}, m))$ .

Unpacking the definition, we find that  $\text{Sign}(\bar{sk}, m)$  is in fact given by

$$\text{Sign}(\bar{sk}, m) = (H(m, pk), \text{Sign}'(sk, H(m, pk))),$$

where  $\text{Sign}'$  denotes the signing algorithm of the underlying signature scheme. As  $H$  compresses by at least  $|pk| + \omega(\log \lambda)$ , we can calculate a lower bound for the expected

value of messages leading to the same signature:

$$\mathbb{E}_{m \leftarrow \mathcal{M}}[|\{m' \in \mathcal{M} \mid H(m', pk) = H(m, pk)\}|] \geq |pk| + \omega(\log \lambda).$$

Which directly gives us the following lower bound for the entropy:

$$\text{HILL}_\infty(m \mid \text{Sign}(\overline{sk}, m)) \geq |pk| + \omega(\log \lambda)$$

and by extension we get the requirement for applying theorem 3.9:

$$\text{HILL}_\infty(m \mid \overline{pk}, \text{Sign}(\overline{sk}, m)) \geq \omega(\log \lambda).$$

This concludes our proof, as we have found an adversary  $\mathcal{A}$  against the non-resignability of  $S^*$  with advantage greater than  $1 - \text{negl}(\lambda)$ , which means  $S^*$  does not provide non-resignability.  $\square$

The assumption that  $H$  compresses at least by  $|pk| + \omega(\log \lambda)$  is reasonable. In reality hash functions with output-size between 256 and 512 bits and signature schemes with more then 3000 bits key size for RSA and FF-DLOG based signatures are recommended ([SOG23]). When we assume a key-size of 4096 (which then is also the security parameter in the case of RSA) and use  $\lambda \mapsto \lambda$  as an explicit function for  $\omega(\log \lambda)$  we can calculate the worst case at which text length  $|w|$  (in binary encoding) the compression requirement holds:

$$\begin{aligned} |w| - (|pk| + \omega(\log \lambda)) &= |H(w)| \\ |w| - 4096 - 4096 &= 512 \\ |w| &= 8204. \end{aligned}$$

As messages are generally much larger then 8204 bits (1026 bytes), we can conclude that in most real world applications, even with large security parameter  $\lambda$ , the assumed compression is reached. For example this section is over 5000 bytes large. So generally the BUFF-Transform does not provide non-resignability.

## ROM

The findings from the plain model carry over to the ROM with some minor adjustments. These changes are necessary, as not all assumptions from the plain model hold in the ROM and the definition of non-resignability is slightly different, as already discussed in section 2.7.3.

**Theorem 3.11** *Let  $S$  be an EUF-CMA secure signature scheme over message space  $\mathcal{M}$  that satisfies the following property:*

$$\Pr_{m \leftarrow \mathcal{M}, (sk, pk) \leftarrow \text{KGen}(1^\lambda)}[H_\infty(m \mid pk, \text{Sign}(sk, m)) \geq \omega(\log \lambda)] = 1$$

*Then there exists in the non-resignability game a probabilistic polynomial time adversary  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  and a probabilistic polynomial time function  $\mathbf{aux}^H$ , that satisfies the computational entropy requirement in definition 2.20, such that:*

$$\text{Adv}_S^{\text{NR}}(\lambda, \mathcal{A}, \mathbf{aux}) \geq 1 - \text{negl}(\lambda)$$

<pre> 1: <math>\mathbf{KGen}^*(1^\lambda)</math>: 2:   <math>(sk, pk) \leftarrow \mathbf{KGen}(1^\lambda)</math> 3:   <b>return</b> <math>(sk, pk)</math> </pre>	<pre> 1: <math>\mathbf{Sign}^*(sk, m)</math>: 2:   <math>s \leftarrow \{0, 1\}^l</math> 3:   <math>h \leftarrow H(m, pk, s)</math> 4:   <math>\sigma \leftarrow \mathbf{Sign}(sk, h)</math> 5:   <math>\sigma^* \leftarrow (\sigma, h, s)</math> 6:   <b>return</b> <math>\sigma^*</math> </pre>	<pre> 1: <math>\mathbf{Vrfy}^*(pk, m, \sigma^*)</math>: 2:   <math>(\hat{\sigma}, \hat{h}, \hat{s}) \leftarrow \sigma^*</math> 3:   <math>h \leftarrow H(m, pk, \hat{s})</math> 4:   <math>d \leftarrow \mathbf{Vrfy}(pk, h, \sigma)</math> 5:   <b>return</b> <math>d = 1 \wedge \hat{h} = h</math> </pre>
------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 3.4: Salted-BUFF-Transform as defined in fig. 8 in [DFHS24]

**Corollary 3.12** *Let  $S$  be an EUF-CMA signature scheme and  $H$  a random oracle that compresses at least by  $|pk| + \omega(\log \lambda)$ . Then the signature scheme  $S^*$  produced by the application of the BUFF-Transform in fig. 3.2 does not provide non-resignability in the ROM.*

The proofs for theorem 3.11 and corollary 3.12 work the same as the corresponding plain model counterparts. The only changes which need to be done, are using the random oracle, instead of the hash function and exchanging the computational entropy with the statistical entropy.

### 3.3 Salted-BUFF-Transform

In the light of the findings, we discussed in section 3.2.2, [DFHS24] proposed a way to alter the BUFF-Transform to add non-resignability, while keeping the other security notions. The new scheme is called Salted-BUFF-Transform ( $\$$ -BUFF). The basic idea is to add a salt to the input of the hash functions. We first define the new transform, then explain why the proof for EUF-CMA from the BUFF-Transform setting carries over. Finally, we show that the Salted-BUFF-Transform also satisfies a weaker non-resignability property in the ROM.

**Definition 3.13** (Salted-BUFF-Transform) *Given a signature scheme  $S = (\mathbf{KGen}, \mathbf{Sign}, \mathbf{Vrfy})$  and a hash function  $H$ , we define a transformed signature scheme  $S^* = (\mathbf{KGen}^*, \mathbf{Sign}^*, \mathbf{Vrfy}^*)$ . The algorithms  $\mathbf{KGen}^*$ ,  $\mathbf{Sign}^*$ , and  $\mathbf{Vrfy}^*$  are specified in fig. 3.4. This transform is called the salted-BUFF-Transform.*

**Theorem 3.14** (EUF-CMA conservation of the Salted-BUFF-Transform) *Let  $S$  be an EUF-CMA-secure signature scheme. Then the application of the Salted-BUFF-Transform in fig. 3.4 produces an EUF-CMA-secure signature scheme  $S^*$ , assuming that the hash function  $H$  is collision resistant.*

*Proof.* The argument from theorem 3.6 applies here with only minor modifications. The sole difference is that the outer adversary  $\mathcal{B}$  must generate a fresh salt for each signing query and must include it in the input to the hash function.  $\square$

#### 3.3.1 Weak Non-Resignability

As shown in section 3.2.2, neither in the plain model nor in the ROM, there can be a signature scheme providing non-resignability where the message has at least a certain entropy conditioned on the signature and public key. When we would analyze the entropy of the message in the salted-BUFF-Transform we would also get, that the salted-BUFF-Transform does not yield non-resignability. To overcome this, we will define a weaker

variant of the non-resignability, where the auxiliary function does not get access to the random oracle.

**Definition 3.15** (Weak Non-Resignability-Game (ROM)) *Given a random oracle  $H$ , a signature scheme  $S^H = (\mathit{KGen}^H, \mathit{Sign}^H, \mathit{Vrfy}^H)$  and an explicit hint function  $\mathbf{aux}$ , we define the security game  $\mathit{NR}_S^{H,\perp}$  for an adversary  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  as follows:*

$$\begin{aligned} (sk, pk) &\leftarrow \mathit{KGen}^H(1^\lambda) \\ m &\leftarrow \mathcal{A}_0^H(pk) \\ h &:= \mathbf{aux}(m, pk) \\ \sigma &\leftarrow \mathit{Sign}^H(sk, m) \\ (\bar{\sigma}, \bar{pk}) &\leftarrow \mathcal{A}_1^H(pk, \sigma, h) \end{aligned}$$

We say  $\mathcal{A}^H$  wins, if  $\mathit{Vrfy}^H(\bar{pk}, m, \bar{\sigma}) = \mathbf{accept}$  and  $\bar{pk} \neq pk$ .

The advantage is defined as follows:

$$\mathbf{Adv}_{S^H}^{\mathit{NR}^{H,\perp}}(\lambda, \mathcal{A}^H, \mathbf{aux}) = \Pr[\mathcal{A}^H \text{ wins}].$$

**Definition 3.16** (Weak Non-Resignability (ROM)) *A signature scheme  $S^H = (\mathit{KGen}^H, \mathit{Sign}^H, \mathit{Vrfy}^H)$  is called weak non-resignable in the ROM, if for any probabilistic efficient adversary  $\mathcal{A}^H$  and any probabilistic efficient function  $\mathbf{aux}$  that satisfies the statistical entropy condition:*

$$H_\infty_{\substack{(sk, pk) \leftarrow \mathit{KGen}^H(1^\lambda) \\ m \leftarrow \mathcal{A}_0^H(pk)}}(m \mid pk, \mathbf{aux}(m, pk), H) \geq \omega(\log \lambda)$$

it holds that  $\mathbf{Adv}_{S^H}^{\mathit{NR}^{H,\perp}}(\lambda, \mathcal{A}^H, \mathbf{aux}) \leq \mathit{negl}(\lambda)$ .

One might ask now, if the BUFF-Transform does fulfill this new weak notion, but as our attack did not use the oracle in the auxiliary function, the BUFF-Transform is also not weak non-resignable. But we can show, that the salted-BUFF-Transform does yield weak non-resignability indeed.

**Theorem 3.17** (Weak Non-Resignability of the Salted-BUFF-Transform) *Let  $H$  be a random oracle which has output space  $\mathcal{Y}$ , let  $S^H = (\mathit{KGen}, \mathit{Sign}^H, \mathit{Vrfy}^H)$  be a signature scheme, where the key generation has no access to the random oracle  $H$  and  $\mathit{Sign}^H$  makes at most  $q_S$  queries to  $H$ . Let  $S^{H*}$  be the signature scheme obtained by applying the Salted-BUFF-Transform given in fig. 3.4. Let  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  be a computational unbounded  $\mathit{NR}^{H,\perp}$  adversary, where  $\mathcal{A}_0^H$  and  $\mathcal{A}_1^H$  make at most  $q_0, q_1$  queries to  $H$  respectively and  $\mathbf{aux}$  a (possibly randomized) function. Assuming*

$$H_\infty_{\substack{(sk, pk) \leftarrow \mathit{KGen}(1^\lambda) \\ m \leftarrow \mathcal{A}_0^H(pk)}}(m \mid H, pk, \mathbf{aux}(m, pk)) \geq \log\left(\frac{1}{\varepsilon}\right)$$

then the Advantage is bounded as follows

$$\mathbf{Adv}_{S^{H*}}^{\mathit{NR}^{H,\perp}}(\mathcal{A}, \mathbf{aux}) \leq q_0 \cdot 2^{-l} + 2(q_1 + q_S) \cdot \varepsilon + \frac{q_0 + 1}{|\mathcal{Y}|}.$$

To prove this, we need to show first, that the random oracle  $H$  has certain properties, which we will call weak- $\Phi$ -salted-non-malleability, a refined variant of the  $\Phi$ -non-malleability we introduced in section 2.6.2.

**Definition 3.18** (weak- $\Phi$ -salted-non-malleability (ROM)) *For a random oracle  $H$  and a class of admissible transformations  $\Phi$  we define the weak- $\Phi$ -salted-non-malleability*

( $\Phi\$NM^{H,\perp}$ ) game between a challenger and an adversary  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  with an auxiliary function  $\mathbf{aux}$  for a given public key  $pk$  as follows:

$$\begin{aligned} m &\leftarrow \mathcal{A}_0^H(pk) \\ h &:= \mathbf{aux}(m, pk) \\ s &\leftarrow \{0, 1\}^l \\ y &:= H(m, pk, s) \\ (\bar{y}, \bar{pk}, \bar{s}) &\leftarrow \mathcal{A}_1^H(y, h, s) \end{aligned}$$

The adversary  $\mathcal{A}$  wins, if  $H(m, \bar{pk}, \bar{s}) = \bar{y}$  and  $\bar{pk} \neq pk$ . We define the advantage as follows:

$$\mathbf{Adv}_H^{\Phi\$NM^{H,\perp}(pk)}(\lambda, \mathcal{A}^H, \mathbf{aux}^H) = \Pr[\mathcal{A}^H \text{ wins}].$$

**Lemma 3.19** *Let  $H$  be a random oracle which maps to  $\mathcal{Y}$ . Let  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  be a computational unbounded adversary against  $\Phi\$NM^{H,\perp}$ , where  $\mathcal{A}_0^H$  and  $\mathcal{A}_1^H$  make at most  $q_0$  and  $q_1$  queries respectively to the oracle  $H$ . Let  $\mathbf{aux}$  be a possibly randomized function. We define  $\varepsilon := \mathbf{guess}_{m \leftarrow \mathcal{A}^H}(m \mid H, \mathbf{aux}(m, pk))$  and set  $pk$  to be an arbitrary valid key for the security parameter  $\lambda$ . Then*

$$\mathbf{Adv}_H^{\Phi\$NM^{H,\perp}(pk)}(\lambda, \mathcal{A}^H, \mathbf{aux}^H) \leq q_0 \cdot 2^{-l} + 2q_1 \cdot \varepsilon + \frac{q_0 + 1}{|\mathcal{Y}|}.$$

*Proof.* To prove lemma 3.19, we will use game hopping with the games defined in fig. 3.5.

**$G_0$  to  $G_1$ :** The difference between  $G_0$  and  $G_1$ , is that in  $G_1$  the oracle  $H$  gets reprogrammed to output a fixed  $y$  when queried for  $(m, pk, s)$ . This change is only relevant, if  $\mathcal{A}_0^H$  has queried the exact same query. As  $s$  gets chosen randomly and is unknown to  $\mathcal{A}_0^H$  this can happen with probability of at most  $q_0 \cdot 2^{-l}$  as there are  $2^l$  possible values for  $s$ . Due to the [difference lemma](#) we can conclude:

$$|\Pr[G_0 = 1] - \Pr[G_1 = 1]| \leq q_0 \cdot 2^{-l}.$$

**$G_1$  to  $G_2$ :** The next difference is, that now the oracle does not get reprogrammed anymore, while keeping the randomly chosen  $y$  as input to  $\mathcal{A}_1^H$ . So these games only differ, if  $\mathcal{A}_1^H$  happens to query for  $(m, pk, s)$ . As  $y$  and  $s$  are independent of either  $m$ ,  $H$  and  $h$ , we get:  $\mathbf{guess}(m \mid H, y, h, s) = \mathbf{guess}(m \mid H, h)$  and by applying the requirement of the lemma 3.19 we have  $\mathbf{guess}(m \mid H, y, h, s) \leq \varepsilon$ . As  $\mathcal{A}_1^H$  makes at most  $q_1$  queries to the oracle, we can conclude, that in at most  $q_1 \cdot \varepsilon$  cases the games  $G_1$  and  $G_2$  differ and by application of the [difference lemma](#):

$$|\Pr[G_1 = 1] - \Pr[G_2 = 1]| \leq q_1 \cdot \varepsilon.$$

**$G_2$  to  $G_3$ :** Now the only change is, that the oracle, as perceived by  $\mathcal{A}_1^H$  behaves different, if a query of the form  $(\cdot, m, \cdot)$  is done. This will happen with probability at most  $q_1 \cdot \varepsilon$  as with the same argument as above:  $\mathbf{guess}(m \mid H, y, h, s) \leq \varepsilon$  holds. Consequently the difference in the success probabilities between the games is bounded by

$$|\Pr[G_2 = 1] - \Pr[G_3 = 1]| \leq q_1 \cdot \varepsilon.$$

**$G_3$  to  $G_4$ :** In this step we change the winning condition, by relaxing it and not requiring  $pk \neq \bar{pk}$  anymore. This just increases the winning probability, so

$$\Pr[G_3 = 1] \leq \Pr[G_4 = 1].$$

<ol style="list-style-type: none"> <li>1: GAME <math>G_0</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y := H(m, pk, s)</math></li> <li>6:</li> <li>7: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^H(y, h, s)</math></li> <li>8: <math>\text{return } H(\bar{pk}, m, \bar{s}) = \bar{y} \wedge \bar{pk} \neq pk</math></li> </ol>	<ol style="list-style-type: none"> <li>1: GAME <math>G_1</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y \leftarrow \mathcal{Y}</math></li> <li>6: <math>H(pk, m, s) := y</math></li> <li>7: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^H(y, h, s)</math></li> <li>8: <math>\text{return } H(\bar{pk}, m, \bar{s}) = \bar{y} \wedge \bar{pk} \neq pk</math></li> </ol>
<ol style="list-style-type: none"> <li>1: GAME <math>G_2</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y \leftarrow \mathcal{Y}</math></li> <li>6: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^H(y, h, s)</math></li> <li>7: <math>\text{return } H(\bar{pk}, m, \bar{s}) = \bar{y} \wedge \bar{pk} \neq pk</math></li> </ol>	<ol style="list-style-type: none"> <li>1: GAME <math>G_3</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y \leftarrow \mathcal{Y}</math></li> <li>6: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^{H[(\cdot, m, \cdot) \mapsto \perp]}(y, h, s)</math></li> <li>7: <math>\text{return } H(\bar{pk}, m, \bar{s}) = \bar{y} \wedge \bar{pk} \neq pk</math></li> </ol>
<ol style="list-style-type: none"> <li>1: GAME <math>G_4</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y \leftarrow \mathcal{Y}</math></li> <li>6: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^{H[(\cdot, m, \cdot) \mapsto \perp]}(y, h, s)</math></li> <li>7: <math>\text{return } H(\bar{pk}, m, \bar{s}) = \bar{y}</math></li> </ol>	<ol style="list-style-type: none"> <li>1: GAME <math>G_5^i</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y \leftarrow \mathcal{Y}</math></li> <li>6: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^{H[(\cdot, m, \cdot) \mapsto \perp]}(y, h, s)</math></li> <li>7: <math>\text{return } H(\bar{pk}, m, \bar{s}) = \bar{y} \wedge</math> <math>(\bar{pk}, m, \bar{s}) = (pk^i, m^i, s^i)</math></li> </ol>
<ol style="list-style-type: none"> <li>1: GAME <math>G_6^i</math>:</li> <li>2: <math>m \leftarrow \mathcal{A}_0^H(pk)</math></li> <li>3: <math>h := \text{aux}(m^i, pk)</math></li> <li>4: <math>s \leftarrow \{0, 1\}^l</math></li> <li>5: <math>y \leftarrow \mathcal{Y}</math></li> <li>6: <math>(\bar{y}, \bar{pk}, \bar{s}) \leftarrow \mathcal{A}_1^{H[(\cdot, m^i, \cdot) \mapsto \perp]}(y, h, s)</math></li> <li>7: <math>\text{return } H(pk^i, m^i, s^i) = \bar{y}</math></li> </ol>	<p><math>H[(\cdot, m, \cdot) \mapsto \perp]</math> denotes that in queries for <math>(\cdot, m, \cdot)</math> the answers are replaced with a special value <math>\perp</math>.</p> <p>In game <math>G_5</math> and <math>G_6</math> <math>(pk^i, m^i, s^i)</math> denotes the <math>i</math>-th Query of <math>\mathcal{A}_0^H</math></p>

 Figure 3.5: The games  $G_0$  to  $G_6^i$  to prove lemma 3.19 as defined in Fig. 11 in [DFHS24]

**$G_4$  to  $G_5^i$ :** Without loss of generality we assume, that  $\mathcal{A}_0^H$  does not repeat queries, as  $H$  is deterministic and answers can be cached without effect. If  $(\overline{pk}, m, \overline{s})$  is different to every query of  $\mathcal{A}_0^H$ , we have that  $H(\overline{pk}, m, \overline{s})$  is random and independent from  $y$ , as  $\mathcal{A}_1^H$  is blocked from querying for any  $(\cdot, m, \cdot)$ . This leads to  $\Pr[H(\overline{pk}, m, \overline{s}) = y] = \frac{1}{|\mathcal{Y}|}$  and therefore we have:

$$\begin{aligned} \Pr[G_4 = 1] &= \frac{1}{|\mathcal{Y}|} + \sum_{i=1}^{q_0} \Pr[G_4 = 1 \wedge (\overline{pk}, m, \overline{s}) = (pk^i, m^i, s^i)] \\ &= \frac{1}{|\mathcal{Y}|} + \sum_{i=1}^{q_0} \Pr[G_5^i = 1]. \end{aligned}$$

**$G_5^i$  to  $G_6^i$ :** The only difference between game  $G_5^i$  and  $G_6^i$  is, that we can exchange  $m$  with  $m^i$ , which does not make a difference, as  $m = m^i$  was required in  $G_5^i$ . This also allows us to drop the requirement that  $(\overline{pk}, m, \overline{s}) = (pk^i, m^i, s^i)$  while only increasing the winning probability, which leads to:

$$\Pr[G_5^i = 1] \leq \Pr[G_6^i = 1].$$

**$G_6^i$  winning probability is small:** For giving an explicit upper bound for winning in  $G_6^i$  we can again assume, without loss of generality, that all queries of  $\mathcal{A}_0^H$  are distinct. Additionally we assume, that  $\mathcal{A}_0^H$  decided to stop and output a message before the  $i$ th query. Due to this, the input of  $\mathcal{A}_1^H$  is independent of  $H(pk^i, m^i, s^i)$ , which leads to  $\mathcal{A}_1^H$ 's output being independent as well, because  $\mathcal{A}_1^H$  cannot query for  $(\cdot, m^i, \cdot)$ . This means  $\mathcal{A}_1^H$ 's best strategy is to guess, so

$$\Pr[G_6^i = 1] \leq \frac{1}{|\mathcal{Y}|}.$$

**Final Advantage Calculation:** When we sum up the differences between the games we get:

$$\Pr[G_0 = 1] \leq q_0 \cdot 2^{-l} + 2q_1 \cdot \varepsilon + \frac{q_0 + 1}{|\mathcal{Y}|}.$$

As  $G_0$  is the weak- $\Phi$ -salted-non-malleability game and the assumptions made on  $\mathcal{A}^H$  imply no loss of generality, we can conclude:

$$\mathbf{Adv}_H^{\Phi\text{SNM}^{H,\perp}(pk)}(\lambda, \mathcal{A}^H, \mathbf{aux}^H) \leq q_0 \cdot 2^{-l} + 2q_1 \cdot \varepsilon + \frac{q_0 + 1}{|\mathcal{Y}|}.$$

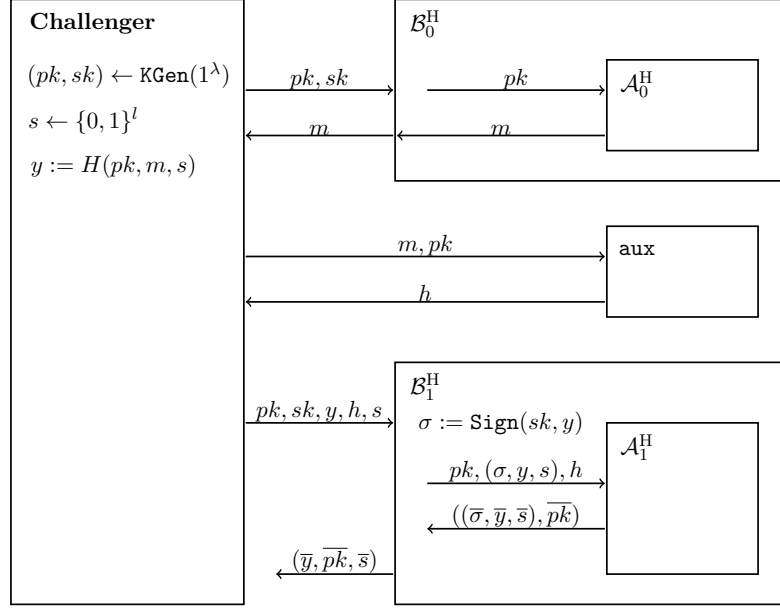
□

After proving this corollary, we can use this result to finally prove theorem 3.17.

*Proof.* Let  $S = (\text{KGen}, \text{Sign}, \text{Vrfy})$  be any EUF-CMA secure signature scheme,  $H$  a random oracle and  $S^{H*} = (\text{KGen}^H, \text{Sign}^H, \text{Vrfy}^H)$  be the signature scheme produced by applying the salted-BUFF-Transform from fig. 3.4.

We will now show, that weak non-resignability of the salted-BUFF-Transform would imply breaking weak- $\Phi$ -salted-non-malleability of the random oracle.

Let  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  be any adversary playing the weak non-resignability game and  $\mathbf{aux}^H$  any auxiliary function. We can now construct an adversary  $\mathcal{B} = (\mathcal{B}_0^H, \mathcal{B}_1^H)$  against weak- $\Phi$ -salted-non-malleability as follows:



$\mathcal{B}^H$  is an efficient adversary, if  $\mathcal{A}^H$  is efficient as well, as  $\mathcal{B}^H$  only forwards messages and generates one signature.

$\mathcal{B}^H$  simulates the weak-NR game for  $\mathcal{A}^H$ . We can directly see, that  $\mathcal{A}_0^H$  gets the desired input from  $\mathcal{B}_0^H$  and its output together with the public key  $pk$  gets then fed into  $\text{aux}^H$ , as expected in the weak NR game.  $\mathcal{A}_1^H$  expects  $\text{Sign}^H(sk, m)$  as the signature and  $\mathcal{B}_1^H$  gives it  $(\text{Sign}(sk, y), y, s) = (\text{Sign}(sk, H(m, pk, s)), y, s) = \text{Sign}^H(sk, m)$ . The public key  $pk$  and the hint  $h$  get forwarded without alteration.

If  $\mathcal{A}^H$  wins in this game, the following holds for its output  $((\bar{\sigma}, \bar{y}, \bar{s}), \bar{pk})$ :  $\bar{pk} \neq pk$  and  $\text{Vrfy}^H(\bar{pk}, m, (\bar{\sigma}, \bar{y}, \bar{s})) = \text{accept}$ . When expanding the definition of  $\text{Vrfy}^H$  we get:  $H(m, \bar{pk}, \bar{s}) = \bar{y}$  and  $\text{Vrfy}(pk, \bar{y}, \bar{\sigma}) = \text{accept}$ .

$\mathcal{B}_1^H$  uses the output of  $\mathcal{A}_1^H$  to output:  $(\bar{y}, \bar{pk}, \bar{s})$ . We require for  $\mathcal{B}^H$ 's output that  $H(m, \bar{pk}, \bar{s}) = \bar{y}$  and  $\bar{pk} \neq pk$ . Both requirements are fulfilled, if  $\mathcal{A}^H$  wins in its weak non malleability game, as discussed above.

So we get, that  $\mathcal{B}^H$  wins, if  $\mathcal{A}^H$  wins which translates to this advantage estimation:

$$\text{Adv}_{\mathcal{S}^H}^{\text{NR}^H, \perp}(\lambda, \mathcal{A}^H, \text{aux}) \leq \text{Adv}_H^{\Phi\text{NM}^H, \perp(pk)}(\lambda, \mathcal{B}^H, \text{aux}^H).$$

As we have shown in lemma 3.19 that the following inequality must hold for any possible  $\mathcal{B}^H$ , where  $q_0$  and  $q_1$  are the number of queries to the oracle of  $\mathcal{B}_0^H$  and  $\mathcal{B}_1^H$  respectively,  $\varepsilon = \text{guess}_{m \leftarrow \mathcal{A}^H}(m \mid H, \text{aux}(m, pk))$  and  $\mathcal{Y}$  is the output space of the random oracle:

$$\text{Adv}_H^{\Phi\text{NM}^H, \perp(pk)}(\lambda, \mathcal{B}^H, \text{aux}^H) \leq q_0 \cdot 2^{-l} + 2q_1 \cdot \varepsilon + \frac{q_0 + 1}{|\mathcal{Y}|}.$$

As  $\mathcal{A}^H$  is an efficient adversary, it can only make polynomial many queries to the oracle, so  $q_0$  and  $q_1$  are poly bounded. We required for the weak NR game that the statistical min entropy of  $m$  conditioned on  $pk, \text{aux}(m, pk), H$  is at least  $\omega(\log \lambda)$  which translates to the guessing probability of  $m$  given  $H$  and  $\text{aux}(m, pk)$  being negligible. The output space of the oracle must be super-poly, so the upper bound given above must be negligible and we get:

$$\text{Adv}_{\mathcal{S}^H}^{\text{NR}^H, \perp}(\lambda, \mathcal{A}^H, \text{aux}) \leq \text{negl}(\lambda)$$

which means, that the salted-BUFF-Transform yields weak non-resignability, as otherwise we can break the weak- $\Phi$ -salted-non-malleability of the random oracle.  $\square$

To conclude this, we have now proven, that there exists signature scheme transformation, that yields a signature scheme providing EUF-CMA, MBS, M-S-UEO and weak-NR in the ROM only requiring an EUF-CMA secure signature scheme and any non malleable hash function.



## 4 Applying the Salted-BUFF-Transform

In this chapter we will analyze, if there are real world signature schemes which used the BUFF transform, where the findings of [DFHS24] could be applied.

Since numerous signature schemes claim to use the BUFF-Transform<sup>1</sup>, we restrict our focus to two signature schemes selected in the NIST Post-Quantum Cryptography Standardization process: CRYSTALS-DILITHIUM ([DKL+18]) and FALCON ([FHK+20]).

We will not look into SPHINCS<sup>+</sup>, which was also selected in the NIST Post-Quantum Cryptography Standardization as a signature scheme and specified in [C+24]. SPHINCS<sup>+</sup> is a hash based signature scheme. This type of signature schemes work completely different from established signature schemes, as their hardness is not based on the hardness of solving a mathematical problem, but instead based on the hardness of finding collisions and pre-images. [CDF+21] conjectured, that SPHINCS<sup>+</sup> yields NR, without needing to apply the (salted-)BUFF-Transform.

### 4.1 Motivation for Post-Quantum Signature Schemes

Most signature schemes in widespread use today, including RSA, DSA, and ECDSA, derive their security from the presumed hardness of problems such as integer factorization and the discrete logarithm problem. While these assumptions are considered secure against classical adversaries, they are rendered ineffective in the presence of a sufficiently large quantum computer. In particular, Shor’s algorithm, introduced in [Sho94], can solve both, integer factorization and discrete logarithms in polynomial time on quantum computers, thereby enabling the efficient computation of private keys being given the corresponding public keys. As a result, any cryptographic system relying on these assumptions would be broken, if a sufficiently potent enough quantum computer would be available.

### 4.2 Introduction to Lattice Cryptography

The above mentioned DILITHIUM and FALCON signature schemes are based on so called lattice cryptography. In this section we will introduce the basics of lattice cryptography, to understand later DILITHIUM and FALCON. This section is based on [Sil06].

---

<sup>1</sup>[DFHS24] alone lists seven such schemes submitted to the NIST Post-Quantum Cryptography Standardization contest that did not become finalists.

**Definition 4.1** (Lattice) *A lattice  $L$  of dimension  $n$  is the  $\mathbb{Z}$ -linear span of a set of  $n$  linearly independent vectors  $v_1, \dots, v_n$ :*

$$L = \{\alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n\}.$$

*The Vectors  $v_1, \dots, v_n$  are called the basis of  $L$ .*

By definition every lattice  $L$  of dimension  $n$  is a subgroup of the vector space  $\mathbb{R}^n$ .

### 4.2.1 apprCVP based Lattice Cryptography

Like the factorization problem, which RSA is based on, in classical rings is hard, we can also find hard problems on lattices too. Some commonly used and easy to understand hard problems are the shortest vector problem (SVP), the closest vector problem (CVP) and its approximation problem variant apprCVP.

**Definition 4.2** (Shortest Vector Problem) *The shortest vector problem (SVP) is, as the name implies, the problem to find the shortest non-zero vector  $v$  in some given lattice  $L$ , or formally: Find a vector  $v \in L$  s.t.  $\nexists v' \in L : 0 < \|v'\|_2 < \|v\|_2$ .*

**Definition 4.3** (Closest Vector Problem) *The closest vector problem (CVP) is to find for a given lattice  $L$  and vector  $t \in \mathbb{R}^n \setminus L$  the vector in  $L$  that is closest to  $t$ , which is formally: Find a vector  $v \in L$  s.t.  $\nexists v' \in L : \|v' - t\|_2 < \|v - t\|_2$ .*

CVP is shown to be NP-hard, SVP is NP-hard under a randomized reduction hypothesis.

We can also define an approximation problem for CVP, where we only require the solution to be worse by a fixed factor  $\kappa$ , than the optimal solution of the CVP.

**Definition 4.4** (Approximate Closest Vector Problem) *The approximate closest vector problem (apprCVP) is to find for a given lattice  $L$ , approximation factor  $\kappa$  and vector  $t \in \mathbb{R}^n \setminus L$  a vector  $v \in L$  s.t.  $\|v - t\|_2 \leq \kappa \min\{\|w - t\|_2 \mid w \in L\}$ .*

When trying to solve apprCVP for some different basis of a Lattice, you will find, that there are basis which are easier and some which are harder to solve. Basis, where it is easy to find a solution for apprCVP we will call “good” basis and for basis where it is hard, we will speak of “bad” basis.

When try to quantify, when a basis is good, we will get, that a basis is better, if its base-vectors are on the one hand closer to being orthogonal to each other and on the other hand shorter. Formalizing these requirements is complicated and an intuition is sufficient for our goals.

There are algorithms, that can transform any basis into a good basis. The best known algorithm for getting a good basis has exponential runtime. There are approximation algorithms, that can transform a basis into a basis that is by a constant factor worse than a good basis, most prominently the LLL algorithm, which finds moderately short and moderately orthogonal basis (up to a factor  $4/3$  less than the optimal solution) in polynomial time.

Based on the problems we described above, multiple cryptographic systems were defined, e.g. GGH (Goldreich, Goldwasser and Halevi). GGH’s security is based on CVP being hard to solve in non-orthogonal (which means bad) basis. But as LLL (and its derivatives) can transform a bad basis into more orthogonal basis efficiently, basis of dimension 500 or more are needed, to get a secure system. When encoding a basis vector with an 32-bit integer this leads to keys with size of at least 250MB.

Additionally the encryption system NTRUENCRYPT was proposed, from which NTRUSIGN was later derived in [HHGP+03]<sup>2</sup>. NTRUENCRYPT and NTRUSIGN are both based on so called NTRU lattices. These are lattices whose basis can be encoded in  $\frac{1}{2}n \log(n)$  bits, so even large basis, where the application of LLL is not practical, can be encoded efficiently.

### NTRUSign

To define NTRUSIGN correctly, we first need to recall polynomial rings and the multiplication and addition in these rings.

Let  $R := \mathbb{Z}[X]/(X^N - 1)$ . We call  $R$  a convolutional polynomial ring. Its elements have the form  $a_0 + a_1X + a_2X^2 + \dots + a_{N-1}X^{N-1}$  with  $a_i \in \mathbb{Z}$ .

For two polynomials  $f = a_0 + a_1X + a_2X^2 + \dots + a_{N-1}X^{N-1}$  and  $g = b_0 + b_1X + b_2X^2 + \dots + b_{N-1}X^{N-1}$  the addition is given by

$$f + g = a_0 + b_0 + (a_1 + b_1)X + (a_2 + b_2)X^2 + \dots + (a_{N-1} + b_{N-1})X^{N-1}$$

and the multiplication is given by

$$f \cdot g = \sum_{k=0}^{N-1} \left( \left( \sum_{i=0}^{N-1} a_i b_{k-i} \pmod{(N-1)} \right) X^k \right).$$

For every element  $R := \mathbb{Z}[X]/(X^N - 1)$  we can construct a lattice  $L$  with the following function:

$$M : R \rightarrow L, a_0 + a_1X + a_2X^2 + \dots + a_{N-1}X^{N-1} \mapsto \begin{pmatrix} a_0 & a_1 & \dots & a_{N-1} \\ a_{N-1} & a_0 & \dots & a_{N-2} \\ \vdots & & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 \end{pmatrix}$$

where the Lattice is given by the rows of the matrix as the basis of  $L$ .

We can use this to define a simplified version of NTRUSIGN based on [HHGP+03] and [GS02].

**Definition 4.5** (NTRUSIGN) *We define NTRUSIGN with the key generation, signing and verification for some hash function  $H$  which maps to  $R_q \times R_q$  (as defined in the key generation) as follows:*

**Key Generation** *On input  $N, p, q, B \in \mathbb{N}^+$ , where  $N$  is prime, and  $p$  and  $q$  are co-prime. The key generation algorithm works as follows.*

- Set  $R := \mathbb{Z}[X]/(X^N - 1)$  and  $R_q := \mathbb{Z}_q[X]/(X^N - 1)$ .
- Choose  $u, f_1, g_1 \in R$  with coefficients in  $\{0, 1, -1\}$  s.t.  $f, g$  from the next step are short.<sup>3</sup>

<sup>2</sup>[HHGP+03] is a newer revision of the original paper published in 2001 which was withdrawn due to an attack against the unforgeability shown in [GS02]. We were not able to find a copy of the original version.

<sup>3</sup>In the non-simplified version the number of coefficients with each value are specified. Having a larger number of 0 will make sure, that  $f$  and  $g$  will be short. Refer to [HHGP+03] for details.

- Set  $f = u + pf_1$  and  $g = u + pg_1$ .
- Find short  $F, G$  s.t.  $f \cdot G - g \cdot F = q$ ,  $\|F\|_2 \geq 2 \cdot \|f\|_2$  and  $\|G\|_2 \geq 2 \cdot \|g\|_2$ .
- Set  $h = f^{-1} \cdot g \in R_q$ . Restart the algorithm if  $f$  is not invertible.
- Set  $pk = h$  and  $sk = (f, g, F, G)$ .

As usual  $pk$  is the public key and  $sk$  is the secret key. The input parameters for the signature algorithm are expected to be known publicly.

**Signing** The algorithm for signing a message  $m$  for a given secret key  $sk = (f, g, F, G)$  works as follows:

- Calculate  $(m_1, m_2) = H(m)$ .
- Find  $A, C, a, c \in R_q$  s.t.:

$$\begin{aligned} G \cdot m_1 + F \cdot m_2 &= A + q \cdot C \\ -g \cdot m_1 + f \cdot m_2 &= a + q \cdot c. \end{aligned}$$

- Set  $s = f \cdot C + F \cdot c \in R_q$ .

**Verification** The algorithm for verifying a signature  $s$  of a message  $m$  under public key  $pk = h$  works as follows:

- Calculate  $(m_1, m_2) = H(m)$  and  $t = s \cdot h \in R_q$ .
- Check if  $\|s - m_1\|_2^2 + \|t - m_2\|_2^2 \leq B$ . Output **accept** if it holds, otherwise **reject**.

An intuition for the correctness can be given by examining the construction of  $s$ :

$$\begin{aligned} s &= f \cdot C + F \cdot c \\ &= f \cdot (G \cdot m_1 + F \cdot m_2 - A) \cdot q^{-1} + F \cdot (-g \cdot m_1 + f \cdot m_2 - a) \cdot q^{-1} \\ &= (f \cdot (G \cdot m_1 + F \cdot m_2 - A) + F \cdot (-g \cdot m_1 + f \cdot m_2 - a)) \cdot q^{-1} \\ &= (f \cdot G \cdot m_1 + f \cdot F \cdot m_2 - f \cdot A + F \cdot -g \cdot m_1 + F \cdot f \cdot m_2 - F \cdot a) \cdot q^{-1} \\ &= \underbrace{((f \cdot G - F \cdot g) \cdot m_1)}_{=q} + \underbrace{2 \cdot f \cdot F \cdot m_2}_{=0} - f \cdot A - F \cdot a) \cdot q^{-1} \\ &= m_1 - (f \cdot A + F \cdot a) \cdot q^{-1} \end{aligned}$$

Thus the difference between  $s$  and  $m_1$  is determined by  $(f \cdot A + F \cdot a)$  i.e.  $\|s - m_1\|_2 = \|f \cdot A + F \cdot a\|_2$ . As  $F$  and  $f$  were chosen such, that their norm is small,  $\|f \cdot A + F \cdot a\|_2$  will be small as well.

There is an attack against the security of this version of NTRUSIGN, as NTRUSIGN leaks some information about the private key when given a signature and its message. This is because the mapping  $m \rightarrow s$  in the signature algorithm is not a permutation, which allows passive attacks to extract the secret key. This was later fixed, by adding some additional randomness (called ‘‘perturbation’’) in the signing process, making the extraction much harder. This is described in [HHGP<sup>+</sup>03]. We did not cover this, as it makes the algorithm more complex and the correctness argument even more complicated.

### 4.2.2 LWE based Lattice Cryptography

Learning with Errors (LWE) is another problem, which we can use to build cryptographic primitives from and it was introduced in [Reg09]. Learning with errors is a problem, that can be reduced to SVP, i.e. LWE is hard to solve, as long SVP is hard to solve.

First we will look into learning without errors. Suppose you have the following linear equation:

$$Ax = b.$$

Given  $A$  and  $b$ , the goal is to recover  $x$ . This can be solved easily, by applying Gaussian elimination, so this problem is not suitable, to build cryptography upon.

Lets look at a slight variation of this problem, again we have a matrix  $A$ , vectors  $x$  and  $b$  and an additional vector  $e$  called noise vector, that fulfills the following equation:

$$Ax + e = b.$$

When given  $A$  and  $b$ , the goal is again, to recover  $x$ . As there are infinitely many solutions, depending on the concrete value of  $e$ , this is impossible to do. This principle is called learning with errors.

There are two widely known variants of LWE, namely RLWE and MLWE. RLWE is the same problem, as LWE, but the used vector spaces are defined over rings and MLWE, where the vector spaces are defined over polynomial modular rings.

Using this construction, we can build our own signature scheme, based on GLYPH as introduced in [Cho17], over the ring  $R_q = \mathbb{Z}_q[X]/(X^N + 1)$  with a hash function  $H$  that maps to  $R_q$  as follows:

**KGen** Generate random polynomials  $a, x, e \in R_q$  and calculate  $t = a \cdot x + e$ . Set the public key  $pk = (t, a)$  and secret key  $sk = (x, e)$ .

**Sign** To sign a message  $m$  generate random polynomials  $y_1, y_2 \in R_q$  and calculate:  
 $w = a \cdot y_1 + y_2$ .  
 Calculate  $c = H(w, m)$ ,  $z_1 = x \cdot c + y_1$  and  $z_2 = e \cdot c + y_2$ . Output  $(c, z_1, z_2)$  as the signature.

**Vrfy** Given  $(c, z_1, z_2)$  compute  $w' = a \cdot z_1 + z_2 - t \cdot c$ . Check if  $H(w', m) = c$ , output **accept** if yes, otherwise **reject**.

This signature scheme is correct, as we can show, that  $w$  and  $w'$  are the same:

$$\begin{aligned} w' &= a \cdot z_1 + z_2 - t \cdot c \\ &= a \cdot (x \cdot c + y_1) + z_2 - (a \cdot x + e) \cdot c \\ &= a \cdot y_1 + z_2 - e \cdot c \\ &= a \cdot y_1 + e \cdot c + y_2 - e \cdot c \\ &= a \cdot y_1 + y_2 \\ &= w \end{aligned}$$

For this scheme to be secure, we need to make our random choices carefully i.e. use special distributions and require the coefficients to be limited. Also in GLYPH it is

required, that the coefficients in  $z_1$  and  $z_2$  are small. For the sake of simplicity, we removed this requirement.

As we used module rings instead of vector spaces for this, our signature scheme is based on MLWE.

### 4.3 FALCON

FALCON is an interesting signature scheme for analysis in the light of the findings of [DFHS24], we discussed in section 3.2.2, even though FALCON does not achieve non-resignability. This is because [DFF25] gave two derivations of FALCON, called FALCON-PS-3 and FALCON-BUFF, which both yield weak non-resignability, while considering the findings in [DFHS24]. FALCON-PS-3 is the application of the third Pornin-Stern transform (PS-3) introduced in [PS05], while the FALCON-BUFF is based on the BUFF-Transform. The third Pornin-Stern transform, is related to the BUFF-Transform. The only difference between them is that the signature in the PS-3 transform does not additionally contain the hash of  $m$  and  $pk$ . They however only did analyze the FALCON-PS-3, as it fulfilled all the security properties, the BUFF-Transform would also achieve, while maintaining the signature size of FALCON, as the hash does not get added to the signature.

FALCON and its derivatives are based on NTRUSIGN which we already discussed.

#### 4.3.1 GPV Framework

To understand FALCON and its derivatives, we first need to discuss the GPV (Gentry-Peikert-Vaikuntanathan) framework originally introduced in [GPV08]. We present explanations based on [FHK<sup>+</sup>20]. The GPV framework is a way to construct signature schemes based on the hardness of SVP.

The GPV framework works on a high abstraction level like this:

**KGen** Generate a matrix  $A \in \mathbb{Z}_q^{n \times m}$  with full rank (i.e. all row vectors are linearly independent). Understand the row vectors as the basis for a lattice  $\Lambda$ .  
 Generate a matrix  $B \in \mathbb{Z}_q^{m \times m}$  where the row vectors are a basis for a lattice  $\Lambda^\perp$ .  
 Choose  $B$  s.t.  $\Lambda$  and  $\Lambda^\perp$  are orthogonal, i.e.  $\forall x \in \Lambda, y \in \Lambda^\perp : \langle x, y \rangle = 0 \pmod q$ .  
 $A$  is the public key and  $B$  the secret key.

**Sign** To sign a message  $M$ , first calculate a  $c_0 \in \mathbb{Z}_q^m$  satisfying  $c_0 A^\top = H(M)$  using basic linear algebra. Use  $B$  to compute a vector  $v \in \Lambda^\perp$  close to  $c_0$ . Set  $s = c_0 - v$  as the signature.

**Vrfy** To verify a signature  $s$ , check that  $s A^\top = H(M)$  holds and that  $s$  is short.

This works, because of  $s = c_0 - v$ , we have  $s A^\top = c_0 A^\top - v A^\top$  and as  $v \in \Lambda^\perp$ ,  $v A^\top = 0$  (as  $v$  is orthogonal to every row in  $A$ ) and thus:  $s A^\top = c_0 A^\top - v A^\top = c_0 A^\top - 0 = c_0 A^\top$ . And by our choosing of  $c_0$  fulfilling  $c_0 A^\top = H(M)$ , we get  $s A^\top = H(M)$ . Also as we chose  $c_0$  and  $v$  to be close,  $s = c_0 - v$  must be short as well.

The framework was proven to be secure if the SIS (Short Integer Solution) problem is hard. The SIS problem is related to SVP, we already discussed. In the SIS problem, we want that for a randomized matrix  $A$  it is hard to find a short non zero vector  $x$  that fulfills  $Ax = 0$ .

```

1: FALCON-KGEN( $N, p, q$ ):
2:    $R := \mathbb{Z}[X]/(X^N + 1)$ 
3:    $R_q := \mathbb{Z}_q[X]/(X^N + 1)$ 
4:    $u, f_1, g_1 \leftarrow R$  with coefficients in  $\{0, 1, -1\}$  s.t.  $f, g$  are short
5:    $f := u + pf_1$ 
6:    $g := u + pg_1$ 
7:    $F, G \leftarrow R$  s.t.  $f \cdot G - g \cdot F = q$ ,  $F, G$  are short, and  $\|F\|_2 \geq 2 \cdot \|f\|_2$  and
    $\|G\|_2 \geq 2 \cdot \|g\|_2$ 
8:    $pk := f^{-1} \cdot g \in R_q$ , restart if  $f$  is not invertible
9:    $sk := \begin{pmatrix} g & -f \\ G & -F \end{pmatrix}$ 
10:  return  $sk, pk$ 

1: FALCON-SIGN( $sk, pk, m$ ):
2:    $r \leftarrow \{0, 1\}^{320}$ 
3:    $c \leftarrow H(r, m)$ 
4:   repeat
5:      $\begin{pmatrix} s_1 & s_2 \end{pmatrix} \leftarrow \text{PreSmp}_{pk}(sk, c)$ 
6:   until  $\left\| \begin{pmatrix} s_1 & s_2 \end{pmatrix} \right\|_2^2 \leq \lfloor \beta^2 \rfloor$ 
7:   return  $\sigma := (r, s_2)$ 

1: FALCON-VERIFY( $pk, m, \sigma$ ):
2:    $(r, s_2) \leftarrow \sigma$ 
3:    $c \leftarrow H(r, m)$ 
4:    $s_1 \leftarrow c - s_2 \cdot pk$ 
5:   return  $\left\| \begin{pmatrix} s_1 & s_2 \end{pmatrix} \right\|_2^2 \leq \lfloor \beta^2 \rfloor$ 

```

Figure 4.1: Simplified version of the FALCON signature scheme based on [FHK<sup>+</sup>20] and [GJK24]

### 4.3.2 Simplified FALCON Signature Scheme

After introducing GPV we can now define a simplified version of the FALCON signature scheme. The original version was introduced in [FHK<sup>+</sup>20].

The original version is optimized to have a fast running time, we will present a simplified version, based on [GJK24], that has the same working principle, but removed all steps that were introduced to make the scheme more efficient. Our version is given in fig. 4.1.

FALCON-KGen works almost the same as the key generation algorithm of NTRUSign. The largest difference is, that we use  $X^N + 1$  instead of  $X^N - 1$  as a modulus for the ring. Using  $X^N + 1$  has the advantage that, for  $N$  being a power of two,  $X^N + 1$  is irreducible. Additionally the secret key is now represented as a matrix instead of a tuple. The matrix is chosen, so we can use matrix multiplication for the mathematical primitives in the signing algorithm.

In FALCON-Sign, we follow the well-known hash-then-sign paradigm, using a hash function  $H$ , instantiated with SHAKE-256 from the SHA-3 family standardized in [D<sup>+</sup>15], modified so that its output lies in  $R_q$ .

The algorithm  $\text{PreSmp}_{pk}$  is an algorithm, that takes an input  $sk, c$  and outputs  $s \in R^2$  s.t.  $pk \cdot s = (c, 0)$  and  $s$  follows a Gaussian distribution over  $sk, c$ .

**Theorem 4.6** (Correctness of FALCON) *The FALCON signature scheme given in fig. 4.1 is correct.*

This follows directly, as the signature is  $(r, s_2)$  and the verifier is given  $pk$  so it can reconstruct  $c = H(r, m)$  and  $s_1$ . If the parameters were constructed by the signing

algorithm the condition:

$$\left\| \begin{pmatrix} s_1 & s_2 \end{pmatrix} \right\|_2^2 \leq \lfloor \beta^2 \rfloor$$

must hold, due to the condition in line 6 of the signing algorithm.

Also, the non-simplified version of FALCON is shown to be UF-CMA secure, a slightly weaker variant of EUF-CMA, and provides MBS. M-S-UEO and (w)NR are not yielded by FALCON.

Intuitively, FALCON is secure, as generating  $s_1, s_2$  s.t. the norm is small and  $s_1, s_2 \in R$  is hard, without knowledge of the secret key, as the secret key is a good basis for  $R$ .

### 4.3.3 FALCON-PS-3

Because the additional security properties, the BUFF-Transform yields, are highly desired, [DF25] showed, that FALCON can be altered to provide wNR and MBS. For this, they altered FALCON as shown in fig. 4.2 and called their new construction FALCON-PS-3.

The only difference between FALCON and FALCON-PS-3 is, that the hash used for the signature process, additionally contains the public key. Due to this being the only change, the UF-CMA security is preserved.

<pre> 1: FALCON-PS-3-SIGN(sk, pk, m): 2:   r ← {0, 1}^320 3:   c ← H(r, <span style="background-color: #e0f0ff;">pk</span>, m) 4:   <b>repeat</b> 5:     ( s<sub>1</sub> s<sub>2</sub> ) ← PreSmp<sub>pk</sub>(sk, c) 6:   <b>until</b> <math>\left\  \begin{pmatrix} s_1 &amp; s_2 \end{pmatrix} \right\ _2^2 \leq \lfloor \beta^2 \rfloor</math> 7:   <b>return</b> σ := (r, s<sub>2</sub>)                 </pre>	<pre> 1: FALCON-PS-3-VERIFY(pk, m, σ): 2:   (r, s<sub>2</sub>) ← σ 3:   c ← H(r, <span style="background-color: #e0f0ff;">pk</span>, m) 4:   s<sub>1</sub> ← c - s<sub>2</sub> · pk 5:   <b>return</b> <math>\left\  \begin{pmatrix} s_1 &amp; s_2 \end{pmatrix} \right\ _2^2 \leq \lfloor \beta^2 \rfloor</math>                 </pre>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 4.2: Simplified version of the FALCON-PS-3 signature scheme based on [DF25].

The key generation algorithm is not presented, as it is the same as FALCON-KGen. Differences to FALCON are marked blue.

To prove, that FALCON-PS-3 yields weak non-resignability we first need to define two additional notions.

**Definition 4.7** (Unpredictability) *For signature scheme with key generation algorithm  $KGen^H$ , an adversary  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  and auxiliary function  $\mathbf{aux}$ , where  $H$  is a random oracle, which  $KGen^H$  and  $\mathcal{A}^H$  has access to, we define the unpredictability game as follows:*

```

(pk, sk) ← KGenH(1λ)
m ← A0H(pk)
h := aux(m, pk)
m' ← A1H(pk, sk, h)
                
```

*We say  $\mathcal{A}^H$  wins, if  $m = m'$ .*

*The advantage is defined as follows:*

$$\mathbf{Adv}_{H, KGen^H, \mathcal{A}_0^H, \mathbf{aux}}^{\text{pred}}(\lambda, \mathcal{A}_1^H) = \Pr[\mathcal{A}^H \text{ wins}].$$

*We say a message is unpredictable given  $\mathcal{A}_0^H$  and  $\mathbf{aux}$ , if for every probabilistic polynomial time adversary  $\mathcal{A}_1^H$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for*

every  $\lambda \in \mathbb{N}^+$ :

$$\mathbf{Adv}_{H, KGen^H, \mathcal{A}_0^H, \mathbf{aux}}^{pred}(\lambda, \mathcal{A}_1^H) \leq \mu(\lambda).$$

**Definition 4.8** (Computational Indistinguishability of Auxiliary Data) *For signature scheme with key generation algorithm  $KGen^H$ , an adversary  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  and auxiliary function  $\mathbf{aux}$ , where  $H$  is a random oracle, which  $KGen^H$  and  $\mathcal{A}^H$  has access to, we define the computational indistinguishability of auxiliary data experiment  $b \in \{0, 1\}$  as follows:*

$$\begin{aligned} (pk, sk) &\leftarrow KGen^H(1^\lambda) \\ m_0 &\leftarrow \mathcal{A}_0^H(pk) \\ m_1 &\leftarrow \mathcal{A}_0^H(pk) \\ h &:= \mathbf{aux}(m_b, pk) \\ b' &\leftarrow \mathcal{A}_1^H(pk, sk, m_0, h) \end{aligned}$$

We define the event  $W_b$  as  $\mathcal{A}_1^H$  outputs 1 in experiment  $b$ .

We define the advantage as follows:

$$\mathbf{Adv}_{H, KGen^H, \mathbf{aux}}^{CIA}(\lambda, \mathcal{A}^H) = |Pr[W_0] - Pr[W_1]|.$$

We say an auxiliary data generated by  $\mathbf{aux}$  is indistinguishable, if for every probabilistic polynomial time adversary  $\mathcal{A}^H$  there exists a negligible function  $\mu : \mathbb{N}^+ \rightarrow \mathbb{R}$  such that for every  $\lambda \in \mathbb{N}^+$ :

$$\mathbf{Adv}_{H, KGen^H, \mathbf{aux}}^{CIA}(\lambda, \mathcal{A}^H) \leq \mu(\lambda).$$

**Theorem 4.9** (Weak Non-Resignability of FALCON-PS-3) *Let  $H$  be a random oracle and let  $KGen$  be the key generation,  $Sign^H$  the signing algorithm and  $Vrfy^H$  the verification algorithm of FALCON-PS-3. Let  $\mathcal{A}^H = (\mathcal{A}_0^H, \mathcal{A}_1^H)$  be a probabilistic polynomial time  $NR^{H, \perp}$  adversary, where  $\mathcal{A}_0^H$  and  $\mathcal{A}_1^H$  make at most  $q_0, q_1$  queries to  $H$  respectively and  $\mathbf{aux}$  a (possibly randomized) auxiliary function, then the weak non-resignability advantage is bounded as follows:*

$$\begin{aligned} \mathbf{Adv}_{FALCON-PS-3}^{NR^{H, \perp}}(\mathcal{A}, \mathbf{aux}) &\leq \mathbf{Adv}_{H, KGen^H, \mathbf{aux}}^{CIA}(\lambda, \mathcal{B}^H) \\ &\quad + (q_0 + q_1) \cdot \mathbf{Adv}_{H, KGen^H, \mathcal{C}_0^H, \mathbf{aux}}^{pred}(\lambda, \mathcal{C}_1^H) \\ &\quad + (q_0 + q_1)2^{(5-k)\frac{n}{2}}. \end{aligned}$$

*Proof.* To prove this, we will use game hopping with the games defined in fig. 4.3.

The game  $G_0$  is the standard wNR experiment.

**$G_0$  to  $G_1$ :** The difference between  $G_0$  and  $G_1$  is, that we run  $\mathcal{A}_0^H$  and  $\mathbf{aux}$  in  $G_1$  another time and do not use their output.  $\mathcal{A}_1^H$  cannot detect this, so we have:

$$\Pr[G_0 = 1] = \Pr[G_1 = 1].$$

**$G_1$  to  $G_2$ :** In  $G_2$  we now use the output  $\bar{h}$  of our second run of  $\mathbf{aux}$  as the hint for  $\mathcal{A}_1^H$ . We can construct a new adversary  $\mathcal{B}^H = (\mathcal{B}_0^H, \mathcal{B}_1^H)$  from  $\mathcal{A}^H$  for the CIA-game, which works as follows:

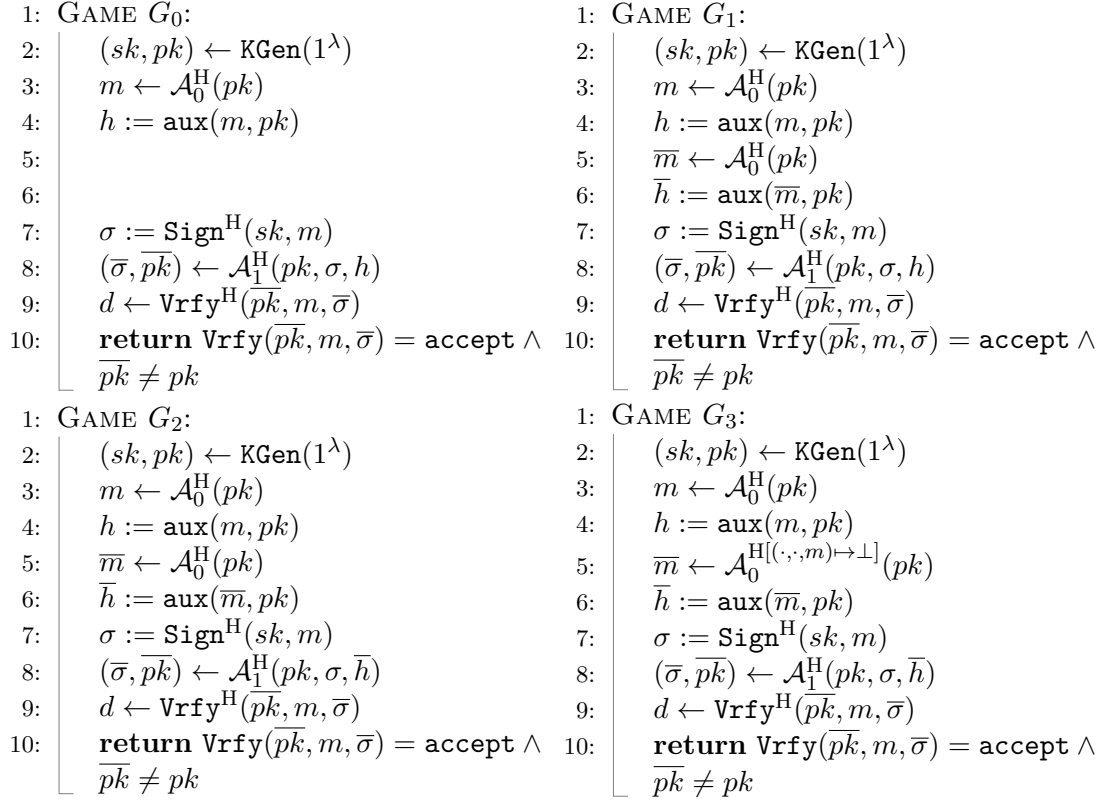
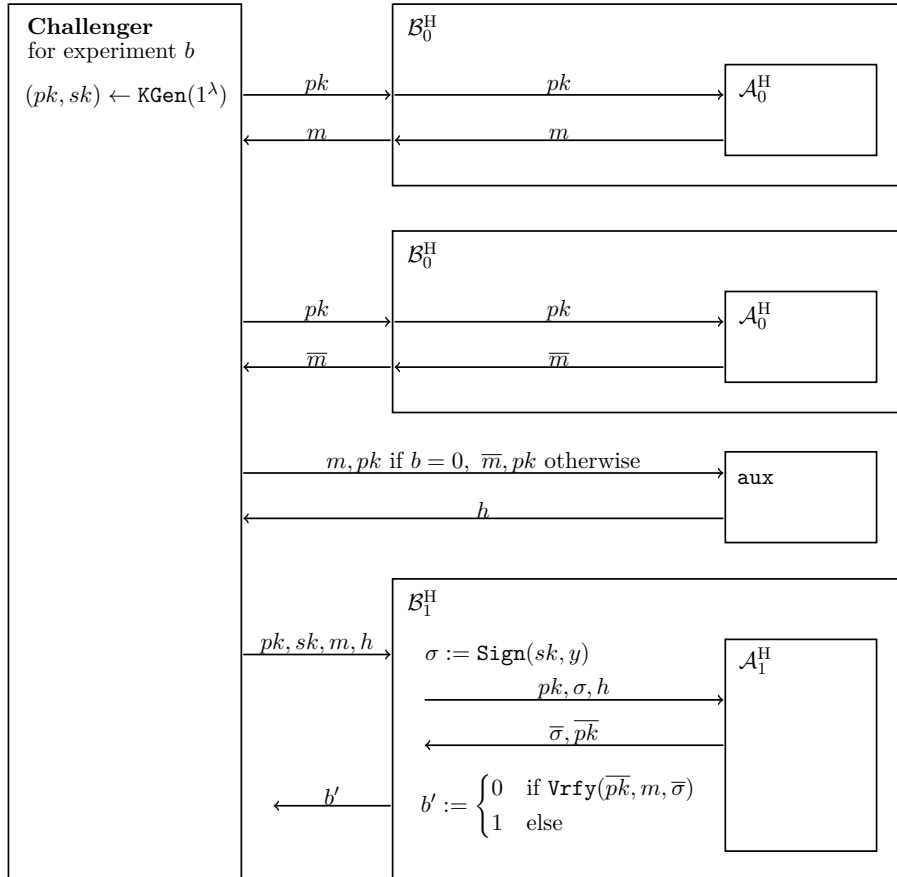


Figure 4.3: The games  $G_0$  to  $G_3$  to prove theorem 4.9 based on the proof of proposition 18 in [DFF25]



We can see that  $\mathcal{B}^H$  simulates  $G_1$  if  $b = 0$  and  $G_2$  otherwise. This means, the probability of  $\mathcal{A}^H$  being able to differentiate between  $G_1$  and  $G_2$  is bound by  $\mathbf{Adv}_{H, \text{KGen}^H, \text{aux}}^{\text{CIA}}(\lambda, \mathcal{B}^H)$ , which yields us:

$$\Pr[G_1 = 1] \leq \Pr[G_2 = 1] + \mathbf{Adv}_{H, \text{KGen}^H, \text{aux}}^{\text{CIA}}(\lambda, \mathcal{B}^H).$$

**$G_2$  to  $G_3$ :** In the switch to  $G_3$ , we alter the oracle to output a special value, if a query for a message  $(\cdot, \cdot, m)$  is made by the second run of  $\mathcal{A}_0^H$  or by  $\mathcal{A}_1^H$ . Using index guessing, we can bound that such a query happens by the prediction advantage:  $\mathbf{Adv}_{H, \text{KGen}^H, \mathcal{C}_0^H, \text{aux}}^{\text{pred}}(\lambda, \mathcal{C}_1^H)$ . As  $\mathcal{A}_0^H$  and  $\mathcal{A}_1^H$  make together at most  $q_0 + q_1$  different queries, this gives us the following advantage bound:

$$\Pr[G_2 = 1] \leq \Pr[G_3 = 1] + (q_0 + q_1) \cdot \mathbf{Adv}_{H, \text{KGen}^H, \mathcal{C}_0^H, \text{aux}}^{\text{pred}}(\lambda, \mathcal{C}_1^H).$$

**$G_3$  winning probability:** In game  $G_3$  the adversary must output a valid signature  $\bar{\sigma}$  under  $\bar{pk}$  for a message  $m$ , it has no information about. We require for the signature  $\bar{\sigma} = (\bar{r}, \bar{s}_2)$  that  $\|\mathbf{H}(\bar{r}, \bar{pk}, m) - \bar{s}_2 \cdot \bar{pk}\|_2$  is bounded by  $\beta$  for  $\bar{\sigma}$  being valid. As the output of the random oracle is uniformly distributed, we get  $\mathbf{H}(\bar{r}, \bar{pk}, m) - \bar{s}_2 \cdot \bar{pk}$  is distributed uniformly as well. For some uniformly chosen value  $x$ ,  $\Pr[\|x\| \leq \beta] \leq 2^{(5-k)\frac{n}{2}}$  holds<sup>4</sup>, leading us to:

$$\Pr[G_3 = 1] \leq 2^{(5-k)\frac{n}{2}}.$$

**Final Advantage Calculation:** When we sum up the differences between the games we get:

$$\begin{aligned} \Pr[G_0 = 1] &\leq \mathbf{Adv}_{H, \text{KGen}^H, \text{aux}}^{\text{CIA}}(\lambda, \mathcal{B}^H) \\ &\quad + (q_0 + q_1) \cdot \mathbf{Adv}_{H, \text{KGen}^H, \mathcal{C}_0^H, \text{aux}}^{\text{pred}}(\lambda, \mathcal{C}_1^H) \\ &\quad + (q_0 + q_1) 2^{(5-k)\frac{n}{2}}. \end{aligned}$$

As  $G_0$  is the wNR game we can conclude:

$$\begin{aligned} \mathbf{Adv}_{\text{FALCON-PS-3}}^{\text{NR}^H, \perp}(\mathcal{A}, \text{aux}) &\leq \mathbf{Adv}_{H, \text{KGen}^H, \text{aux}}^{\text{CIA}}(\lambda, \mathcal{B}^H) \\ &\quad + (q_0 + q_1) \cdot \mathbf{Adv}_{H, \text{KGen}^H, \mathcal{C}_0^H, \text{aux}}^{\text{pred}}(\lambda, \mathcal{C}_1^H) \\ &\quad + (q_0 + q_1) 2^{(5-k)\frac{n}{2}}. \end{aligned}$$

□

The unpredictability and computational indistinguishability notion can be reduced back to the statistical entropy requirement we used in theorem 3.17, which allows us to use the same assumptions for the weak non-resignability of FALCON-PS-3 and the salted-BUFF-Transform. Intuitively, unpredictability and computational indistinguishability are guaranteed by the guessing probability of the message, conditioned on the oracle and hint, being low.

<sup>4</sup>The proof for this statement is not in scope for this thesis. Refer for a proof to lemma 11 of [DF25].

<pre> 1: DILITHIUM-KGEN(<math>N, q, k, l, \eta, \gamma_1</math>): 2:   <math>A \leftarrow R_q^{k \times l}, s_1 \leftarrow S_\eta^l, s_2 \leftarrow S_\eta^k</math> 3:   <math>t = A \cdot s_1 + s_2</math> 4:   <math>sk = (s_1, s_2), pk = (A, t)</math> 5:   <b>return</b> <math>sk, pk</math> </pre>	<pre> 1: DILITHIUM-SIGN(<math>sk, m</math>): 2:   <math>(s_1, s_2) := s</math> 3:   <math>y \leftarrow \tilde{S}_{\gamma_1}^l</math> 4:   <math>w := A \cdot y \in R_q^k</math> 5:   <math>w_1 := \text{HighBits}(w)</math> 6:   <math>c := H(m, w_1)</math> 7:   <math>z := y + c \cdot s_1 \in R_q^l</math> 8:   Restart if <math>\text{LowBits}(w - cs_2)</math> is not    small 9:   <b>return</b> <math>\sigma := (c, z)</math> </pre>
<pre> 1: DILITHIUM-VERIFY(<math>pk, m, \sigma</math>): 2:   <math>(c, z) \leftarrow \sigma</math> 3:   <math>w'_1 = \text{HighBits}(Az - ct)</math> 4:   <b>return</b> <math>c = H(m, w'_1)</math> </pre>	

Figure 4.4: Simplified version of the DILITHIUM signature scheme based on [Men24]

## 4.4 CRYSTALS-Dilithium

CRYSTALS-DILITHIUM (or for short DILITHIUM) is another signature scheme of special interest, as [CDF<sup>+</sup>21] claims it provides non-resignability. DILITHIUM is a post quantum secure signature scheme, that is based on the MLWE problem and was introduced in [DKL<sup>+</sup>18].

Before describing how DILITHIUM works, we need to introduce some notations:

- $R_q := \mathbb{Z}_q / (X^N + 1)$ .
- $\text{mods } q$ : the operation mod  $q$  normalized to output between  $-q/2$  and  $q/2$ .
- $S_\eta = \{a_0 + a_1X + \dots + a_NX^N \mid a_i \in [-\eta, \eta] \cap \mathbb{Z}_q\} \subseteq R_q$ : the set of polynomials with coefficients mod  $q$  between  $-\eta$  and  $\eta$ .
- $\tilde{S}_{\gamma_1} = \{a_0 + a_1X + \dots + a_NX^N \mid a_i \in (-\gamma_1, \gamma_1] \cap \mathbb{Z}_q\} \subseteq R_q$ : the set of polynomials with coefficients mod  $q$  between  $-\gamma_1$  and  $\gamma_1$ , excluding  $-\gamma_1$ .

Using this, we can give a simplified version of the DILITHIUM signature scheme, seen in fig. 4.4. We split the bit representation of polynomial coefficients in the vectors in half. These are accessed via `HighBits` and `LowBits`, for the most and least significant half respectively.

**Theorem 4.10** *DILITHIUM as provided in fig. 4.4 is a correct signature scheme.*

*Proof.* To prove this statement we first need to make some observations: Due to  $z = y + cs_1$ , we can get by multiplying with  $A$ :  $Az = Ay + c(As_1) = w + c(t - s_2)$  and by transformation  $Az - ct = w - cs_2$  must hold. If  $\text{LowBits}(w - cs_2)$  is small, it follows, that  $\text{LowBits}(cs_2)$  is small as well, which allows us to conclude  $\text{HighBits}(w - cs_2) = \text{HighBits}(w)$ . Then we can combine this and get:  $w'_1 = \text{HighBits}(Az - ct) = \text{HighBits}(w - cs_2) = \text{HighBits}(w) = w_1$ , so for a valid signature  $c = H(m, w_1)$  must hold.  $\square$

When we analyze the security of this DILITHIUM, we will find that a signature will leak information about  $s_1$ . Thus this DILITHIUM version is not secure. This can be fixed, by choosing  $y$  so, that the coefficients of  $z$  are limited even further. We will not discuss this here.

The parameters of the key generation algorithm are standardized to these values:

- $q = 2^{23} - 2^{13} + 1$
- $n = 256$
- $k = 8$
- $l = 7$
- $\eta = 2$
- $\gamma_1 = 2^{19}$

and SHAKE-256 is used as a hash function.

When we analyze key size of DILITHIUM with this parameters, we get:  $|pk| = \log(q) \cdot n \cdot k \cdot l + \log(q) \cdot k \cdot n = 41216 + 5888$  Bytes = 47104 Bytes  $\approx$  47.1 KB and  $|sk| = \log(q) \cdot n \cdot (k + l) \cdot (2\eta - 1) \approx 1440$  Bytes = 1.4 KB. These rather large key sizes are not suitable, but can be fixed easily, by using a pseudo random number generator to generate  $A, s_1$  and  $s_2$  and instead of giving  $A, s_1$  and  $s_2$  in the keys explicitly give the needed seeds to reconstruct the values. Additionally, the vector  $t$  can also be compressed. Using all the optimizations described in [DKL<sup>+</sup>18], signatures are 2.7KB and the public key is 1.5KB large.

#### 4.4.1 Dilithium and Non-Resignability

We will now analyze, whether DILITHIUM yields (w)NR. We will only consider this in the ROM, as analyzing DILITHIUM in the plain model would require looking into SHAKE-256 thoroughly. Our argumentation will be based on theorem 3.11 and corollary 3.12.

**Theorem 4.11** (Resignability of DILITHIUM) *DILITHIUM as introduced in fig. 4.4 with the key-compression techniques applied, does not provide non-resignability in the ROM for messages compressed by the random oracle by more than  $|w_1| + \omega(\log(\lambda))$  bytes.*

*Proof.* To prove this theorem, we will show, that the statistical entropy requirement

$$\Pr_{m \leftarrow \mathcal{M}, (sk, pk) \leftarrow \text{KGen}(1^\lambda)}[\mathbf{H}_\infty(m \mid pk, \text{Sign}(sk, m)) \geq \omega(\log \lambda)] = 1$$

is provided by DILITHIUM, as defined in theorem 3.11, which this theorem will follow directly from.

As in the proof of corollary 3.12 we will use the auxiliary function and adversary given in fig. 3.3. For the auxiliary function  $\mathbf{aux}$  we have, that  $\overline{pk}$  gets chosen independently from  $m$  and thus  $\mathbf{H}_\infty(m \mid \overline{pk}, \text{DILITHIUM-Sign}(\overline{sk}, m)) = \mathbf{H}_\infty(m \mid \text{DILITHIUM-Sign}(\overline{sk}, m))$ .

When we unpack the definition of  $\text{DILITHIUM-Sign}(sk, m)$  we get:

$$\text{DILITHIUM-Sign}(sk, m) = (H(m, w_1), y + H(m, w_1) \cdot s_1).$$

We can find a lower bound of the expected number of messages leading to the same hash value:

$$\mathbb{E}_{m \leftarrow \mathcal{M}}[|\{m' \in \mathcal{M} \mid H(m', w_1) = H(m, w_1)\}|] \geq |w_1| + \omega(\log \lambda).$$

Which gives us a lower bound for the number of messages leading to the same signature when we fix the choice of  $y$ :

$$\mathbb{E}_{m \leftarrow \mathcal{M}} \left[ \left| \left\{ m' \in \mathcal{M} \mid \begin{array}{l} \text{DILITHIUM-Sign}(m', sk) \\ = \text{DILITHIUM-Sign}(m, sk) \end{array} \right\} \right| \right] \geq |w_1| + \omega(\log \lambda).$$

Directly giving us this lower bound for the entropy:

$$H_\infty(m \mid \text{DILITHIUM-Sign}(\overline{sk}, m)) \geq |w_1| + \omega(\log \lambda)$$

and by extension, we get the requirement for applying theorem 3.9:

$$H_\infty(m \mid \overline{pk}, \text{DILITHIUM-Sign}(\overline{sk}, m)) \geq \omega(\log \lambda).$$

Which concludes our proof, as we have found an adversary  $\mathcal{A}$  against the non-resignability of DILITHIUM with advantage greater than  $1 - \text{negl}(\lambda)$ , which means DILITHIUM does not provide non-resignability.  $\square$

This finding is not really surprising, as the construction of DILITHIUM is really similar to the idea of the BUFF-Transform. In both cases the signature consists of a hash and some signing value. The hash in the BUFF-Transform is calculated from the message and public key, in DILITHIUM from the message and  $A \cdot y$ , where  $A$  is part of the public key and  $y$  a random value. Due to the hash in DILITHIUM containing  $A \cdot y$ , where  $y$  is chosen at random, we see, that this is like the salted-BUFF-Transform, which leads us directly to our next theorem:

**Theorem 4.12** (Weak Non-Resignability of DILITHIUM) *DILITHIUM as introduced in fig. 4.4 provides weak non-resignability in the ROM.*

This follows directly from theorem 3.17, when we interpret the generation of  $z$  as the signing step and the generation of  $c$  as applying the salted-BUFF-Transform, which then allows us to apply the proof of weak-NR of the salted-BUFF-Transform.

## 5 Conclusion

This thesis examined the BUFF-Transform and its variants. We discussed what additional security features can be gained by the BUFF-Transform and BUFF-Lite-Transform. We also analyzed the negative findings of [DFHS24] on the resignability of the BUFF-Transform and showed that the salted-BUFF-Transform can provide weak non-resignability.

Then we used this to base an analysis of DILITHIUM and FALCON regarding their (weak) non-resignability. We reproduced the findings of [DF25], showing that the FALCON variant FALCON-PS-3 provides weak non-resignability and adopted their weak non-resignability notion to the one we presented based on [DFHS24].

We also showed, that DILITHIUM does not provide non-resignability, contrary to the claims made in [CDF<sup>+</sup>21], which is in line with the negative findings of [DFHS24]. But DILITHIUM provides weak non-resignability. This can be proven with the same approach as the proof for weak non-resignability of the salted-BUFF-Transform.

### 5.1 Future Work

When we analyzed FALCON-PS-3 for its weak non-resignability, we did only bind the advantage by newly introduced measures for the unpredictability and computational indistinguishability of auxiliary data. When we analyzed the salted-BUFF-Transform we used the guessing probability of the message as a bound. Bounding the weak non-resignability advantage of FALCON-PS-3 by its guessing probability would be useful, as it enables easier comparison of different signature schemes with respect to weak non-resignability.

In our analysis of post-quantum signature schemes, we left out multiple schemes. Of special interest is SPHINCS<sup>+</sup>, another NIST standardized post-quantum signature scheme. We did not analyze this scheme, as the negative and positive findings of [DFHS24], cannot be transferred to SPHINCS<sup>+</sup> directly. Currently there are only informal arguments on the non-resignability of SPHINCS<sup>+</sup>, which were made before the publication of [DFHS24], which makes us believe, these arguments do not hold. Analyzing the (weak) non-resignability of SPHINCS<sup>+</sup> formally would be an important contribution. Also the post-quantum signature schemes Squirrels, Racoon, HAWK, PROV, Vox and eMLE refer to the BUFF-Transform. Analyzing these schemes regarding non-resignability and weak non-resignability would also be a useful direction for future work.



# Bibliography

- [BCFW09] Alexandra Boldyreva, David Cash, Marc Fischlin, and Bogdan Warinschi. Foundations of non-malleable hash and one-way functions. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, pages 524–541, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
- [BS23] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. <https://toc.cryptobook.us/book.pdf>, January 2023.
- [BSW03] Boaz Barak, Ronen Shaltiel, and Avi Wigderson. Computational Analogues of Entropy. In Sanjeev Arora, Klaus Jansen, José D. P. Rolim, and Amit Sahai, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 200–215, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [C<sup>+</sup>24] David Cooper et al. Stateless Hash-Based Digital Signature Standard, 2024.
- [CDF<sup>+</sup>21] Cas Cremers, Samed Düzlü, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1696–1714, 2021.
- [CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM (JACM)*, 51(4):557–594, 2004.
- [Cho17] Arjun Chopra. Glyph: A New Instantiation of the GLP Digital Signature Scheme. *Cryptology ePrint Archive*, 2017.
- [D<sup>+</sup>15] Morris J Dworkin et al. SHA-3 standard: Permutation-based hash and extendable-output functions. 2015.
- [DFF25] Samed Düzlü, Rune Fiedler, and Marc Fischlin. Buffing falcon without increasing the signature size. In Maria Eichlseder and Sébastien Gambs, editors, *Selected Areas in Cryptography – SAC 2024*, pages 131–150, Cham, 2025. Springer Nature Switzerland.
- [DFHS24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, and Patrick Struck. On the (In)Security of the BUFF Transform. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024*, pages 246–275, Cham, 2024. Springer Nature Switzerland.
- [DKL<sup>+</sup>18] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018.

- [DL01] Whitfield Diffie and Susan Landau. The Export of Cryptography in the 20th Century and the 21st. 2001.
- [ECH24] The European Court of Human Rights. Case of Podchasov v. Russia, 2024. [https://hudoc.echr.coe.int/eng/#{"itemid":\["001-230854"\]}](https://hudoc.echr.coe.int/eng/#{).
- [FHK<sup>+</sup>20] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, Zhenfei Zhang, et al. Falcon: Fast-fourier lattice-based compact signatures over NTRU, 2020.
- [GJK24] Phillip Gajland, Jonas Janneck, and Eike Kiltz. A closer look at Falcon. *Cryptology ePrint Archive*, 2024.
- [GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206, 2008.
- [GS02] Craig Gentry and Mike Szydlo. Cryptanalysis of the revised NTRU signature scheme. In *International conference on the theory and applications of cryptographic techniques*, pages 299–320. Springer, 2002.
- [HHGP<sup>+</sup>03] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSign: Digital signatures using the NTRU lattice. In Marc Joye, editor, *Topics in Cryptology — CT-RSA 2003*, pages 122–140, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A Pseudorandom Generator from any One-way Function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
- [HS93] Francis Harry Hinsley and Alan Stripp. *Codebreakers: The Inside Story of Bletchley Park*. Oxford University Press, 1993.
- [KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 1st edition, 2007.
- [KRS09] Robert König, Renato Renner, and Christian Schaffner. The Operational Meaning of Min- and Max-Entropy. *IEEE Transactions on Information Theory*, 55(9):4337–4347, 2009.
- [Men24] Alfred Menezes. Kyber and Dilithium, Aug. 2024. <https://cryptography101.ca/kyber-dilithium/>.
- [PS05] Thomas Pornin and Julien P. Stern. Digital Signatures Do Not Guarantee Exclusive Ownership. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 138–150, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- [Reg09] Oded Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

- [Sha48] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.
- [Sho94] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. IEEE, 1994.
- [Sil06] Joseph H Silverman. An introduction to the theory of lattices and applications to cryptography. *Computational Number Theory and Applications to Cryptography, University of Wyoming*, pages 1–212, 2006.
- [Sno15] Edward Snowden. *The Art of Privacy in the Age of Surveillance*. Metropolitan Books, 2015.
- [SOG23] SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms. Specification, SOG-IS Crypto Working Group, Feb 2023. <https://sogis.eu/documents/cc/crypto/SOGIS-Agreed-Cryptographic-Mechanisms-1.3.pdf>.