LUND UNIVERSITY

Code-based Cryptography: Attacking and Constructing Cryptographic Systems

Nguyen, Vu

2025

Code-based Cryptography: Attacking and Constructing Cryptographic Systems

Vu Nguyen

LUND
UNIVERSITY

ISBN 978-91-8104-523-9 (print)
ISBN 978-91-8104-524-6 (electronic)
Series of licentiate and doctoral theses
No. 186
ISSN 1654-790X-186

Vu Nguyen
Department of Electrical and Information Technology
Lund University
Box 118
SE-221 00 Lund
Sweden

Typeset using LaTeX.
Printed in Sweden by Tryckeriet i E-huset, Lund, 2025.

Abstract

This thesis discusses novel results in the area of code-based cryptography, ranging from cryptanalyses on several code-based cryptographic constructions to proposing a code-based authentication scheme based on a novel Syndrome Decoding variant.

To address the looming threat of large-scale quantum computers that would break many widely-used public-key cryptosystems, the *National Institute of Standards and Technology* (NIST), in 2016, promptly called for new quantum-resistant cryptographic standards. After several comprehensive evaluation stages, several algorithms were chosen as the way forward. Recently, in 2022, "NIST expressed particular interest in additional general-purpose signature schemes based on a security assumption that did not use structured lattices as well as signature schemes with short signatures and fast verification..." [NIS22]. This process reignited the competition, introducing many interesting and novel proposals.

Code-based cryptography has been pivotal in both processes, being the security keystone for many proposals, particularly in the selected HQC algorithm. Its strong track record of research offers strong confidence in code-based cryptosystems. From a complexity theory point of view, code-based assumptions are often $\mathcal{NP}$-hard computational problems, which have been the staples for provably-secured systems. On the other hand, cryptanalysis algorithms have witnessed significant advancements, employing increasingly more sophisticated techniques. Yet, the fundamental Syndrome Decoding Problem remains resistant, suggesting that code-based cryptography is an exceptionally well-founded and dependable tool.

Many variants/alternatives of the Syndrome Decoding Problem have been put forward to offer better performance without compromising security. Specifically, in this thesis, one encounters the *Learning Parity with Noise* (LPN) and *Restricted Syndrome Decoding Problem* (RSDP). LPN has been a notable candidate in lightweight cryptography, and RSDP has gained traction due to its appearance in CROSS- a remaining candidate in the Round-2 of NIST Additional Digital Signature Schemes. Code-based cryptography, as a research field, shows immense versatility and richness far beyond its origin as a PKE proposed by McEliece [McE78].

We analyze several lightweight code-based cryptosystems in the first three works, ranging from stream ciphers to wPRFs and authentication protocols. We investigate the design weaknesses that allow us to launch attacks using various techniques. In particular, we analyze a novel LPN-based stream cipher called Firekite, a wPRFs construction, and an HB-like authentication protocol named LCMQ. Using diverse techniques in conjunction with information-set decoding algorithms (ISD), our studies improve previous results (if any) and impose stronger security parameters for said constructions.

Then, we draw connections between lattice-solving algorithms and traditional syndrome decoding algorithms with our new proposal: a *sieving-style* ISD algo-

rithm. Our algorithm offers a novel time-memory trade-off in solving relevant code-based parameters. In the low error-weight regime, the sieving-style ISD can use memory more efficiently without losing its competitiveness in computational performance. Thus, we introduce a valuable and practical alternative to cryptanalysis.

The last two papers look at the novel RSDP problem from a new perspective - the *Oracle model*, analogous to the LWE or LPN problems. We construct an HB-like authentication protocol, replacing the LPN problem with the (Oracle) RSDP problem, showing its remarkable adaptiveness to the most secure designs. In practice, RSDP structures allow incredibly efficient operations, rivaling those of LPN. Moreover, RSDP also achieves high-security guarantees with modest parameters, yielding significant superiority regarding communication cost. Finally, we expand the cryptanalysis of the RSDP problem, especially when many RSDP samples are allowed with a BKW-style solver. We analyze the concrete complexity of RSDP in new regimes outside of CROSS parameters. Hence, our work is a useful calibrating tool for similar RSDP-based cryptosystems in the future.

Acknowledgements

I dedicate this thesis to my dearest wife, Thuong, who has made my Ph.D. journey the most memorable and significant period of my life, for it is when I married her. Words cannot describe how much I appreciate you for always supporting and being with me throughout my academic journey, wherever it is. Despite all the sacrifices you have made to be together, you remain a supportive, sympathetic, and understanding partner, enduring moments that I am not proud of. Your love, sweetest personality, encouragement, and delicious food made this journey more pleasant and worthwhile.

Firstly, I want to extend my sincerest and deepest gratitude to my main supervisor, Thomas Johansson, for saying you needed a mathematics Master's student at a dinner 6 years ago. Above all, thank you for being patient, understanding, and confident in me even when cryptography was new to me. You gave me the courage, time, and space to grow and always came to help when I needed it the most. Your intellectual curiosity and enthusiasm remind me every day why I wanted to research in the first place. I now concur with Jing that you are the best supervisor ever. Also, I wish to express the same admiration and appreciation for my cosupervisor, Qian, who has been an inspiration for me and other junior researchers with your discipline, work ethic, and incredible astuteness. You and Thomas not only provided invaluable guidance and insights in every project we had been working on together, but also taught me an optimistic, curious, and positive attitude when there was a problem (and many of them, there were!). I could not have asked for better research mentors.

Besides my supervisors, many published papers would not have been possible without my coauthors, Willi and Mustafa. Thank you, Willi, for allowing me to work on your ideas and for your collaboration in the first two papers, which were crucial in my formative years as a Ph.D. Together with Thomas, you have both been generous in lending an immense wealth of knowledge and wisdom to me. Thank you, Mustafa, for the short but fruitful time we had. You always knew how to improve a paper!

I have been blessed to know so many fantastic colleagues and seniors in my research group, all of whom I wish I had known earlier. The office corridor is not the same without your lovely personalities. Special thanks to Syafiq, my Southeast-Asian brother, who put up with my antics and stupid moments and allowed me to be at my worst while working in our shared office(s). Most importantly, you convinced me to use Ubuntu, among many other things. I will miss our chats, for they are as random and funny as unhinged. I want to apologize to Denis and Dachao, who never denied an unauthorized office invasion with unintelligible and self-deprecating monologues of mine. Thank you for always letting me talk through my problems and questions, whether it is about research, programming, or even badminton. To the older Chinese sisters, Jing and Hui (and Dachao), thanks for indulging my obsession with the Chinese language and culture. Joakim and

Karim, thank you both for showing me "How to Adult (appropriately)" and saying (roughly) "Get a grip, Vu, you are a Ph.D." I wish to thank the other (new and old) Ph.D.s: Erik, Martin, Alexander, Pegah, Rohon, Markus, Maggie, Shouran, Arthur..., for all the lunch and fika discussions (mostly banters). My appreciation to the seniors, Paul, Elena, Christian, and Sara, for being great research role models and occasional guidance. To the finance and administration staffs of the faculty: Elisabeth, Erik, Linda..., you have been so helpful in assisting with all the paperwork during my time here. You are now rid of another Ph.D. student.

It is getting too long and over-the-top, so I prefer to conclude with an acknowledgement to my friends and family. My time in Sweden has been made more remarkable and colorful due to my Vietnamese cohort: Mi, Vu (not me), Huan, Chi, Dat, Ha... Thanks for all the fantastic hangouts, cooking, and trips together. You have been with me through good and hard times. Linus, you made me right at home when I first came here, have always told me what I need to hear, and you appreciate my humor. Let's agree not to be moody simultaneously so that we can lend a hand to each other. Finally, thanks to my parents and siblings for everything and for making me the way I am now. Every step I made in foreign lands was firm and steadfast because I knew you were always there looking out for me.

Vu Nguyen
Lund, April 2025

Contribution Statement

The following papers are included in this dissertation:

Paper I Thomas Johansson, Willi Meier, Vu Nguyen. "Attacks on the Firekite Cipher". In *Fast Software Encryption (FSE) 2023, Beijing, China*. IACR Transactions on Symmetric Cryptology. 2022, 3, p. 191–216.

Paper II Thomas Johansson, Willi Meier, Vu Nguyen. "Differential cryptanalysis of Mod-2/Mod-3 constructions of binary weak PRFs". In *International Symposium on Information Theory 2023, Taipei, Taiwan*. 2023 IEEE International Symposium on Information Theory (ISIT), pp. 477-482, IEEE.

Paper III Vu Nguyen, Thomas Johansson, Qian Guo. "A Key-Recovery Attack on the LCMQ Authentication Protocol". In *IEEE International Symposium on Information Theory (ISIT) 2024, Athens, Greece*. 2024 IEEE International Symposium on Information Theory (ISIT), pp. 1824-1829, IEEE.

Paper IV Q. Guo, T. Johansson and V. Nguyen. "A New Sieving-style Information-set Decoding Algorithm". In IEEE Transactions on Information Theory, vol. 70, no. 11, pp. 8303-8319, IEEE.

Paper V Thomas Johansson, Mustafa Khairallah, Vu Nguyen. "Efficient Authentication Protocols from the Restricted Syndrome Decoding Problem". In *IEEE European Symposium on Security and Privacy 2025 (EuroS&P), Venice, Italy*.

Paper VI Thomas Johansson, Qian Guo, Vu Nguyen. "A BKW-Style Solver for the Restricted Syndrome Decoding Problem". In *International Workshop on Code-Based Cryptography 2025, Madrid, Spain*.

The table below summarizes the contribution that Vu Nguyen made to each paper. In particular, The capital letters "YES" indicate roles where Vu Nguyen took primary responsibility for the given role, whereas "yes" indicates partial involvement. The hyphen shows a minor contribution to the tasks. In all papers, the writing was given to Vu with increasing independence and involvement.

Paper	*Writing*	*Concepts*	*Implementation*	*Evaluation*
I	yes	yes	YES	yes
II	YES	-	YES	yes
III	YES	-	YES	YES
IV	yes	yes	YES	yes
V	YES	YES	-	yes
VI	yes	YES	YES	yes

The details of Vu Nguyen's contributions to each work are described in the following paragraphs. In Paper I, he was tasked with investigating and implementing the algorithm of which his co-authors proposed the initial ideas. The evaluation and analysis of the paper was carried out with feedback from the co-authors.

In Paper II and Paper III, Vu was introduced to the wPRFs/LCMQ constructions by his co-authors. He worked on deriving the heuristic arguments necessary for the attack based on crucial insights from his senior co-authors. Vu was responsible for simulating, verifying, and evaluating the attack complexity.

In Paper IV, the basic concept of sieving in a code-based cryptography context was introduced by his senior co-authors. Vu investigated and materialized the algorithms. Many algorithm optimizations were possible with his co-authors' contributions, including sketching an adaptation for the *Decoding-One-Out-of-Many* technique. All authors are involved in deriving theoretical probabilistic arguments for the algorithms, evaluating, and writing the manuscript. Vu was also tasked with implementing the algorithm in many versions and writing complexity estimation scripts.

In Paper V, Vu was tasked with investigating the prospect of building a lightweight authentication protocol from the novel Restricted Decoding Problem. He proposed the initial designs and adapted well-known reduction proofs for the designs, which his co-authors subsequently improved. He investigated the Restricted Syndrome Decoding Problem and the state-of-the-art cryptanalysis to derive parameters for the protocols. His co-authors chiefly did the hardware implementations/evaluations.

In Paper VI, Vu and his co-authors were all involved in developing a new algorithm for the Restricted Syndrome Decoding Problem by finding low-weight vectors. Vu experimented with many approaches and settings, exploring the viability of the ideas, but was not fruitful when attacking the parameters of the digital

signature scheme CROSS. Following Paper V, Vu was tasked with investigating the algorithm in the novel RSDP settings where many samples are allowed. He developed the skeleton of the algorithms, arguments, implementation, and complexity estimate. His co-authors then suggested a multiple-depth variant, which yielded significant improvement. Most importantly, their insights contributed greatly to the implementation evaluation, especially in understanding the algorithm's correctness and success probability. A further description of the papers' contributions *to the research field* is presented in Section 4.1.

Contents

Overview of the Research Field

Introduction to Cryptology

At the time of writing this thesis, the subject of privacy had become normalized to every internet user. Social networks and data storage services have evolved so much that they have become conveniences in our daily lives. In today's interconnected world, sensitive data, such as financial records and personal information, is constantly transmitted and stored digitally. People are reasonably concerned about their privacy, even innocent conversations with their friends and loved ones, which can be used maliciously against them. However, such worrying thoughts, as well as countermeasure practices, are not precisely recent in our history. In fact, *cryptography* - the art of "secret writing" has been around for millennia and probably not too long after the invention of writing itself. It originated from the simple need to conceal the meaning of messages only meant for intended receivers, primarily if the communication is eavesdropped/intercepted. The (simplistic) cryptography framework can be described in Figure 1.1.

In essence, the goal of (early-age) cryptography is to render the plaintext *unintelligible* by some mechanism (*encryption*), and only those who have the proper method can unscramble the ciphertext to retrieve the actual plaintext (*decryption*). The main antagonist in our scenario is called the *adversary*, which is abstract and ambiguous despite being overly personified with names like Eve. It can be anything: a nosy neighbor, a spy, an insecure or leaky channel, or even someone who provides storage for our (hopefully) encrypted data. The bottom line is that they

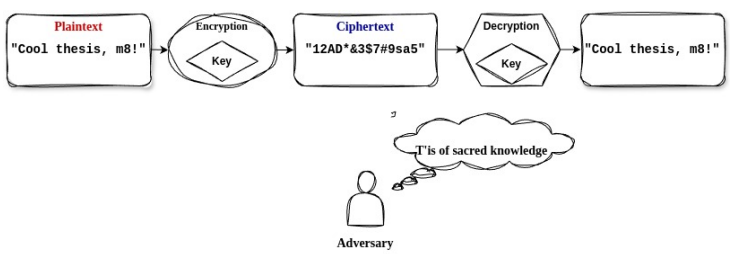


Figure 1.1: A typical model of encryption and decryption in cryptography.

all have an insatiable thirst for reading our messages and information.

In ancient times, it became a valuable tool to protect messages of military significance. Indeed, the most well-known ancient cipher is perhaps *Caesar's cipher*. It replaces every letter in the plaintext with another one, some fixed number of positions down the alphabet (shift). However, since the alphabet is not large, testing all possible shifts (i.e., "key") is not a bad strategy. Caesar's cipher is a classic example of *substitution* cipher. Fast-forwarding a few centuries, it was used more generically and securely. In particular, one letter can be mapped to any other arbitrary one, making the space of possible keys much larger (e.g., 26! for the English alphabet). However, a closer look reveals that the frequencies of the letters do not change, and certain words are much more frequent (e.g., "the"). Thus, it is possible to break such a cipher without trying all possible keys from such extra information. Even the "seemingly" unbreakable *Enigma* was eventually decoded, which played a crucial role in World War II. The history of cryptography has no shortage of similarly broken cryptosystems. They all share common pitfalls: relying on a cipher without properly analyzing or designing without sound scientific methods or a security framework, or even worse: "security through obscurity".

Knowing the (mal)practice of predecessors, cryptographers nowadays have to play the role of the adversary. The goal is to discover weaknesses and oversights as well as analyze the security of cryptographic constructions. This scientific practice is called *cryptanalysis*, and it is complementary to cryptography to form *cryptology*.

In modern times, especially in the age of information, the applications of cryptography extend to many more aspects of data protection than merely confidentiality. For example, the scope of cryptography also includes data integrity, authenticity, secure communication, verification, secure multi-party computation, identity protection, and so on. More familiar applications include (but are not limited to) banking, anonymous voting, credit cards, and new-age financial movements such as cryptocurrency. Hence, it is not an exaggeration to claim that cryptography has become an indispensable tool and has penetrated every fabric of our digital way of life. To adapt to the ever-growing list of needs, cryptography has also been enriched and developed with equally suitable theories and techniques. As a science, cryptography is closely related and lies in the intersection of many other foundational scientific disciplines, such as mathematics, computer science, and information theory.

This chapter provides a brief introduction to concepts in cryptology. First, we learn about the general framework of (classical) cryptography, cryptanalysis, and security notions. In particular, we present the motivation for the subfield of post-quantum cryptography and code-based cryptography (our focus). In broad strokes, we summarize the research area of code-based cryptography and its status in the research community. We strive to make the thesis self-contained without being too lengthy. Therefore, we will only cover notions that are relevant to our publications. Moreover, the goal is to give general intuitions unless it is necessary to present formulae and definitions. The readers will eventually encounter precise

and rigorous formulations in the publications, and we provide many references to well-written surveys/articles throughout the thesis.

1.1 Notations and Typesetting

In the first part of this thesis, we frequently use the following notations.

Algebra

We write $\mathbb{F}_q$ for a finite field of q elements, and $\mathbb{F}_q^n$ as the n-dimensional vector space over $\mathbb{F}_q$. We use bold letters such as $\mathbf{a}, \mathbf{A}$ for vectors (row, unless mentioned otherwise) and matrices, respectively. The coordinates of a vector are often denoted by the corresponding non-capitalized characters. For example, a vector $\mathbf{x}$ of length n is usually expressed as

$$\mathbf{x} = (x_1, x_2, \ldots, x_n).$$

The transpose of a matrix is denoted by $\mathbf{A}^\intercal$, and a matrix can be written as

$$\mathbf{A} = (\mathbf{x}_1^\intercal, \mathbf{x}_2^\intercal, \ldots, \mathbf{x}_n^\intercal),$$

where $\mathbf{x}_i^\intercal$ are its column vectors. The identity matrix of dimension $n \times n$ is $\mathbf{I}_n$. The vector addition, written as $\mathbf{x} + \mathbf{y}$, is to be understood as the bit-wise addition. In particular, it is the bit-wise XOR operation in $\mathbb{F}_2$. The dot product of $\mathbf{x}$ and $\mathbf{y}$ is simply denoted by $\mathbf{x} \cdot \mathbf{y}$.

The set of integers $1, \ldots, n$ is denoted by $[\![1, n]\!]$. For a vector $\mathbf{x} \in \mathbb{F}_q^n$, we call $\mathbf{x}_\mathcal{I}$ its projection on the coordinates that belong to an index set $\mathcal{I} \in [\![1, n]\!]$. In particular, $\mathbf{x}_{[\ell]}, \ell \leq n$ is the truncated vector of the first ℓ coordinates. Similarly, $\mathbf{A}_{[\ell]}$ is the submatrix that comprises the first ℓ rows of $\mathbf{A}$.

Probability

Let x be a random variable, taking values from a set X, and χ be a probability distribution on X. If x is sampled according to χ, we write

$$x \xleftarrow{\chi} X,$$

or sometimes

$$x \leftarrow \chi,$$

if the set X is evident and the distribution χ is in our focus and interest. When x is sampled uniformly at random, we use the notation

$$x \xleftarrow{\$} X.$$

Complexity

Below is a list of asymptotic complexity notations that occur in the thesis.

$$\begin{aligned} f(n) \in \mathcal{O}(g(n)) &\quad |f(n)| \leq k \cdot g(n) \text{ for some positive } k, \\ f(n) \in \Omega(g(n)) &\quad |f(n)| \geq k \cdot g(n) \text{ for some positive } k, \\ f(n) \in \Theta(n) &\quad \ell \cdot g(n) \leq |f(n)| \leq k \cdot g(n) \text{ for some positive } \ell, k, \\ f(n) \in o(n) &\quad |f(n)| \leq k \cdot g(n) \text{ for every positive } k. \end{aligned}$$

We frequently use the statement "...in the order of ..." interchangeably with the notation $\mathcal{O}$.

1.2 Cryptography

Generally, we can categorize cryptosystems into the following classes of *cryptographic primitives*. Note that in cryptographic applications, the technical description of a primitive can be convoluted, involving multiple-step protocols. Moreover, cryptography does not always revolve around exchanging plaintexts and ciphertexts. For instance, cryptography includes other crucial concepts such as key distribution, digital signatures, zero-knowledge proofs, and so on. However, for the sake of a simplistic presentation, we can safely assume that for our cryptosystem, Alice and Bob are trying to communicate, and they have come up with a clever encryption/decryption scheme (E, D) against a nosy adversary, Eve. The pair (E, D) denotes a generic encrypting and decrypting function where E (resp. D) takes a *key* and plaintext (resp. ciphertext) as input.

Symmetric cryptography Alice and Bob share an identical key K for encryption and decryption. Using the key K, Alice can encrypt the message (plaintext) $\mathbf{m}$ by computing the ciphertext $\mathbf{c} = E_K(\mathbf{m})$. Upon receiving, Bob decrypts by computing $\mathbf{m} = D_K(\mathbf{m})$ by the same key. Examples of symmetric primitives in encryption are stream ciphers and block ciphers. In a stream cipher, the key can be used to initiate a *pseudorandom generator* that produces a pseudorandom bit string called *keystream*. The keystream is then added (XOR) to the plaintext to obfuscate the message. Since the key is shared, the receiver can reproduce the same keystream to retrieve the plaintext. Block ciphers, on the other hand, operate by encrypting the plaintext block-wise. Due to the nature of stream ciphers, they introduce very little data latency and can be implemented quite efficiently, which is suitable for data-in-transit encryption, low-latency applications, or resource-constrained environments. Block ciphers, despite being bulkier than stream ciphers, are even more ubiquitous nowadays (e.g., AES, DES, Blowfish) and have many applications in data storage, internet communication protocols...

Another case of symmetric primitives in this thesis is *authentication*, albeit not for encryption purposes. In addition to the encryption module, other build-

ing blocks are necessary for a cryptosystem. They come from other cryptography objects such as *Pseudorandom functions* (PRFs), *weak Pseudorandom functions* (wPRFs), or *Pseudorandom number generators* (PRNG) that facilitate key derivation, nonce, etc. Despite not being explicit, they are crucial to secure encryption (e.g., a "weak" key might tell the adversary something). In Paper II, we come across PRFs and wPRFs. Briefly speaking, PRFs are a family of efficiently computable functions indexed by a "key-space". Given all input and a random function of the family, the output appears *indistinguishable* from uniform random distribution without knowing the key (that is, the chosen function). In the weak version, wPRFs, a distinguisher can only observe pairs (*input*, *output*) for uniformly random *input*. We will formulate these notions in later publications. For a comprehensive survey, we refer to [BR17a].

Asymmetric cryptography It is also known as *Public-key Cryptography* and involves a pair of mathematically related keys (pub, sec). The public key pub is available to everyone (including Eve), while the secret key sec is kept secret by, e.g., Alice. Suppose Bob wants to communicate with Alice securely. Alice produces a key pair (pub, sec) and makes public pub. A message $\mathbf{m}$ is encrypted by Bob with $E_{pub}(\mathbf{m})$. The secret key is then used to decrypt and retrieve the original message as $D_{sec}(E_{pub}(\mathbf{m}))$. In contrast to symmetric primitives, Bob does not need to know the secret key. The idea is that it is mathematically infeasible (for Eve) to compute sec from pub. In other words, it is easy to decrypt with the secret key and hard to do otherwise. Such a situation is called *trapdoor one-way function*, which can be inspired by computationally hard mathematical problems.[1] Two main problems for asymmetric cryptography before post-quantum cryptography are the *factoring* (e.g., RSA [RSA78]) and *discrete-log* problems (e.g., *Diffie-Hellman* key-exchange [DH76]). However, compared to symmetric cryptography, functions arising from such problems can often be slow for many purposes. Therefore, public-key cryptography can be used in hybrid cryptosystems to encrypt and facilitate the key in a symmetric cipher.

Other than the above classes, there are other primitives that are outside the scope of this thesis. Traditional key-based cryptography inherently carries many nuances and intricacies in key generation and key management that we have glossed over. In contrast, *keyless* cryptography, in particular *cryptography hash function*, does not need an explicit key. Essentially, a hash function (efficiently) produces a fixed-size "fingerprint" of (variable-length) data.

1.3 Security Notions

We have seen the types of cryptography primitives, but we have yet to address the following questions:

[1] The existence of one-way functions is an open conjecture.

> Given a cryptosystem, what exactly do we mean by "secure"?

Even before that, we should discuss which system components are encompassed by the security notions. Surprisingly, this is not exactly straightforward. As history has shown, ancient cryptography designers and users often falsely trusted encryption schemes that were far from secure simply because not enough tinkering or serious scrutiny was conducted on them. However, given enough time and analysis, reverse engineering would eventually be possible. It was not until 1883 that Auguste Kerckhoffs, in his article, stated his famous principle, which has become ever so integral for cryptography. The *Kerckhoffs' principle* can be interpreted as "The security of a system should only rely on the key". In other words, when designing a cryptosystem, one must assume that the adversary has a copy of the device, and only the keys are kept secret. In modern cryptography, there are many more nuances to this principle. For example, the standard in today's encryption is that the keys must also be sampled uniformly at random.

So, how do we *quantify* the security of a cryptosystem? Claude Shannon, in 1945, was the first person to formulate an answer rigorously. His ground-breaking insight came with the introduction of information theory, which is crucial to cryptography and many other disciplines, such as communication, information processing, and so on. Shanon's idea of security, called *information-theoretic security*, or colloquially *unconditional security* can be summarized in the following.

Unconditional Security For a cryptosystem to achieve unconditional security (information-theoretic security or perfect secrecy), it states that

> A ciphertext should not reveal any additional information about the plaintext.

More specifically, it means that an adversary, even with unlimited computing power, cannot do better than guessing the plaintext among all possible ones after receiving the ciphertext.

A classic example defining the "security game" in many cryptography textbooks is such that: Given two messages m_0, m_1 and $b \in \{0, 1\}$. Assume we choose b and a key K at random and encrypt m_b with a perfectly secure encryption scheme (E, D), then the probability that an adversary guesses b correctly after seeing the ciphertext $E_K(m_b)$ is no greater than $1/2$.

A notable cipher that achieved unconditional security is the Vernam Cipher, a.k.a. the one-time pad (OTP): to encrypt a length-k message, we need a length-k key. In addition, it must hold

- The keys have to be sampled uniformly at random.
- The same key is never to be used twice.

However, this is infeasible with the abundance of information/communication going on nowadays. For instance, users will eventually run out of ideas for new passwords and re-use their keys, which defeats the purpose of perfect secrecy. Historically, the one-time pad was employed when secrecy is paramount (e.g., military and espionage documents), and it is still used today when appropriate (e.g., OTP for bank log-in). Other examples of cryptosystems that achieve unconditional security are Shamir's secret sharing scheme [Sha79], or secure *multi-party computation*. To have more usability, cryptographers turn to more relaxed notions of secrecy.

Instead of striving for perfect secrecy, we can be satisfied with an encryption scheme as long as it takes a **really long time** to break it. For example, if the best algorithm takes 2^{256} operations to break an encryption scheme, then for all intents and purposes, one might as well employ such a scheme. We say that a cryptosystem has t-bit security if the best attack is comparable to an exhaustive search over t bits.[2] In essence, we recognize the fact that an adversary might have enormous computing capability, but the effort might be so great that he/she might as well guess it.

With this notion, we can discuss the security of a cryptosystem in quantifiable senses, such as operations (time) and data (space) complexity. Categorically, we speak of *exponential*, *sub-exponential*, and *polynomial* complexity. We interchangeably use "hard", as in exponential time, and "efficient", as in polynomial time. It suffices to remember that exponential-time algorithms are much more expensive than polynomial-time ones, at least asymptotically. However, we must tread carefully with these notions. Asymptotic complexity can be misleading and often says little about *concrete complexity* (i.e., empirical evidence of a cryptosystem). For instance, a very slow-growing exponential function can be smaller than a polynomial function in certain input regions, which means that, for cryptographically relevant parameters, a hard problem (exponential complexity) is not always more costly to solve than an easy problem (polynomial complexity).

Empirical Security In contrast to unconditional security, which is secure under any circumstance, this type of security is only "as good as" the most efficient attack. The idea is that confidence in a cipher is accreted after a reasonably long time, especially under the intense scrutiny of many cryptanalysis attempts. However, it is entirely possible that newer attacks can be found. However, there is no formal proof ("guarantee") for an empirically secured cryptosystem, only a strong track record of its resistance against attackers. The AES block cipher is a candidate for this category.

[2] More formally, we can relate to the "guessing game" above: Any algorithm using at most 2^t operations cannot guess b correctly with a probability greater than $1/2 + 2^{-t}$.

Provable security On the other hand, this type refers to any security that can be proved. However, the catch is that it only works if the underlying assumption is valid. Usually, it involves *reducing* the task of breaking a system to solving a *computationally hard problem*.[3] The assumption is that problems chosen cannot be solved efficiently (i.e., in polynomial time). Formal techniques for *proof of reduction* have been developed to ensure soundness when establishing security. However, whether a problem is (unconditionally) hard is unknown to us, which calls for empirical evidence for a problem. Thus, to prevent the entire collapse of hardness-based security, we generally choose the most well-understood problem that is widely regarded as really hard. Certain problems (e.g., $\mathcal{NP}$-complete) for which, if an efficient attack is found, would lead to significant implications in complexity theory. Therefore, they are considered very safe bets.

1.4 Cryptanalysis

Besides constructing cryptographic primitives, it is equally important to scrutinize them with the best attacks. The goal is not to undermine but to understand and justify whatever security claims are made about a design. That is, we cannot rely on the enemy to discover a system weakness first (and usually when it is too late). Designers and advocates for a cryptosystem bear as much responsibility to investigate, under scrutiny from various angles, all the potential exploitations. This is the only way we can ensure confidence, and it is, in fact, at the core of any other intellectual science. This is the gist of cryptanalysis. Furthermore, we can only discuss the security of a cryptosystem in the context of an adversary. This was already hinted at when we discussed the security notions (e.g., we go from unlimited to reasonably large computational power). With this in mind, we mathematically model an adversary and "give" them as much power as one reasonably assumes. As discussed before, it does not matter who the adversary is (although we often give the name "Eve"). What is important to cryptography practitioners is its goals and capabilities (or power) when attacking a cryptosystem in question.

Types of attacks for encryption schemes

Generally, the goal of an adversary can be put into the following categories.

Key recovery The adversary attempts to recover the secret key.

Message recovery The adversary attempts to recover a specific message.

[3] Note that there are many "hard" math problems, but we are focusing on *cryptography-relevant* ones.

Distinguishing attack The adversary attempts to distinguish whether a piece of information is random or generated by a cipher.

We can see a clear hierarchy among these attack goals. Trivially, an adversary that can successfully recover the secret key can launch a message recovery attack. Similarly, a message recovery attack implies a distinguishing attack. Therefore, resisting distinguishing attacks is a stronger security requirement than, e.g., key-recovery attacks.

Adversarial Power

Similarly, we describe how the adversary interacts with our cryptosystems, which heavily impacts the possibility of the above attacks.

Ciphertext-only attack (COA) The adversary is assumed to have access to arbitrary ciphertexts. This is the weakest adversarial assumption, corresponding to the weakest notion of security.

Chosen-plaintext attack (CPA) The adversary can choose plaintexts and observe the corresponding ciphertexts. A stronger notion of this attack is *adaptive* chosen-plaintext attack, where the adversary can perform a series of adaptively chosen plaintexts to be encrypted. In this sense, the attacker may retrieve some information from each encryption and can use it for the next plaintext, attempting to reveal even more information.

Chosen-ciphertext attack (CCA) The adversary can choose ciphertexts and observe the corresponding plaintexts. Similarly to CPA, we also have a boosted, adaptive version of CCA, called CCA2, where the adversary can adaptively choose the ciphertexts to be decrypted.

We have a similar situation with the attack goals. A known-ciphertext attack refers to the weakest adversary, and a chosen-ciphertext attack is the strongest adversary. In combination, we can strive for the highest security guarantee, called IND-CCA2, that is *indistinguishability under adaptive chosen-ciphertext attack*.

In the publications, we will come across differently named notions of adversary that are more specific to the cryptosystem at hand. However, they can all be traced back to the above notions. For example, for an authentication system, we discuss attacks in *passive, active,* or *Man-in-the-Middle* models. These models emphasize the adaptive nature of the adversary. For instance, it may only "passively" observe the ciphertext, or it can adaptively choose a specific plaintext and observe the encryption, or even alter the communication between parties and observe the reaction. Another example is the *differential attack* that we perform on wPRFs, where the input (plaintexts) are chosen to be different in a strategic manner so that the output (ciphertexts) can reveal beneficial information for an attacker. Such an attack falls under the category of CPA.

The above notions cover the cryptanalysis we present in this thesis. However, they are by no means comprehensive. For instance, we often brush over the brute-force attack that is usually associated with exhaustive search, i.e., the adversary disregards every exploitable underlying structure of a cryptosystem. Such an attack can provide a useful (but crude) upper bound on the computational security of a cryptosystem. On the other hand, we also encounter more sophisticated attackers that reflect the advancement in technology, such as *side-channel attacks*. In this scenario, the adversary is assumed to have access to "leaks" (power consumption, timing, electromagnetic...) from physical devices implementing our cryptosystems. Such leaks represent additional information that can be used to speed up the above attack or even devise a novel attack strategy.

1.5 Post-quantum cryptography

1.5.1 The quantum threat

Given the mathematical richness, more cryptographers have leaned towards provable security, and the field has been rapidly expanding. Instead of frustrating generations of mathematicians, stubborn problems reveal the untapped potential to construct cryptographic primitives, which are ever so diversified and sometimes hard to keep track of. However, hardness assumptions are not impervious to the test of time and technology.

In the pre-quantum era, we often relied on the *factoring* and *discrete-log* problems. They are crucial to cryptography because they underpin the security of several widely used applications. RSA, a cornerstone of secure online transactions and digital signatures, relies on the difficulty of factoring large numbers. Similarly, the Diffie-Hellman key exchange is vital for establishing secure communication channels (like HTTPS). Its security ensures that even if an attacker intercepts the exchanged messages, they cannot easily determine the shared secret key. Beyond these, variations like *Elliptic curve Cryptography* are important for resource-constrained devices like smartphones, securing everything from messaging apps to cryptocurrency transactions. In essence, the computational hardness of these problems is what makes our digital world secure.

Quantum computers, introduced by Manin [Man80] and Feynman [Fey82], can use quantum mechanics to perform computations. The topic has become even more relevant in cryptography with the prospect of using them to tackle (some) hard problems faster than classical computers. Shor [Sho94], in his seminal work, introduced his trailblazing quantum algorithms that solve the problems mentioned above in polynomial time. To a lesser extent, in symmetric cryptography, Grover's algorithm [Gro96], a.k.a. quantum search, is another quantum algorithm that can be generically applied to cryptosystems. Fundamentally, it reduces the search time in an unsorted database from $\mathcal{O}(N)$ to $\mathcal{O}(\sqrt{N})$. A straightforward example would be finding the inversion of a hash function or a brute-force attack on ci-

phers. It effectively forces us to double the length of the key for the same security guarantee.

Although quantum computers, even today, have not been commercially available, the mere possibility is sufficient to breed concerning thoughts and speculations (e.g., "store now, decrypt later" attack for data that needs to stay secret in a long time frame). After all, those hard problems have become ubiquitous and trustworthy in our digital lives. The dire consequences cannot be overestimated if we are not well prepared. Therefore, cryptographers have begun the pilgrimage in the search for quantum-safe cryptographic algorithms. More importantly, the replacements must also be practical; that is, they can be implemented on classical computers and cater to commercial needs. This research field is called *Post-quantum Cryptography*. It should be noted that even if (practical) quantum computers are only our preoccupation with a distant possibility, cryptography still greatly benefits. As a science, the quantum threat is a profound and compelling motivation for cryptographers to explore and bring forward novel, more secure candidates that only strengthen the confidence and diversity in our cryptography toolbox.

1.5.2 Post-quantum hardness assumptions

Over the years, we have seen the rapid growth of several branches in Post-quantum Cryptography. Culminating in 2016, NIST initialized the process of standardizing post-quantum cryptography (PQC) algorithms. This initiative aims to identify and standardize cryptographic algorithms that are secure against both classical and quantum computers, ensuring the long-term security of sensitive data in the future when powerful quantum computers may become a reality. The standardization process involved a public competition in which experts submitted and evaluated cryptographic algorithms to find the best proposals for standardization. This rigorous process is crucial for establishing robust and reliable cryptographic standards that protect our information in the post-quantum era. The applications were diverse and can be grouped into the following.

- Code-based Cryptography
- Lattice-based Cryptography
- Multi-variate Cryptography
- Hash-based Cryptography
- Isogeny-based Cryptography

In this thesis, we focus only on code-based cryptosystems, those that rely on hard problems in coding theory, such as *Syndrome Decoding Problem* (SDP), *Code Equivalence Problem*, or *Learning Parity with Noise Problem* (LPN). Notably, McEliece

[Ber+20], BIKE [Ara+23], and HQC [Mel+23a] are representatives of code-based cryptography that persist in later rounds of the standardization process (round 4). In March 2025, it was announced that HQC is among the algorithms selected to be standardized in the future, highlighting the importance of code-based cryptography. The *computationally hard* problem underpinning the security of the aforementioned schemes is the SDP. For a more complete survey, we direct the readers to [Ber09].

Lattice-based cryptography enjoyed popularity, featuring in many selected algorithms to be standardized (Crystals-Kyber, Crystals-Dilithium, and Falcon). However, code-based cryptography remains an active field of research because of its reliability and understanding by the research community. At the time of writing, another call for *Post-quantum Additional Signature Scheme* is ongoing, where we are witnessing a plethora of code-based candidates, such as [Bal+24a; Bal+24b; Ban+23]. This can be seen as an attempt to diversify our post-quantum cryptography profile. The new wave of code-based proposals sees the introduction of many novel coding theory problems that are promising in both security and efficiency. In particular, the last two publications focus on the *Restricted Syndrome Decoding* problem (RSDP), pioneered by [Bal+20a].

1.6 Motivation of the thesis.

Beyond the code-based cryptosystems like PKE, KEMs, or digital signature schemes in the NIST processes, there are many other useful code-based applications. For instance, we will see coding theory in ciphers, wPRFs constructions, or lightweight authentication protocols. Although their security foundation is reliable, we recognize that they are not well understood, and novel designs can be vulnerable to special attacks. For example, implementation and design choices significantly and negatively impact security. In other words, we are invested in the empirical security aspect of cryptosystems. One key aspect of this part of the thesis is the use of several other techniques in conjunction with traditional cryptanalysis algorithms (*BKW* or *Information-Set Decoding* (ISD)). Such techniques aim to exploit design weaknesses that are often overlooked. Therefore, in the first part of the publication, we explore possible ways to attack such constructions to improve the security estimate and confidence for various code-based primitives.

Similarly, the popularity of several new problems (e.g., RSDP) in coding theory also brings interesting challenges. Although promising in terms of efficiency, the settings in which they are employed are limited. Hence, it can be hard to have a complete picture of their concrete hardness. To illustrate, we have a good idea of the traditional SDP due to its extensive and reliable track record of cryptanalysis, which provides different optimal algorithms for various settings. The same cannot be said for RSDP. Hence, the last part of the thesis is devoted to adapting RSDP to well-known designs, giving rise to an efficient, lightweight authentication pro-

tocol, and proposing a new cryptanalytic approach to this uncharted territory of the RSDP setting.

Somewhat separated, Paper IV deals with the "hardcore" SDP problem. As noted above, it is quite challenging to develop a breakthrough in solving SDP. However, we are curious to explore other approaches, such as applying sieving ideas in lattice-based cryptography, to incorporate into the framework of ISD algorithms.

1.7 Outline

The thesis consists of two parts. Part I, including this chapter, is the overview of the research field. Following is Chapter 2, where we review the technical preliminaries of code-based cryptography. In particular, we will see, e.g., coding theory background, computationally hard problems in coding theory, their state-of-the-art cryptanalysis, and a few notable code-based candidates in the NIST post-quantum cryptography process. Chapter 3 describes an interesting research subfield of code-based cryptography called *Learning Parity with Noise*, and its role in lightweight cryptography. Finally, we conclude the first part of the thesis with a summary of the thesis contributions and conclusions in Chapter 4. Part II consists of all publications in the thesis.

Code-based Cryptography

Erroneous messages and data have always been inevitable in communications. The extent to which the intended information is corrupted varies and can be due to multiple reasons. The most common examples are that the medium (in which the communications are carried out) is disturbed, such as satellite communication, physical data storages are damaged, transmitted signals are interfered with, and so on. Therefore, robust error-correcting methods are vital in such scenarios. The (rather ingenious) principle is to add *redundancy*, or *encoding* the data before transmission, and the receivers will reconstruct, or *decode*, to obtain the error-free information. However, this comes at the cost of lowering the transmission throughput as we need more symbols per information bit in our communication.

Assume we have a message of k binary symbols, i.e., an element $\mathbb{F}_2^k$. A *linear code* is simply a subspace of $\mathbb{F}_2^n$. A linear code is often represented by its generator matrix $\mathbf{G} \in \mathbb{F}_2^{k \times n}$. That is, it will map a *message* $\mathbf{m} \in \mathbb{F}_2^k$ into a codeword $\mathbf{c} = \mathbf{mG}$. After transmission through a noisy channel, the receiver might obtain an erroneous copy $\mathbf{y} = \mathbf{c} + \mathbf{e}$. The task on the receiving end is to recover $\mathbf{m}$. We often describe the channel in a simplistic and probabilistic manner, e.g., *Binary Symmetric Channel* with parameter $0 < \epsilon < 1$. This medium can flip a bit with probability ϵ. Due to this nature, the principle of decoding is to look for the *most likely* codeword. That is, we look for a codeword $\mathbf{c} \in \mathcal{C}$ that maximizes the probability

$$\Pr[\mathbf{y} \text{ received } | \mathbf{c} \text{ sent }].$$

A decoding method that solves the above maximization problem is called *Maximum-likelihood decoder* (ML). We can reasonably assume that ϵ is quite small. In other words, the error $\mathbf{e}$ only has a few bits set to 1 on average. Hence, we intuitively look for the "closest" codeword in $\mathcal{C}$ to $\mathbf{y}$ and turn the ML decoder into a *Minimum-distance decoder* (MD), for which we provide a mathematical formulation in this chapter.

In error detection and error correction, coding theory provides the foundation for constructing efficient codes, e.g., reasonable redundancy and efficient retrieval of information. In other words, we are looking for robust codes that provide robust

encoding/decoding of the information. Fortunately, there are families of codes for which we have efficient decoding algorithms. Codes that use special algebraic structures are, e.g., the Reed-Solomon codes [RS60], or its generalization Goppa codes [Gop81]. On the other hand, we also have probabilistic-decoding codes such as convolutional codes [Eli55], LDPC [Gal62], and so on.

Interestingly, the same idea can be applied to cryptography. By deliberately perturbing a message (e.g.), only the intended receiver with an efficient decoding algorithm can correctly retrieve it. However, if the structure of the code is known to an adversary, he or she is equally capable of reading the plaintext. Therefore, in code-based cryptography, it is usually the case that characteristics (e.g., generator matrix or parity-check matrix) of the code are further obfuscated, making the code appear indistinguishable from a randomly generated one. In that sense, an adversary without trapdoor knowledge might as well attempt to decode a *random* linear code. Fortunately, as we will see, it is an intractable problem.

In this chapter, we will go through the coding theory and related hard problems that allow code-based cryptography to be one of the fundamental pillars of post-quantum cryptography. We briefly look at the history of cryptanalysis from which we draw resounding confidence in code-based cryptosystems. In addition, given its history and extensive research, many variants of the original decoding problem have been proposed to further expand the width of applications where code-based cryptography can provide solutions.

2.1 Coding theory

Let $\mathbb{F}_q$ be a finite field and n, k be integers.

Definition 1 (Linear codes). *A $[n,k]_q$ linear code, denoted by $\mathcal{C}$, is defined as a k-dimensional linear subspace of $\mathbb{F}_q^n$. A vector element* $\mathbf{c}$ *of a code is called a codeword.*

Besides its dimension and length, a $[n,k]_q$ code $\mathcal{C}$ is further characterized by its code rate $R := k/n$ and redundancy $n-k$. Equivalently, we can say that one transmits $1/R$ symbols per symbol of information. Since it is a linear subspace of $\mathbb{F}_q^n$, we can succinctly represent a code by its *generator matrix*, often denoted by $\mathbf{G}$ and

$$\mathcal{C} = \{\mathbf{m}^\intercal \mathbf{G} | \mathbf{m} \in \mathbb{F}_q^k\}.$$

Representations of a code In general, there are many generator matrices that describe the same code $\mathcal{C}$. However, one particularly useful form is the *the systematic form*, that is, if $\mathbf{G} = \begin{pmatrix} \mathbf{I}_k & \mathbf{A} \end{pmatrix}$ for some $\mathbf{A} \in \mathbb{F}_q^{k \times (n-k)}$. In this case, we observe that the information is now stored on the first k symbols of the codeword. Analogously, we can also define $\mathcal{C}$ via its *dual code*, denoted by $\mathcal{C}^\intercal$, of which the generator matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ is defined as the *kernel* of $\mathbf{G}$. It follows that $\forall \mathbf{c} \in \mathcal{C}, \mathbf{H}\mathbf{c}^\intercal = \mathbf{0}$ and $\mathbf{H}$ is usually referred as a *parity-check matrix* of $\mathcal{C}$. Let

$\mathbf{x}$ be a vector in $\mathbb{F}_q^n$, the *syndome* of $\mathbf{x}$ w.r.t the code $\mathcal{C}$ with parity-check matrix $\mathbf{H}$ is defined as $\mathbf{s} = \mathbf{H}\mathbf{c}^\intercal$. Moreover, if $\mathbf{G}$ is given in its systematic form, then $\mathbf{H} = \begin{pmatrix}\mathbf{I}_{n-k} & \mathbf{A}^\intercal\end{pmatrix}$. Evidently, all codewords of $\mathcal{C}$ have syndrome $\mathbf{0}$.

In $\mathbb{F}_q^n$, we can define the well-known *Hamming weight*, written as $\omega_H(\cdot)$, as the number of nonzero symbols of a vector. We then straightforwardly define *Hamming distance* between two vectors as $d_H(\mathbf{x}, \mathbf{y}) = \omega_H(\mathbf{x} - \mathbf{y})$. Essential for a code, especially for many hard coding problems, is a special property called *minimum distance*.

Definition 2 (Minimum distance). *Let $\mathcal{C}$ be a code. The minimum distance of $\mathcal{C}$ is*

$$d := \min_{\substack{\mathbf{x},\mathbf{y}\in\mathcal{C}\\ \mathbf{x}\neq\mathbf{y}}} d_H(\mathbf{x}, \mathbf{y}) = \min_{\substack{\mathbf{x}\in\mathcal{C}\\ \mathbf{x}\neq\mathbf{0}}} \omega_H(\mathbf{x}).$$

The minimum distance d is a crucial property of a code. In particular, a code $\mathcal{C}$ with a minimum distance d can be uniquely corrected up to $\delta = \lfloor \frac{d-1}{2} \rfloor$. This upper threshold is called the *error-correction capability* of $\mathcal{C}$, and this quantity has significant implications for both communication and cryptographic purposes. In communication, we understandably want our code to have a large minimum distance d without adding too much redundancy (i.e., high code rate), which is often a trade-off scenario. This relation between the minimum distance and the code rate can be seen in the next definition.

Definition 3 (Gilbert-Vashamov bound). *Let n, k and d be positive integers such that*

$$\sum_{i=0}^{d} \binom{n}{i} (q-1)^i \leq q^{n-k}.$$

Then, there exists a $[n, k]_q$ linear code with minimum distance d. The largest d that satisfies the above inequality is often referred to as the Gilbert-Vashamov distance d_{GV}.

We call a $[n, k]_q$ linear code *random* if it does not have any particular mathematical structure. A random linear code has a high probability of having a "good" minimum distance, close to d_{GV}. However, the proof is not constructive. Therefore, constructing a specific (and hopefully useful) code to achieve such a distance is not an easy task. In addition to coding theory, d_{GV} is pivotal in studying the hardness of cryptographically relevant coding problems.

2.1.1 Coding Problems

We visit the coding problems that lie at the heart of code-based cryptography and discuss their hardness.

Problem 1 (Decoding Problem (DP)). *Given a $[n, k]_q$ code $\mathcal{C}$, and $t \leq n$ be a positive integer. Let $\mathbf{y} = \mathbf{c} + \mathbf{e} \in \mathbb{F}_q^n$, for some $\mathbf{c} \in \mathcal{C}$, and $\omega_H(\mathbf{e}) = t$. Find $\mathbf{e}$.*

Note that Problem 1 can be defined with different metrics other than the Hamming weight. However, the goal of error correction was originally to find the *closest* codewords, as in the corrupted message $\mathbf{y}$ differs by the smallest number of coordinates from the original $\mathbf{c}$. Therefore, Hamming weight is by far the most popular setting to be considered.

Problem 2 (Computational Syndrome Decoding Problem (SDP)). *Given a* $[n, k]_q$ *code* $\mathcal{C}$ *with the parity-check matrix* $\mathbf{H}$*, and* $t \leq n$ *be a positive integer. Let* $\mathbf{y} \in \mathbb{F}_q^n$*, and* $\mathbf{s} = \mathbf{H}\mathbf{y}^\intercal$ *. Find* $\mathbf{e}$ *such that* $\omega_H(\mathbf{e}) = t$ *and* $\mathbf{H}\mathbf{e}^\intercal = \mathbf{s}$*.*

The decisional version of SDP is to answer if any of such $\mathbf{e}$ exists. It was shown that the decisional and computational variants are polynomially equivalent [AB09] and the decisional problem is NP-complete [BMT78b], in the worst case. Therefore, throughout this thesis, the reader may encounter statements in the nature of the SDP being NP-complete, despite this notion only applying to decision problems. In addition, it can be readily checked that the two above problems are again equivalent. Indeed, suppose $\mathbf{G} \in \mathbb{F}_q^{k\times n}$ is a generator matrix of $\mathcal{C}$, we can readily produce a parity-check matrix $\mathbf{H}$. Since $\mathbf{y} = \mathbf{c} + \mathbf{e}$, it follows that $\mathbf{H}\mathbf{y}^\intercal = \mathbf{H}\mathbf{c}^\intercal + \mathbf{H}\mathbf{e}^\intercal = \mathbf{H}\mathbf{e}^\intercal$. Therefore, if a weight-t error $\mathbf{e}$ can be found with its syndrome $\mathbf{s} = \mathbf{H}\mathbf{y}^\intercal$, it is also a solution to the Decoding Problem.

However, it should be clear that given an SDP instance, its hardness depends on several factors (worst case vs average case). For instance, does the code have a special structure that can be exploited, or does the target Hamming weight t somehow make the problem easier? As we briefly discussed error-correcting codes, there are codes for which, up to some Hamming weight t, we do have efficient decoding algorithms. Such codes are fundamental to cryptography. After all, we still need to reveal the original message. However, for cryptography, in particular cryptanalysis, we ensure that the parity-check matrix does not reveal any information about the underlying code $\mathcal{C}$ (except for the intended receiver). Therefore, in the next section, we are interested in *generic decoding algorithms*. That is, we are challenged with the task of decoding random input codes. The hardness of such a problem now relies on the relation between t and the Gilbert-Vashamov distance d_{GV}. The research direction of generic decoding, called *Information-set Decoding*, has been fruitful over the years [Pra62; Leo88; Dum91; Ste88; FS09; BLP08; MMT11a; Bec+12a; BM17b; BM18]. Although we have identified the error rate region where SDP is easier, for a well-chosen t (e.g.,$q = 2$, and $t/n < (1 - k/n)$), the best SDP-solving algorithm is still exponential.

2.2 Information-set Decoding Algorithms

In this section, we briefly go through the code-based cryptanalysis efforts over the years. Obviously, a brute-force attack that requires an order of $\binom{n}{t}$ operations is to be avoided. However, with some simple algebra, we encounter the first non-trivial improvement in solving SDP called Prange Information-set Decoding algorithm

[Pra62]. For the sake of simplicity, from now on, we will mainly be concerned with the binary case $q = 2$ (unless otherwise mentioned), and we omit the transposition symbols when they are clear from the context.

Prange ISD algorithm

Definition 4 (Information set). *Let $\mathcal{C}$ be a $[n, k]_q$ code with a generator matrix $\mathbf{G}$. A subset $\mathcal{I} \subset [\![1, n]\!]$ is called an information set of $\mathcal{C}$ if the collection of columns of $\mathbf{G}$, indexed by $\mathcal{I}$, is linearly independent.*

Let a SDP instance be given by a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_2^{n-k}$, and a positive integer $t \leq n$. Essentially, the Prange algorithm chooses an information set $\mathcal{I} \subset [\![1, n]\!]$ of size k and places a bet that the projection of the error vector $\mathbf{e}$ onto indices in $\mathcal{I}$ is all zero. To detect if this is indeed the case, we send these columns to the left of the matrix and perform a Gaussian elimination. Equivalently, we can represent the action by a permutation $\mathbf{P}$ and transform the problem into

$$\mathbf{He} = \mathbf{HPP}^{-1}\mathbf{e} = \overline{\mathbf{H}}\mathbf{e}' = \mathbf{s},$$

where the first k column of $\overline{\mathbf{H}}$ are independent (information set). Gaussian elimination yields an invertible $\mathbf{Q} \in \mathbb{F}_2^{(n-k)\times(n-k)}$ and $\mathbf{Q}\overline{\mathbf{H}}\mathbf{e}' = \mathbf{Qs} = \mathbf{s}'$. We can rewrite the equation as

$$\begin{pmatrix}\mathbf{H}' & \mathbf{I}_{n-k}\end{pmatrix}\mathbf{e}' = \mathbf{s}'. \tag{2.1}$$

Suppose we have made a correct bet with $\mathbf{P}$, that is, we have made the first k coordinates of $\mathbf{e}$ all 0. Then, we have $\mathbf{e}' = (\mathbf{e}_1, \mathbf{e}_2) = (\mathbf{0}, \mathbf{e}_2) \in \mathbb{F}_2^k \times \mathbb{F}_2^{n-k}$, and it must hold that $\omega_H(\mathbf{e}_2) = \omega_H(\mathbf{s}') = t$. If the weight check of $\mathbf{e}_2$ is not fulfilled, we go back and choose another information set.

As we can see from Algorithm 2.1, the workload we have to perform for each iteration (permutation) is not computationally expensive. Therefore, the complexity of the algorithm is primarily dictated by the (inverse) success probability of each iteration, factored by the cost for each iteration.

For each randomly selected permutation to achieve the correct weight distribution, we need to precisely choose k out of n coordinates for which the error vector is 0. Therefore, the success probability of each permutation for the Prange ISD, called $\mathrm{Pr}_{\text{iter-PR}}$, can be expressed as

$$\mathrm{Pr}_{\text{iter-PR}} = \frac{\binom{n-k}{t}}{\binom{n}{t}}. \tag{2.2}$$

Gaussian Elimination We now discuss the Gaussian elimination with more nuances. First, in the algorithm description, we note that there is an innocuous-looking loop before the enumeration steps that we have glossed over, indicating

Algorithm 2.1: Prange ISD

Input: Parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$, syndrome $\mathbf{s} \in \mathbb{F}_2^{n-k}$, and an integer $t \leq n$

Output: A vector $\mathbf{e} \in \mathbb{F}_2^n$, $\omega_H(\mathbf{e}) = t$, and $\mathbf{He} = \mathbf{s}$.

```
while true do
    repeat
        P ← a random permutation ;
        H̄ ← HP;
    until The last n − k columns of H̄ is full rank;
    (H′  I_{n−k}) ← systematic form of H̄ with Q ∈ F_2^{(n−k)×(n−k)};
    s′ = Qs;
    if ω_H(s′) = t then
        Return: P^{−1}(0_{1×k}, s′)
```

that the transformed matrix is not always admissible. Indeed, there is no guarantee that the last $n-k$ columns of $\bar{\mathbf{H}}$ are of full rank. There is, in fact, only roughly a $1/4$ chance for such a matrix, adding a factor 4 to the probability of an iteration. Canteaut and Chabaud [CC98] suggested that it is not necessary to pick an entirely different information set. It suffices to change one index to go to another permutation. This idea is later generalized by [BLP08], where we can simultaneously change more indices. However, we have introduced dependencies between iterations, yielding a different probability distribution. Such a scenario can be modeled by Markov chains, and more detailed estimates can be derived. We refer the readers to more in-depth analysis [CC98; BLP08; Pet10]. In our brief analysis, we disregard these techniques as they are outside the scope of the thesis.

In literature, the standard cost of Gaussian elimination is set as $(n-k)\cdot k^2/2$. However, there is no shortage of improvements for this step, including the pivot reuse techniques above. Our goal is to have a succinct overview of the ISD algorithms, and for all the involved publications, we chiefly investigate the parameter regimes where Gaussian elimination does not have a significant impact on our complexity estimation. Therefore, we (naively) abuse the notation by universally calling the cost of Gaussian Elimination, including transforming $\mathbf{s}$ to $\mathbf{s}'$, in all algorithms that follow by C_{Gauss}.

Moreover, for "simple" operations such as XOR-ing vectors and checking the Hamming weight, we often conflate with the notation W - the work factor to perform such operations without getting too deep into practical implementations. [1] When comparing algorithms, especially for concrete performances, it is, however,

[1] They are often n, up to a small constant.

crucial to unify them under the same framework. We refer the readers to more thorough analyses, such as [Bal+19; Lön14; Meu13].

Theorem 1. *The complexity of Prange ISD, called* $C_{\text{ISD-PR}}$*, is*

$$C_{\text{ISD-PR}} = \min_{p}\{\text{Pr}_{\text{iter-PR}}^{-1} \cdot C_{\text{iter-PR}}\} \tag{2.3}$$

where $\text{Pr}_{\text{iter-PR}}$ *is given by Equation (2.2), and*

$$C_{\text{iter-PR}} = C_{\text{Gauss}} + W_{\text{sol}}.$$

The second term, W_{sol}, of $C_{\text{iter-PR}}$ represents the work factor for checking the weight and reconstructing the solution.

Lee-Brickell ISD algorithm

We can observe that there are many SDP instances for which Prange ISD does not provide an optimal approach. Indeed, when the code rate is high, as in McEliece where $R \approx 1/2$ and the error rate is in the order of $\log n$, it is exceedingly unlikely that one comes by a good permutation $\mathbf{P}$. In other words, the success probability is heavily penalized. Therefore, Lee and Brickell [LB88] generalized Prange's simple idea by allowing some small weight p that can be present in $\mathbf{e}_1$. Consequently, we have to enumerate all $\binom{k}{p}$ candidates of $\mathbf{e}_1$, and check for

$$\omega_H(\mathbf{H}'\mathbf{e}_1 + \mathbf{s}') = t - p. \tag{2.4}$$

Theorem 2. *The complexity of the Lee-Brickell ISD algorithm, denoted by* $C_{\text{ISD-LB}}$*, is given as*

$$C_{\text{ISD-LB}} = P_{\text{iter-LB}}^{-1} \cdot C_{\text{iter-LB}}, \tag{2.5}$$

where

$$P_{\text{iter-LB}} = \frac{\binom{k}{p}\binom{n-k}{t-p}}{\binom{n}{t}},$$

and

$$C_{\text{iter-LB}} = C_{\text{Gauss}} + \binom{k}{p} W,$$

where W denotes the work factor needed for each weight-p vector in Equation (2.4). Therefore, the trade-off proposed by Lee-Brickell is that we can reduce the number of iterations needed at the cost of more computations per iteration.

Leon ISD algorithm

At its core, Lee-Brickell ISD tries to test all possible candidates for each iteration. As one might expect, the majority of them will not yield a solution for Equation

(2.4). Therefore, Leon [Leo88] proposed a preliminary check for the candidates that drastically decreases the amount of "obsolete" work for each inner iteration. In particular, one can further condition that the first ℓ coordinates of $\mathbf{s}'$ are only constituted by $\mathbf{e}_1$. In other words, there is a length-ℓ "strip" of 0 at the beginning of $\mathbf{e}_2$. On average, the expected number of candidates to pass this check is $\binom{k}{p}/2^{\ell}$. However, this does not come for free because the success probability, denoted by $\mathrm{Pr}_{\text{iter-Le}}$ for each iteration, is slightly worse as

$$\mathrm{Pr}_{\text{iter-Le}} = \frac{\binom{k}{p}\binom{n-k-\ell}{t-p}}{\binom{n}{t}}.$$

We skip the complexity theorem for the Leon ISD algorithm as it is similar to Lee-Brickell ISD with an extra parameter ℓ to be optimized.

Remark 1. *Up until this point, we can implement the above algorithms in a memory-efficient fashion. In particular, besides the matrices, it is not necessary to store all the weight-p candidates $\mathbf{e}_1$. Indeed, a simple function that maps an integer to a specific configuration of p ones often suffices. Moreover, the above algorithms are sometimes referred to in the literature as* permutation-based *ISDs, as the number of iterations is often the overwhelming factor. However, the following algorithm marks the beginning of* enumeration-based *ISDs, which all employ* exponential *spatial complexity. As we will see, many enumerating techniques will be involved for each iteration to construct a solution efficiently. Over time, the scale was tipped more towards the cost of each iteration.*

Stern/Dumer ISD algorithm: Collision Decoding

In contrast to Lee-Brickell, Stern/Dumer [Ste88; Dum91] forcibly construct $\mathbf{e}_1$ vectors that satisfy $\mathbf{H}'_{[\ell]}\mathbf{e}_1 = \mathbf{s}'_{[\ell]}$ via a *Meet-in-the-Middle* approach. In particular, we can split up $\mathbf{e}_1 = \mathbf{e}_1^{(0)} + \mathbf{e}_2^{(0)}$ where $\omega_H(\mathbf{e}_i^{(0)}) = p/2$. To be more precise, we distribute p set bits into disjointed partitions of size $k/2$, containing $p/2$ bits each. To ease the notation, one might assume that $\mathbf{e}_1^{(0)} = (*, \mathbf{0}_{1\times\frac{k}{2}})$, and $\mathbf{e}_2^{(0)} = (\mathbf{0}_{1\times\frac{k}{2}}, *)$, where $*$ represents a length-$k/2$ vector of weight $p/2$. The equation for $\mathbf{e}_1$ becomes

$$\mathbf{H}'_{[\ell]}\mathbf{e}_1^{(0)} + \mathbf{s}'_{[\ell]} = \mathbf{H}'_{[\ell]}\mathbf{e}_2^{(0)}. \tag{2.6}$$

To realize the equation, we can construct two lists as follows.

$$\mathcal{L}_1 = \{\left(\mathbf{e}_1^{(0)}, \mathbf{H}'_{[\ell]}\mathbf{e}_1^{(0)} + \mathbf{s}'_{[\ell]}\right) : \omega_H(\mathbf{e}_1^{(0)}) = \frac{p}{2}\}$$

$$\mathcal{L}_2 = \{\left(\mathbf{e}_2^{(0)}, \mathbf{H}'_{[\ell]}\mathbf{e}_2^{(0)}\right) : \omega_H(\mathbf{e}_2^{(0)}) = \frac{p}{2}\}.$$

One can use the "syndrome" for $\mathbf{e}_i^{(0)}$ as the "key" in the lists, and the collisions between the two lists can be found efficiently. The size of each list is $|\mathcal{L}_i| = \binom{k/2}{p/2}$, the expected number of collisions between the two lists is $|\mathcal{L}_i|^2/2^\ell$.

Theorem 3. *The complexity of the Lee-Brickell ISD algorithm, denoted by $C_{\text{ISD-SD}}$, is given as*

$$C_{\text{ISD-SD}} = \min_{p,\ell}\{\Pr_{\text{iter-SD}}^{-1} \cdot C_{\text{iter-SD}}\}, \tag{2.7}$$

where

$$\Pr_{\text{iter-SD}} = \frac{\binom{k/2}{p/2}^2 \binom{n-k}{t-p}}{\binom{n}{t}},$$

$$C_{\text{iter-SD}} = C_{\text{Gauss}} + 2 \cdot \binom{k/2}{p/2} W + \frac{\binom{k/2}{p/2}^2}{2^\ell} W_{\text{sol}},$$

and W, W_{sol} denote the work factor for each vector in the lists (computing the syndrome) and checking for solutions, respectively.

Remark 2. *Compared to the Lee-Brickell ISD algorithm, the Stern/Dumer ISD variant essentially sees a quadratic decrease in the cost of each iteration while the success probability is slightly worse. Moreover, when enumerating $\mathcal{L}_1$ and $\mathcal{L}_2$, we only have to materialize one list. Therefore, the spatial complexity of Stern ISD is $\mathcal{O}(\binom{k/2}{p/2})$.*

Improvement from Finiasz-Sendrier framework

To improve the success probability for each iteration, Finiasz-Sendrier [FS09] proposed a simple tweak in the setup of all previous ISD algorithms. In essence, instead of a full Gaussian elimination, Finiasz-Sendrier used a *partial* Gaussian elimination with parameter ℓ, which yields

$$\begin{pmatrix} \mathbf{H}_1 & 0 \\ \mathbf{H}_2 & \mathbf{I}_{n-k-\ell} \end{pmatrix} \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{s}_1 & \mathbf{s}_2 \end{pmatrix}. \tag{2.8}$$

Note that, contrary to Equation (2.1), the dimensions of blocks and vectors have been changed. In particular, $\mathbf{H}_1 \in \mathbb{F}_2^{\ell \times (k+\ell)}$, $\mathbf{H}_2 \in \mathbb{F}_2^{(n-k-\ell)\times(k+\ell)}$, $\mathbf{e}_1 \in \mathbb{F}_2^{k+\ell}$, $\mathbf{e}_2 \in \mathbb{F}_2^{n-k-\ell}$, and $\mathbf{s}_1 \in \mathbb{F}_2^{\ell}$, $\mathbf{s}_2 \in \mathbb{F}_2^{n-k-\ell}$. As a result, we have the identities

$$\mathbf{H}_1\mathbf{e}_1 = \mathbf{s}_1, \tag{2.9}$$

$$\mathbf{H}_2\mathbf{e}_1 + \mathbf{e}_2 = \mathbf{s}_2. \tag{2.10}$$

However, the benefit of this simple maneuver is twofold. Firstly, we remove the condition for a strip of zeros in the error vectors, and secondly, we allow p error bits to be allocated on $k+\ell$ positions. Overall, they increase the success probability

of each iteration. Again, this is no free lunch as we also have to enumerate slightly more vectors, from $\binom{k/2}{p/2}$ to $\binom{(k+\ell)/2}{p/2}$.

Ball-collision decoding: A generalization of Stern.

There is yet another way to improve the Stern ISD algorithm. Later, Bernstein et al. [BLP11] generalized collision decoding by allowing some weight p' in the previous ℓ-strip of zeroes in the solution. For this length-ℓ error subvector, deploying the same strategy for $\mathbf{e}_1$, we can construct using $\mathbf{e}_1^{(1)} = (*, \mathbf{0}_{1\times\ell/2})$ and $\mathbf{e}_2^{(1)} = (\mathbf{0}_{1\times\ell/2}, *)$, where $*$ represents a vector in $\mathbb{F}_2^{\ell/2}$ of Hamming weight $p'/2$. We then enumerate the triplets (still using the "syndrome" as the key)

$$\mathcal{L}_1 = \{\left(\mathbf{e}_1^{(0)}, \mathbf{e}_1^{(1)}, \mathbf{H}'_{[\ell]}\mathbf{e}_1^{(0)} + \mathbf{e}_1^{(1)} + \mathbf{s}'_{[\ell]}\right) : \omega_H(\mathbf{e}_1^{(0)}) = \frac{p}{2}, \omega_H(\mathbf{e}_1^{(1)}) = \frac{p'}{2}\},$$

$$\mathcal{L}_2 = \{\left(\mathbf{e}_2^{(0)}, \mathbf{e}_2^{(1)}, \mathbf{H}'_{[\ell]}\mathbf{e}_2^{(0)} + \mathbf{e}_2^{(1)}\right) : \omega_H(\mathbf{e}_2^{(0)}) = \frac{p}{2}, \omega_H(\mathbf{e}_2^{(1)}) = \frac{p'}{2}\}.$$

Similarly, a collision between the two lists is checked if they produce a Hamming weight of $t - p - p'$ in the remaining part of the error $\mathbf{e}_2$. For better visualization, a solution found by the ball-collision decoding algorithm has the form:

$\mathbf{e}_1^{(0)}$	$\mathbf{e}_2^{(0)}$	$\mathbf{e}_1^{(1)}$	$\mathbf{e}_2^{(1)}$	$\mathbf{e}_2$
$\frac{p}{2}$	$\frac{p}{2}$	$\frac{p'}{2}$	$\frac{p'}{2}$	$t-p-p'$

In conclusion, this generalization of Bernstein et al. replaces Stern ISD with a much more reasonable assumption, similar to Finiasz et al. It has been shown in an asymptotic analysis that these two variants are equally competitive [MMT11a].[2]

Remark 3. *In the original work, the ball-collision algorithm can be further optimized using $p_1 + p_2 = p$, $p'_1 + p'_2 = p'$, $\ell_1 + \ell_2 = \ell$ for each small subvector.*

Representation technique

Recall that the collision decoding technique splits the weights in $\mathbf{e}_1^{(0)}$ and $\mathbf{e}_2^{(0)}$ to, e.g., the leftmost and rightmost $(k + \ell)/2$ positions of $\mathbf{e}_1$. Using the *representation technique*[3] by Howgrave-Graham and Joux [HJ10], May et al. [MMT11a] proposed a natural generalization by allowing these set bits to be in disjoint sets

[2] Parameters for ball-collision decoding can be transformed to those in Finiasz et al. [FS09]

[3] The technique was introduced in the context of solving the *Subset-sum Problem*.

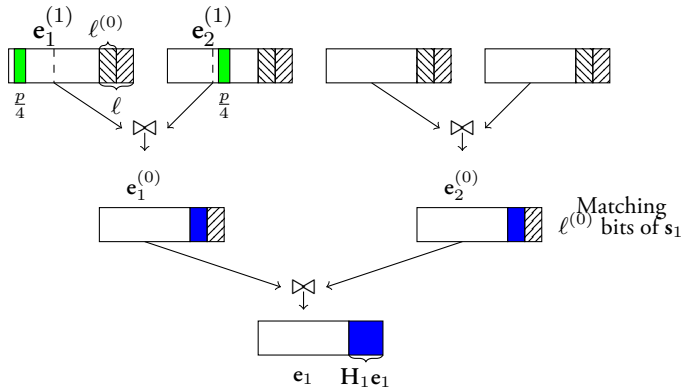


Figure 2.1: A visualization of the error construction in the depth-2 MMT algorithm.

across all indices $\{0, \ldots, k+\ell\}$. To construct $\mathbf{e}_1^{(i)}$, we can expand our algorithm depth by setting

$$\mathbf{e}_1^{(0)} = \mathbf{e}_1^{(1)} + \mathbf{e}_2^{(1)},$$

and similarly for $\mathbf{e}_2^{(0)}$. We now obtain a tree-like construction in Figure 2.1.

Now, we can apply the same collision-decoding technique in Stern for each tree layer to speed up the enumeration process. First, we can further split the ℓ-bit syndrome, correspondingly to $\ell^{(0)}$ and $\ell - \ell^{(0)}$, while only asking for the middle-layer errors to match $\ell^{(0)}$ bits of the syndrome. The rest of the syndrome is reserved for the final matching in the bottom layer, as depicted with the color blue in Figure 2.1. Similar to the previous algorithms, we can also set $\mathbf{e}_i^{(1)}, i = 1, 2$ (top layer) of Hamming weight $p/4$. In particular, they can be from leftmost and rightmost $(k+\ell)/2$ positions of $\mathbf{e}_1^{(0)}$, as long as they do not overlap with the construction for $\mathbf{e}_2^{(0)}$.

The MMT algorithm offers two improvements. Since the p error positions are now not restricted to different regions, the success probability per iteration increases. Indeed, we have $\binom{k+\ell}{p}$ instead of $\binom{(k+\ell)/2}{p/2}^2$ in the success probability expression. Moreover, given a valid solution $\mathbf{e}_1$, there are $R = \binom{p}{p/2}$ ways to form pairs $(\mathbf{e}_1^{(0)}, \mathbf{e}_2^{(0)})$. However, it suffices to construct just **one** of them. Therefore, we can control the list size in each layer so that, on average, 1 out of R solutions survives. The common practice in literature is to set $\ell^{(0)} \approx \log R$. In other words, the enumeration process is further sped up by employing smaller lists. Becker et al. [Bec+12a] further perfected the use of the representation technique in solving SDP by allowing overlapping error positions across layers in the algorithm, resulting in the BJMM variant.

Remark 4. *For the sake of simplicity, we only present the depth-2 version of these al-*

*gorithms. It is certainly possible to recursively expand the depth. However, for relevant code-based cryptography parameters, we are operating with a relatively sparse error vector. Therefore, it is not likely that the error can be overly decomposed in the above manner. Depth-*2 *or* 3 *variants usually yield the best results.*

The MMT/BJMM algorithm further tips the scale of complexity balance towards the cost per iteration. In addition, these variants employ exponential memory usage, which can be enormous at times. Therefore, recent analyses have taken into account the cost of memory access [Bal+19; EB22c]. Again, we skipped over many details of the representation technique. For instance, the choice of partitions in the top list may affect the probability of finding the solutions. For a more detailed dissection, we refer the readers to [MMT11a; Bec+12a] for a better understanding.

Nearest-neighbor search

We recall the Equations (2.9). To summarize the representation technique, we enumerate the desired error $\mathbf{e}_1$ in a binary tree, wherein each layer of the tree, we look for the exact matching of the syndrome (first $\ell^{(0)}$, then ℓ bits). However, the second identity gives rise to another interpretation.

$$\mathbf{H}_2(\mathbf{e}_1^{(0)} + \mathbf{e}_2^{(0)}) = \mathbf{s} + \mathbf{e}_2.$$

Since $\mathbf{e}_2$ is unknown, but its Hamming weight is $t - p$ (assuming a correct permutation), one can rephrase the problem as: finding $\mathbf{e}_1^{(0)}$ and $\mathbf{e}_2^{(0)}$, such that their syndromes (w.r.t $\mathbf{H}_2$) are *close* to each other (i.e., exact match but on $t - p$ positions). Still using the binary-tree construction, one can adapt the algorithm to: instead of finding exact matching on $\ell^{(0)}$ syndrome bits on the middle layer, we find $\mathbf{e}_1^{(1)}$ and $\mathbf{e}_2^{(1)}$ so that their syndrome differs on $t^{(0)}$ positions. The same arguments apply to the other side of the tree (i.e., $\mathbf{e}_2^{(0)}$). Each merge of pair of lists is now precisely an instance of the *Nearest-neighbor Search*.[4]

Definition 5 (Nearest-neighbor (NN) Problem). *Let* $n \in \mathbb{N}, 0 < \alpha < 1/2$, *and* $0 < \lambda < 1$. *In the* (n, α, λ)*-NN problem, we are given two lists* $\mathcal{L}, \mathcal{R}$ *of equal size* $2^{\lambda n}$ *with uniform and pairwise independent vectors. If there exists a pair* $(\mathbf{x}, \mathbf{x}') \in \mathcal{L} \times \mathcal{R}$ *with Hamming distance* $\omega_H(\mathbf{x}, \mathbf{x}') = \alpha n$, *we are asked to output a list that contains* $(\mathbf{x}, \mathbf{x}')$.

May and Ozerov, in [MO15], proposed their NN-Search algorithm, which was later used extensively by Both and May [BM17b; BM18] in their ISD algorithms. Besides the NN-Search algorithm, there have been several studies in solving the NN Problem or its variants (see [IM98; EKZ21; Ess23; Duc+24]).

[4] The problem is not only relevant in our context but to many other disciplines such as machine learning, data science, or bioinformatics.

The Both-May variant and later advancements that optimize the NN-Search subroutine achieve impressive asymptotic complexity improvement. However, it is often hard to estimate the concrete performance due to the polynomial overhead in the complexity expression of NN-Search.

Statistical Decoding

Due to its significance, SDP is subjected to many other approaches besides the line of ISD algorithms. Notably, before the recent proliferation in enumeration-based ISD, *Statistical Decoding* was proposed by [Jab01]. In simple words, we tackle the SDP under the lens of probability. As we previously discussed, the SDP is equivalent to: given a code $\mathcal{C}$ with parity-check matrix $\mathbf{H}$, look for weight-t error $\mathbf{e}$, such that $\mathbf{He} = \mathbf{Hy}$ where $\mathbf{y}$ is a noisy codeword of $\mathcal{C}$. For each (row) parity $\mathbf{h}$ of $\mathbf{H}$, we rewrite

$$\sum h_i e_i = \mathbf{h} \cdot \mathbf{y}. \tag{2.11}$$

Assume the first position h_1 is non-zero, we can split the left side of the equation into:

$$e_1 + \sum_{i \neq 1} h_i e_i = \mathbf{h} \cdot \mathbf{y}.$$

Since $\mathbf{e}$ is of Hamming weight $t - 1$ in the second term of the equation, if the parity $\mathbf{h}$ is also of (very) small Hamming weight, it is possible to have a detectable bias ϵ. With $\mathcal{O}(1/\epsilon^2)$ such parity, one can recover e_1 with high probability.

Therefore, the crux of Statistical Decoding is to find low-weight parity checks $\mathbf{h}$. However, it is not hard to see that one requires quite many of them, specifically when the bias is exponentially small (which is usually the case). Equivalently, we are asking for codes to have a very small code rate. Optimistic arguments showed that they are not competitive compared to previous traditional ISD algorithms [DT17]. Recently, Statistical Decoding has enjoyed a resurgence of interest due to breakthrough studies by Carrier et al. [Car+22; Car+24]. Essentially, instead of restricting low Hamming weights on $n - 1$ positions, we can relax by allowing more secret bits to be recovered with the Fourier transform. Specifically, let $\mathcal{P}, \mathcal{N}$ be two disjoint partitions, such that $\mathcal{P} \cup \mathcal{N} = [n]$, we write Equation (2.11) as

$$\sum_{i \in \mathcal{P}} h_i e_i + \sum_{i \in \mathcal{N}} h_i e_i = \mathbf{h} \cdot \mathbf{y}.$$

This observation resembles the *Learning Parity with Noise* problem (which we will cover subsequently): by viewing $\mathbf{x} := (e_i)_{i \in \mathcal{P}}$ and the rest as "noise", we effectively have a noisy product of $\mathbf{x}$ and $(h_i)_{i \in \mathcal{P}}$. Consequently, as $\mathcal{N}$ is smaller than $n - 1$, the Hamming weight can be smaller on $\mathcal{N}$, yielding a more forgiving bias.

In addition, more advanced and efficient methods to find low-weight parity checks are employed (such as the representation technique). As a result, the au-

thors broadened the code rate regions, where Statistical Decoding outperformed the best ISD significantly, covering more cryptographically relevant contexts.

Remark 5. *As we have seen, numerous advancements have been made throughout the years toward solving SDP that provide us with a very good understanding of the security of code-based cryptosystems. However, asymptotically speaking, the improvements from all sophisticated ISD variants (compared to Prange ISD) have been incremental. In particular, the asymptotic complexity for all ISD variants (in the region where SDP is hard) is all* $2^{\alpha n(1+o(1))}$*, where* α *is a function of the code rate and error rate and it is only slowly decreasing with better algorithms.*[5] *That is to say, foundationally, code-based cryptography has been an extremely reliable pillar of cryptography.*

2.2.1 Notable code-based PKEs

McEliece Public-key encryption using hard coding problems was originally proposed by McEliece [McE78], but was often left in the dark due to its rather cumbersome key size. However, it has become a viable candidate for post-quantum cryptography. In broad strokes, we pick an efficiently decodable code $\mathcal{C}$ with a generator matrix $\mathbf{G}$, then make public an obscured version $\mathbf{G}_{pub}$ that appears uniformly random from the perspective of an adversary. Assume two parties, Alice and Bob, and Bob wants to send a message $\mathbf{m}$ with $\mathbf{G}_{pub}$ from Alice. Bob then chooses a weight-t error (at most at the decoding capability of the $\mathcal{C}$), and computes $\mathbf{mG}_{pub} + \mathbf{e}$. Since Alice knows how to decode $\mathcal{C}$, the message $\mathbf{m}$ can be retrieved readily.

The security of McEliece PKE relies on: 1) the public $\mathbf{G}_{pub}$ is indistinguishable from a random matrix and 2) Minimum Distance Decoding is hard on average for a well-chosen t and a random input code. The original proposal was employed with Goppa codes, and the McEliece team in the NIST PKE standardization process chose a variant called Niederreiter [Nie86], which is the dual version of McEliece (Syndrome Decoding Problem, i.e., the error is the plaintext and the syndrome is the ciphertext). Such a ciphertext compression technique yields remarkably small ciphertext sizes for McEliece. Despite rather large public keys, McEliece provides robust security. In addition, in applications where one key can be reused many times, or key generation is not a concern, McEliece can be an excellent choice.

Many subsequent proposals have used other families of codes that provide more compact key representations, such as LPDC codes, Quasi-Cyclic Goppa codes, convolutional codes, and so on. Further cryptanalysis studies pointed out invulnerability in many cases. There were, however, other secure code-based alternatives that appeared in the NIST standardization process.

[5] Statistical Decoding can offer better asymptotic improvement when the code rate approaches 0 [Car+22].

BIKE Rather than using random Goppa codes, BIKE [Ara+23] (Bit-flipping Key Encapsulation) uses the so-called QC-MDPC (Quasi-cyclic Moderate Density Parity Check) in its construction. Not only do QC-MDPC codes have a more compact key representation, but they also repair the vulnerability of using LDPC codes (much sparser parity check matrix). We describe BIKE roughly as follows. Similarly to Niederreiter, the plaintext in BIKE is a sparse error vector, and the ciphertext is its syndrome.

- Let $\mathcal{C}$ be a QC-MDPC code represented by $\mathbf{H} = \begin{pmatrix}\mathbf{H}_0 & \mathbf{H}_1\end{pmatrix}$, where $\mathbf{H}_0$ is invertible. The public key is $\mathbf{H}_{pub} = \mathbf{H}_1\mathbf{H}_0^{-1}$.
- A message is a weight-t vector $\mathbf{e} = (\mathbf{e}_0, \mathbf{e}_1)$ and the ciphertext is $\mathbf{s} = \mathbf{e}_0 + \mathbf{H}_{pub}\mathbf{e}_1$.
- The receiver compute $\mathbf{s}\mathbf{H}_0 = \mathbf{e}_0\mathbf{H}_0 + \mathbf{e}_1\mathbf{H}_1$ and applies a designated syndrome decoder to retrieve $\mathbf{e}_0$.

In contrast to McEliece, BIKE suffers slightly from *Decoding Failure* (small probability) due to the probabilistic nature of the decoding algorithms (Bit-flipping decoders [Gal62]). Note that the error weight t employed here is also smaller than McEliece (in the order of $\sqrt{n}$, compared to $n/\log(n)$). In addition, due to the cyclic structure of the parity-check matrix, given a syndrome $\mathbf{s}$ and its i-th cyclic shift $\mathbf{s}_i$, there exists a weight-t error vector $\mathbf{e}_i$ that solves $\mathbf{H}\mathbf{e}_i = \mathbf{s}_i$ and it can be used to reconstruct the original solution $\mathbf{e}$. Therefore, the message-recovery attack becomes 'slightly' easier with more potential solutions for the SDP problem (as many as the size of $\mathbf{H}_i$). A technique called *Decoding-One-Out-of-many* (DOOM), providing speed-up for ISD algorithms, was proposed by Sendrier et al. [Sen11]. However, the trade-off is that for the same security level, the key size of QC-MDPC codes can be hundreds of times smaller than that of Goppa codes.

HQC The Hamming Quasi-Cyclic (HQC) employs two codes: 1) a public, agreed-upon $[n, k]$ code $\mathcal{C}$ with decoding capability δ, and 2) a quasi-cyclic, double circulant $(2n, n)$ code. Other public parameters are quantities t, ω_e, ω_r. We describe very roughly the mechanism of HQC.

- Sample a generator matrix $\mathbf{G}$ for $\mathcal{C}$, and a parity check matrix $\begin{pmatrix}\mathbf{I} & \mathbf{H}\end{pmatrix}$ for the QC code. Set the secret key as $(\mathbf{x}, \mathbf{y}) \in \mathbb{F}_2^n$, both of Hamming weight t, and make public $(\mathbf{H}, \mathbf{s} = \mathbf{x} + \mathbf{H}\mathbf{y}^\intercal)$.
- To encrypt $\mathbf{m} \in \mathbb{F}_2^k$, generate $\mathbf{e}$ of Hamming weight ω_e, $(\mathbf{r}_1, \mathbf{r}_2)$, each of Hamming weight ω_r. Compute

$$\mathbf{u} = \mathbf{r}_1 + \mathbf{H}\mathbf{r}_2^\intercal,$$

$$\mathbf{v} = \mathbf{m}\mathbf{G} + \mathbf{s}\mathbf{r}_2 + \mathbf{e}.$$

- The ciphertext is $\mathbf{c} = (\mathbf{u}, \mathbf{v})$.

The security of HQC relies on the fact that it is infeasible to retrieve $\mathbf{m}$, even if the code $\mathcal{C}$ is public, as the error weight exceeds the code's error-correction capability (for well-chosen t, ω_e, ω_r). However, with the secret key, we can perform $\mathbf{v} - \mathbf{uy}$ and decrypt correctly when

$$\omega_H((\mathbf{x} + \mathbf{Hy}^\intercal)\mathbf{r}_2 - (\mathbf{r}_1 + \mathbf{Hr}_2^\intercal)\mathbf{y} + \mathbf{e}) = \omega_H(\mathbf{xr}_2 - \mathbf{r}_1\mathbf{y} + \mathbf{e}) \leq \delta.$$

Again, the parameters are chosen so that the above holds except for a small probability, which is the decoding failure rate of HQC. The adversary can try to bypass this problem by directly decoding $\mathbf{x}, \mathbf{y}$ from $\mathbf{s}$, but it is also infeasible.

We have omitted many details regarding the above schemes, as we will not focus on them in our work, but rather discuss the underlying SDP assumptions. Therefore, we again refer the readers to the specification documentation for better understanding. Besides the well-known candidates, in the next section, we move into a different approach to employing coding-theory problems efficiently and securely: SDP with special noise.

2.3 Restricted Syndrome Decoding Problem

In the previous section, we have seen a few code-based constructions. Although BIKE and HQC use structure keys for a more compact key representation, it is often a trade-off, and other benchmarks usually suffer [Kuz+23].

Therefore, cryptographers have proposed other variants of SDP that are more competitive in terms of performance without sacrificing too much security. Another interesting research direction is employing SDP with *structured noise*. That is, variants of SDP where the noise is of particular configurations or taken from special sets. An example of the prior is the *Regular Syndrome Decoding Problem* where the noise is divided into blocks with equal Hamming weight. On the other hand, we have the *Restricted Syndrome Decoding Problem* for the latter, which is notable for its application in CROSS - a viable code-based candidate for NIST Post-quantum Additional Signature Schemes. Note that these are not the only examples. For instance, we also have SDP in other metrics such as *rank metric*, *Lee metric*, or the *Permuted Kernel Problem* [Sha89; Bet+24].

In this section, we briefly discussed RSDP, which was first introduced in [Bal+20a]. We reserve a more detailed examination for the later publication of this thesis, and we refer the readers to the original works [Bal+20a; Bal+24c].

Definition 6 (Restricted Syndrome Decoding Problem). *Let $\mathbb{F}_q$ be a finite field and $n, k, t \leq n$ be positive integers. Let $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$, and $\mathbb{E}$ be a $\mathbb{F}_q$ multiplicative subgroup of order z. Find $\mathbf{e} \in (\mathbb{E} \cup \{0\})^n$, such that $\omega_H(\mathbf{e}) \leq t$ and $\mathbf{He}^\intercal = \mathbf{s}$.*

It is closely related to other hard problems. If $z = q$, we have the usual SDP, and if $z = 1$, it is the *Subset-sum Problem*. They are both $\mathcal{NP}-$ complete, and this is also the case for RSDP. The hardness was shown in [Bal+20a] for $z = 2$ and later proved for the general case [Bal+24c]. In addition, the authors in [Bal+20a] also established other well-known coding theory bounds (e.g., GV bound, Singleton bound) in the new setting and studied the behaviors of random codes. For cryptographic applications, they introduced a new RSDP-based identification scheme, which eventually led to CROSS [Bal+24a].

Note that CROSS, as well as the publications in this thesis, revolves around a special form of RSDP called *full-weight* RSDP. In particular, the error vector $\mathbf{e}$ has the maximum Hamming weight ($t = n$).[6] Such subtle difference allows efficient communication and simplified implementation in CROSS.

2.4 Solvers for the Restricted Decoding Problem

Since it is an SDP problem, algorithms in Section 2 can be tailor-made to this new setting. Substantial cryptanalysis efforts have been made in this direction. Baldi et al. [Bal+24c] proposed their Stern and BJMM adaptations to investigate the concrete hardness of RSDP, especially those that are relevant in their signature proposals. Meanwhile, Bitzer et al. [Bit+23] further ventured into scenarios where a too-small restricted set can be used against the system. That being said, with CROSS setting, the cryptanalytic works suggest RSDP can achieve remarkable security with fairly compact parameters, such as $q = 127$ and $z = 7$.

It is only reasonable for us to ask: "So, where is the catch then?". Indeed, intuition tells us that, with more information at hand, the problem should be easier. In fact, structured noise, such as in RSDP or the Regular Syndrome Decoding Problem, allows us to model the problem as a system of equations, and algebraic methods can be applied. However, recent works have shown that this approach is not much more fruitful than its combinatoric counterpart [Bal+24c; BM24].

2.4.1 RSDP in this thesis

In most SDP applications, we deal with a fixed-rate code, i.e., we have a limited number of rows in the generator (or parity-check) matrix. However, LPN is a special case where the code rate can be arbitrarily small by treating the problem as an *Oracle* that gives noisy dot products of the secret and uniformly random vectors (that constitute a code).

In our work, we also examine the RSDP from a slightly different angle, e.g., in its dual form. Similarly to the case of SDP and the LPN problem, we are drawn to situations where the code rate can be rather small. Our motivation is that it can enable RSDP in many other lightweight, resource-constrained cryptographic applications, as LPN has achieved.

[6] This variant is still $\mathcal{NP}-$complete

Definition 7 (RSDP Oracle). *Let p be a prime number and $\mathbb{E} = \{g^i, i = 1, \ldots, z\}$ be a multiplicative subgroup of order z in $\mathbb{F}_p^k$. Fix a secret $\mathbf{s} \in \mathbb{F}_p^k$. The RSDP Oracle gives pairs of samples*

$$\{\mathbf{a} \in \mathbb{F}_p^k, b = \mathbf{a} \cdot \mathbf{s} + e \bmod p\}$$

where $\mathbf{a} \xleftarrow{\$} \mathbb{F}_p^k$ and $e \xleftarrow{\$} \mathbb{E}$.

In Paper V, we employ this form of RSDP in a lightweight authentication protocol that can rival and exceed the performance of LPN-based counterparts. Introducing the RSDP Oracle also invites another generic attack, namely the *linearization* from [AG11]. The abundance of samples can now be exploited to solve the problem in polynomial time, in contrast to the ISD-like algorithms. However, with a careful choice of parameters, the number of required samples can be prohibitively large and impractical. Our motivation is to seek an answer for the middle ground - what can be done in the case where we have many samples but not quite enough for linearization? It resembles the situation with the LF-2 variant of the BKW algorithm, where only a conservative amount of oracle queries can be made. Moreover, for classical SDP, recently *Statistical Decoding* [Car+22; Car+24] has been revisited and improved. By finding low-weight codewords in the dual codes and modeling the noise with heuristic probability arguments, the authors achieved promising results, especially where the code rate can be smaller than "more traditional" instantiations (e.g., McEliece, BIKE, HQC). Their work improved the state-of-the-art cryptanalysis for SDP, covering relevant code rate regimes, and this should be the case for all variants of SDP, including RSDP.

Therefore, in Paper VI, we provide a BKW-style RSDP solver, addressing this exact question. We adapt several of the latest BKW techniques to meet the unique requirements of the new setting. Besides its own intellectual interest, enriching the state-of-the-art cryptanalysis for RSDP, it provides a useful security-measuring tool for further RSDP applications, such as in Paper V.

Learning-based Cryptography

Through advancements in technology, the presence of digital accessories has been woven into the fabric of our everyday lives and activities to an unprecedented extent. However, such conveniences pose unique challenges to cryptography. In particular, the application of cryptography to low-powered, constrained-environment devices. Indeed, as we have seen from the previous chapters, code-based cryptography often seeks out coding problem variants that are more computation and resource-friendly while still achieving respectable levels of security. In this chapter, we will cover such a variant, called the *Learning Parity with Noise* (LPN) problem. It is a well-studied NP-hard problem intimately related to the Syndrome Decoding Problem, highlighted by the fact that there has not been a quantum algorithm that can solve LPN faster than classical ones. Moreover, there have been substantial cryptanalysis works over the years that provide researchers with confidence in employing LPN in various cryptographic constructions. And above all, as we will see shortly, the inherent simplicity of operations involved in LPN makes it a prime candidate for lightweight cryptography.

3.1 Learning-based Assumptions

Let k be a positive integer, $\eta \in (0, 1/2)$ and Ber_η denote the Bernoulli distribution with parameter η. We define the LPN oracle as follows.

Definition 8 (LPN Oracle). *An LPN oracle called $\mathcal{O}_{k,\eta}$, parameterized by k and η, for a secret $\mathbf{x} \in \mathbb{F}_2^k$, returns pairs of the form*

$$\left(\mathbf{g} \xleftarrow{\$} \{0,1\}^k, z = \mathbf{g} \cdot \mathbf{x} + e\right), \tag{3.1}$$

where $e \leftarrow \mathsf{Ber}_\eta$.

In other words, an LPN oracle gives noisy dot products of $\mathbf{x}$ and uniformly random $\mathbf{g}$. We next define the main (hard) problems of this chapter, coming in two flavors: decisional and search variants.

Problem 3 (Decisional LPN). *Given an LPN oracle $\mathcal{O}_{k,\eta}$, the (decisional) Learning Parity with Noise problem, $\mathsf{LPN}_{k,\eta}$, asks to distinguish $\mathcal{O}_{k,\eta}$ output from uniformly random $\mathbb{F}_2^{k+1}$ samples.*

We formalize the hardness of the (decisional) problem. The $\mathsf{LPN}_{k,\eta}$ is said to be (T, δ, Q)-hard if for all algorithms $\mathcal{A}$, running in time T, using Q oracle queries

$$\left|\Pr[\mathcal{A}\left(\mathcal{O}_{k,\eta}\right) = 1] - \Pr[\mathcal{A}\left(\mathcal{U}_{k+1}\right) = 1]\right| \leq \delta.$$

Problem 4 (Search LPN). *Given an LPN oracle $\mathcal{O}_{k,\eta}$, the Search Learning Parity with Noise problem asks for the recovery of* $\mathbf{x}$.

Similarly, the search problem is said to be (T, δ, Q)-hard if for all algorithms $\mathcal{A}$, running in time T, using Q oracle queries, and

$$\Pr[\mathcal{A}\left(\mathcal{O}_{k,\eta}\right) = \mathbf{x}] \leq \delta.$$

The decision LPN problem is $\mathcal{NP}$-hard, and it has been shown that the search and decisional LPN problems are polynomially equivalent [KS06a].

3.1.1 LPN solving algorithms

One can immediately draw parallels between LPN and the previously introduced (binary) Decoding Problem. Indeed, for a fixed amount of queries n, the noisy products essentially form noisy codewords $\mathbf{z} = (z_1, \ldots, z_n)$, of which the generated matrix

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_1^\intercal & \cdots & \mathbf{g}_n^\intercal \end{pmatrix}$$

can be deduced. Therefore, we can view the LPN problem as a Decoding Problem. The only distinction is that the adversary now works with a code with an arbitrarily small code rate. Consequently, all generic decoding algorithms introduced in Chapter 2 apply to LPN. However, Blum, Kalai, and Wasserman [BKW03a] proposed the BKW algorithm to exploit the small code rate. In particular, the BKW algorithm can solve LPN in sub-exponential time $\mathcal{O}(2^{n/\log n})$, requiring as many queries from an LPN Oracle. We briefly look at BKW and several notable improvements since its introduction.

In essence, the original BKW algorithm can be divided into the following steps:

1. **Reduction step:** The BKW algorithm takes parameters t and b. First, it finds all combinations of two columns $\mathbf{g}_i, \mathbf{g}_j$ that cancel out b last positions. This can be done by a *sort-and-match* procedure. In particular, BKW first

distributes all combinations into different equivalence classes, depending on the value of the last b positions (i.e., they have the same projection on these coordinates). Then, a class representative is chosen to combine with others to nullify the last b coordinates. For every combination indexed with i and j, the *observed* symbol $\mathbf{z}_i + \mathbf{z}_j$ is updated.

In summary, the reduction step has produced a new matrix $\mathbf{G}'$ where the last b coordinates of every column are zero. Consequently, the contribution of the last b bits of secret $\mathbf{s}$ is nullified. The trade-off is that every new symbol contains two error bits rather than only one. The BKW algorithm repeats this step t times, eventually removing $b \cdot t$ secret bits to guess while having $2^t \cdot b$ error bits convoluted for each symbol. The bias of the sum of such errors can be estimated using the *Piling-up* lemma [Mat94].

2. **Solving step:** In the final step, the BKW algorithm looks for a column in the (transformed) matrix such that its only first bit is non-zero. That is, the corresponding symbol comes from only the first bit of the secret and (many) errors. The observed symbol is stored, and the reduction step is repeated to find more such occurrences. Once enough symbol is observed, since we know the bias of the convoluted errors, we can determine the first bit of the secret with high probability.

The original BKW algorithm is a very strong theoretical construction as it operates solely on independent samples. One can provide rigorous probabilistic arguments for the algorithm's performance. However, we should note that in the reduction step, we must discard all class representatives, reducing 2^b samples after each reduction step. Hence, the BKW algorithm typically requires an extreme abundance of LPN samples. Leveil and Fouque [LF06b] addressed this issue and improved significantly in their LF-1 and LF-2 variants, which can be summarized by the following points.

- Instead of scanning for only weight-1 columns in the solving step, one can simultaneously guess $k - t \cdot b$ bits of the secret using a fast Walsh-Hadamard transform, thus avoiding querying the LPN oracle multiple times and making use of all samples after the reduction steps.
- Instead of producing only independent combinations in each reduction step, one can combine all vectors in the equivalent class, producing more samples at the cost of introducing dependency (LF-2). With a proper choice of parameters t and b, the number of samples after each reduction step can even be preserved. Thus, it makes BKW a very practical choice when the amount of LPN samples is limited.

Besides the above improvements, there have also been many notable investigations of LPN-solving algorithms over the years, focusing on different regimes.

In particular, if the number of samples allowed is polynomial in k, then the secret can be found in $\mathcal{O}(2^{k/\log\log k})$ [Lyu05]. Alternatively, once we are restricted to only $\Theta(k)$, i.e., a linear sample amount, the best algorithms solve in exponential time (ISD algorithms). Using a similar idea to the ISD algorithm, Bogos et al. [BV15] achieved $\mathsf{poly}(k) \cdot e^{\sqrt{k}}$ for low noise-rate $\eta = 1/\sqrt{k}$.

Covering codes The state-of-the-art LPN algorithm is achieved by Guo et al. [GJL14] with the introduction of *Covering codes* technique and later perfected by Bogos et al. [BV16]. Using coding techniques, Guo et al. [GJL14] introduced an intermediate phase in the BKW algorithm. In particular, they reduced more (remaining) guessing bits for the secret by replacing some coordinates in the samples (after reduction) with perturbed codewords from a carefully chosen code. This step introduces additional errors, which can be estimated depending on the code characteristics. Using this technique, the authors improved security estimates for several LPN-based cryptosystems, such as Lapin, LPN-C, and several variants of HB-like authentication protocols.

3.2 LPN-based lightweight cryptography

Other than being a hard problem that motivates secure cryptographic constructions, LPN is renowned for its simplicity, which is achieved by using only cheap operations. In addition, its close affiliation with the SDP implies that LPN is quantum safe. Therefore, it is unsurprising that LPN is a prime candidate for building environment-constrained, low-powered cryptographic schemes where hardware limitation is an important factor, such as RFID tags. In contrast, these devices often lack the capacity to be employed with standard, well-known primitives like AES.

The use of low-cost devices has become more prominent, which puts increasing importance on suitable lightweight cryptography. Therefore, Hopper and Blum [HB01b] pioneered this research direction with their elegant and minimalistic LPN-based authentication protocol called HB (Figure 3.1). This 2-round protocol, however, was only secured in the relatively weak adversary model (*passive attack*) that was considered in the same work. In particular, the adversary did not have much power besides observing the authentication protocols. Juels and Weis [JW05b] later showed that HB could be easily exploited by the (more realistic) threat, called *active attack*, in which an attacker is equipped with some query power that can send the initial *challenges* of their own choice. They further suggested an augmented version HB+, which added an extra round to HB to resist the more potent adversary. Gilbert et al. [GRS05] further challenged the protocol with a *Man-in-the-middle* (MitM) attack, where the attacker could intercept and alter the communications. They proved that HB+ is vulnerable under this adversarial assumption.

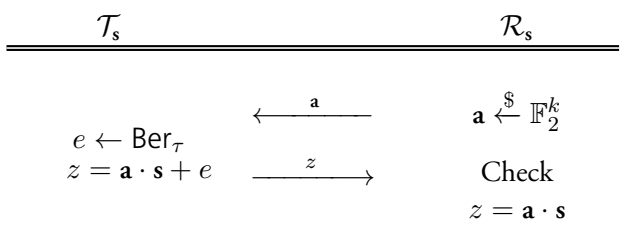


Figure 3.1: One round of the HB authentication protocol with the Tag ($\mathcal{T}$) and Reader ($\mathcal{R}$) sharing a secret $\mathbf{s} \in \mathbb{F}_2^k$.

Subsequently, a proliferation of HB-family authentication patches was proposed [DK07; MP07; LMM08; BC08; GRS08c], attempting to achieve security in the MitM model. Most notable was HB# [GRS08b], which employed a variant of LPN, called Toeplitz-LPN, to reduce communication costs. However, later cryptanalysis works, such as [GRS08a; OOV08], eventually put the LPN-based authentication research against a gloomy backdrop.

Several breakthrough works that resisted MitM adversaries managed to resurrect confidence in LPN-based authentication. In particular, another LPN variant called Subset-LPN was used to construct message authentication codes (MACs) by Kiltz et al. [Kil+11], or Lyubashevski et al. [LM13a] showed a generic MitM-secured authentication construction from any wPRFs with some reasonable properties. In addition, we also see a different approach in [LGQ13b], where the authors devised an elegant encryption scheme to mask LPN samples, thus achieving better security while pushing aggressive LPN parameters.

Besides authentication, one finds LPN in several other scenarios. For example, in Paper I, we investigated an LPN-based stream cipher called Firekite [Bog+21b]. This cipher was preceded by LPN-C by Gilbert et al. [GRS08d] and tackled the problem of generating new randomness (which can be expensive) for LPN. Duc and Vaudenay [DV13a] also demonstrated how to build an LPN-based public-key encryption scheme called Helen. In addition, we also encounter LPN in more foundational cryptographic objects such as wPRFs [Bon+18; Din+21], exploring computation-friendly Multi-party computation. This further shows the potential of the LPN problem in many cryptographically relevant scenarios.

3.2.1 LPN in this thesis

The first part of the publications revolves around devising specialized attacks that exploit unforeseen design weaknesses of several LPN-based cryptosystems, e.g., Firekite, LCMQ, and alternating-moduli wPRFs. Our attacks often employ several creative techniques that can speed up traditional LPN-solving algorithms such as BKW or ISD. In these applications, the LPN problem is often tweaked or used in conjunction with other encryption schemes to mitigate existing issues (expensive randomness generation or security). Although their hardness assumption is

usually substantiated with reasonable assumptions or security reduction proofs, novel designs must be subjected to sufficient cryptanalysis efforts. The security parameters are often chosen according to the best attack for the underlying computational hardness problem. However, several aspects of the design/implementation cannot be contained in the security reductions, which could result in overestimating security levels. Our work ensures a clearer picture of the concrete hardness/security of the LPN problem in these particular settings and helps avoid pitfalls in future LPN-based applications.

Contributions and Conclusions

This dissertation has focused on several novel cryptanalysis techniques of code-based cryptosystems, with the exception of Paper V. The different contributions can be visualized within the area as in Figure 4.1. After this, the contributions are described in more detail in Section 4.1, followed by conclusions in Section 4.2.

4.1 Contributions

In this section, the main contributions of this dissertation are described.

4.1.1 Attacks on the Firekite Cipher

Since its introduction, LPN has been prevalent in many cryptographic constructions that emphasize their efficiency due to the simple nature of the operations involved. Besides popular proposals for lightweight authentication, we also see LPN in sophisticated primitives, such as PKE [Yu+18; DV13a] and stream ciphers [GRS08d] (LPN-C). However, a drawback of various LPN-based encryption schemes is that new *randomness* has to be repeatedly generated, thus adding substantial computational overhead. Bogos et al. [Bog+21b] have recently proposed a novel LPN-based stream cipher called Firekite. Notably, the authors tackled the mentioned adversity by cleverly reserving a (small) part of the output for the next round of keystream generation. Thus, except for the initialization round, they have readily available new random samples due to the intractability of the LPN problem. However, contrary to LPN-C, where an additional error-correcting code is present, Firekite reuses the same (matrix) key for the entire encryption. In addition, Firekite was also promising for its high-performance rate, and the authors showed the prospect of implementing efficient LPN-based encryption in low-end devices.

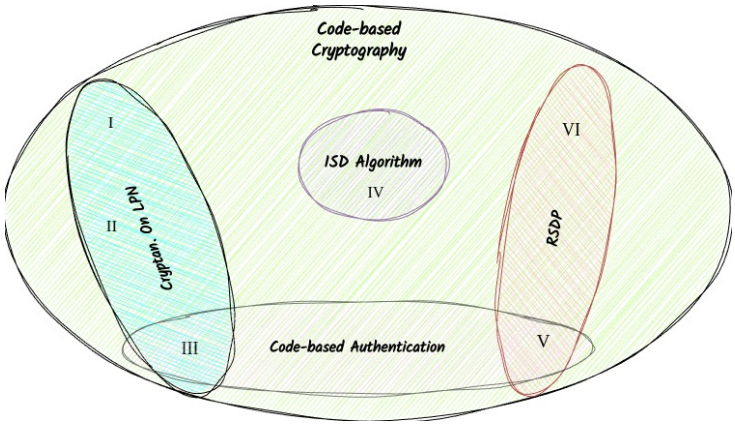


Figure 4.1: Venn diagram of the included papers' contributions

In Paper I, we show how we can exploit such design features as leverage for a distinguishing and key-recovery attack. In particular, since the keystream is generated by the noisy products of vectors with the same matrix, dependencies in the output might leak to the keystream, giving more advantage to a suitable distinguisher. The type of dependency that we investigate is *zero-sums* from k lists, which can be seen as a generalization of the famous *Birthday paradox*. Due to their significance in cryptanalysis, solvers for this problem have been studied extensively, pioneered by Wagner in his seminal work [Wag02]. The algorithm complexity is sub-exponential and requires as many samples (or, in our case, keystreams).

We demonstrate that such a combination will yield an abnormally low-weight sum of the corresponding keystreams and can easily be distinguished. Subsequently, we argue that once we obtain enough anomalies, we eventually recover the key of the cipher with the help of ISD algorithms. Our work provides a much-needed cryptanalytic tool for new LPN-based encryption, ensuring more secure parameters.

4.1.2 Differential cryptanalysis of Mod-2/Mod-3 constructions of binary weak PRFs

Several attempts to employ learning-based assumptions in building lightweight, computationally friendly cryptographic primitives have been made. In particular, foundational cryptographic objects such as wPRFs can be constructed efficiently using LPN in [Bon+18; Din+21]. Cheon et al. [Che+22] showed that such constructions with low Hamming-weight secret keys can be susceptible to their distinguishing attacks and proposed simple fixes. However, the practicality of this attack is questioned as it requires 2^{80} samples, which can be thwarted by simple preventative measures.

In Paper II, we devise yet another attack. We observe that certain low Hamming-weight input combinations yield detectable bias in the corresponding outputs. We employ the *Nearest-Neighbor Search* algorithm to find the combinations efficiently and then compute the exact corresponding bias in the output. The heuristic arguments are verified with simulations, affirming their soundness. The contribution is twofold. First, our attacks are superior to previous work with respect to asymptotic complexity and required wPRFs evaluations. In particular, while Cheon et al.'s work needs an exceptionally large number of evaluations, our approach operates under very practical security scenarios. Secondly, relying on simulations, we observed that the fixes previously proposed are even more vulnerable to our attack. Therefore, our attack can be a good reference security point for future LPN-based wPRF proposals.

4.1.3 A Key-Recovery Attack on the LCMQ Authentication Protocol

Since the introduction of HB-like authentication protocols and subsequent cryptanalysis, the prospect of a practical, lightweight (yet MitM-secure) LPN-based authentication has always been an open question. Despite multiple breakthroughs in the fields, such as [Dod+12; Kil+11] having provided definitive answers, they often ask for (prohibitively) large key sizes, which makes it impractical for, e.g., an RFID tag. Lyubashevsky et al. [LM13a] showed a generic construction (without MACs) that is MitM-secured for any wPRFs as long as they provided reasonable properties. The LPN problem can be seen as an instantiation, and using special variants of LPN (Toepliz-LPN or Ring-LPN), the key size is reduced from quadratic to polynomial, thus suitable for memory-constrained devices.

Besides this direction, in 2013, Li et al. [LGQ13b] proposed an alternative. They devised an efficient encryption scheme based on *circulant matrix*, masking LPN samples and effectively turning the problem into a (noisy) quadratic multivariate problem, which is also $\mathcal{NP}$-hard. As the LPN samples are now further obfuscated, one can attain remarkable benchmarks such as key size and communication cost. In Paper III, we proposed an attack strategy as follows.

- Fixing the challenge in each invocation of the authentication, we formulate a noisy system of equations where the unknowns are related to the keys.
- One can solve such systems readily with ISD algorithms and recover the key.

Our attack shows an overestimation in the parameters for the desired levels of security, and suitable countermeasures should be considered.

4.1.4 A New Sieving-style Information-set Decoding Algorithm

The essence of the decoding problem is finding low-weight codewords (in the Hamming metric) in a given code. One can draw parallels with the task in the

Closet Vector Problem, which asks for a short vector (in the Euclidean sense) in a lattice. Given the close affiliation, the natural question arises of whether a lattice-solving algorithm exists for the Syndrome Decoding Problem.

Paper IV gives the first and definitive answer to this question. We observe that given a subset of low-weight vectors, we can "guide" this subset closer to the desired syndrome and eventually find the solution to the Syndrome Decoding Problem. We achieve this by combining the vectors and maintaining the weight while asking the vectors in the subset to satisfy gradually more syndrome bits. We devised an algorithm to detect weight-preserving combinations from the input vectors, which can be considered as an instance of the *fixed-weight Nearest-Neighbor Search* [Duc+24]. In particular, we investigate many interesting aspects of our algorithms, providing heuristic arguments and simulations to confirm our analyses.

As for performances, we apply our algorithms to solve the SDP parameter sets that appear in notable KEMs such as McEliece, BIKE. We observe a new time-memory trade-off behavior for McEliece parameters, and the algorithm offers improvements in the memory-restricted scenarios. For BIKE, the sieving algorithm again shows efficient memory usage, achieving comparable complexity with much less memory than previous work. As a concluding remark, our approach can be a valuable alternative in certain cryptanalytic scenarios where memory is a factor.

4.1.5 Efficient Authentication Protocols from the Restricted Syndrome Decoding Problem

As an attempt to diversify our Post-quantum Cryptography portfolios, NIST has recently called for the *Additional Post-Quantum Signature Schemes* [AG11], which features many code-based signature constructions. Round 2 of the competition has seen the survival of two code-based proposals: CROSS [Bal+24c] and LESS [Bal+24b]. In particular, we are interested in CROSS that features a new $\mathcal{NP}$-complete variant of SDP called RSDP, where the error is drawn from a subset of $\mathbb{F}_q$, rather than sampled uniformly at random.

In Paper V, we introduce the *Oracle model* of the Restricted Decoding problem, analogously to the LWE/LPN Oracle. Moreover, our goal is to devise an efficient authentication that can rival the efficiency of the LPN problem, as shown in the HB family of authentication protocols. We show that RSDP can offer the same elegance when choosing a suitable restricted set to make operations as simple as bit-flipping and cyclic shifting. In contrast to the previous problems, RSDP in this form can be solved in polynomial time, using Aurora-Ge algebraic attack [AG11], once the number of Oracle calls is arbitrarily large. However, under reasonable assumptions, we show that the (Oracle) RSDP offers remarkably appealing characteristics in the building of lightweight authentication protocols, such as key size, communication, and computational cost.

The paper presents two secure designs in the Active and Man-in-the-Middle attack models. The designs take inspiration from various well-known LPN-based

lightweight authentication protocols, such as [JW05b; LM13b]. We observe that RSDP can remarkably adapt to previous reduction techniques. Parameters for relevant security parameters are proposed, where we compare the performance of our proposal with other established constructions, showing promising benchmarks.

4.1.6 A BKW-Style Solver for the Restricted Syndrome Decoding Problem

The (classical) SDP has been studied extensively, highlighted by the fact that a diversified portfolio of cryptanalysis exists for the problem. For example, the most well-known is the ISD algorithms pioneered by Prange [Pra62]. Recently, *Statistical Decoding Algorithm* has been revisited [Car+22; Car+24], presenting promising results in certain code-rate regimes. In addition, we reserve BKW algorithm [BKW03a] when the code rate is arbitrarily small (e.g., LPN).

NP-hard variants, particularly structured-noise alternatives, such as *Regular Syndrome Decoding Problem* and RSDP, have emerged in favor of their classical counterpart in cryptographic constructions due to their performance advantages. However, novel problems require sufficient cryptanalysis to strengthen our confidence in employing them. Indeed, several adaptations of ISD algorithms have been proposed [Bal+24c; Bit+23], and algebraic approaches [BØ23; BM24] to exploit the additional structure have been considered recently. The consensus is that, in practice, ISD-derived algorithms are the main cryptanalysis tools for RSDP-based constructions. However, the applications of such solvers are restricted to either CROSS parameters [Bal+24c] or to peculiar cases of restricted sets [Bit+23].

In Paper VI, we adapt the BKW algorithm to solve RSDP, suitable for RSDP-based applications where an adversary encounters an RSDP instance with a very small code rate, e.g., as in Paper V. We reintroduce the concept of an RSDP Oracle, and the inspiration for our work is as follows. An RSDP 'sample' resembles the well-known *Learning with Errors* problem, and

- Via a simple transformation, we can make the secret follow the same distribution with the errors.
- Since the restricted set has a relatively small entropy, we suspect that the output from an RSDP Oracle has a detectable bias.
- Therefore, by making the guess space smaller, at the expense of introducing more noise, we can eventually recover the correct partial secret and solve the RSDP problem.

Our approach makes use of state-of-the-art BKW techniques such as *Covering codes* and *Subspace Hypothesis Testing*, relying on reasonable heuristics that can be verified with simulations. We observe significant improvements over traditional solvers with additional samples, particularly before algebraic attacks can be considered viable. We broaden the scope of our algorithm by considering RSDP

instances with larger restricted error sets. Contrary to ISD and algebraic solvers, our algorithm is remarkably adaptive, and our preliminary analysis shows that the BKW-solver achieves even more impressive results. To strengthen our results, we conducted Known-Answer tests on small parameters to rectify our arguments made throughout the paper.

In addition to enriching the cryptanalytic arsenal for RSDP, we improve the understanding of RSDP security in different scenarios. Our future works revolve around perfecting and fine-tuning various steps towards the goal of reducing the required number of samples needed for our algorithms. In particular, using better-suited covering codes and solving the dependencies problem when samples are not abundant.

4.2 Conclusions

The topic of this dissertation has been to present new results on code-based cryptography.

The contributions covered several aspects, including cryptanalysis and the construction of new cryptographic schemes.

Paper I, Paper II, and Paper III examine, under various angles, the security of LPN-based cryptographic objects, ranging from stream ciphers to weak pseudorandom functions and authentication protocols. These constructions lean on the appealing simplicity of LPN while achieving provable security. In particular, the Firekite cipher smartly minimizes the cost of generating randomness by reserving a portion of their keystream for the next round while LMCQ masks their LPN-samples with an efficient encryption scheme, allowing for aggressive choice of parameters. They are all great advocates for LPN, especially in the context of lightweight, computationally friendly cryptography. However, pioneering constructions are only as secure as the corresponding cryptanalysis efforts, which are, oftentimes, lacking due to their novelty. Therefore, our work improves the understanding and confidence in the security of employing learning-based assumptions and serves as a cautionary note for potential users.

On the other hand, Paper IV is our attempt to venture outside of the well-known *representation*-based paradigm of the Information-set Decoding algorithm. Drawing inspiration from sieving algorithms, we devise a Sieving-style solver for the SDP problem that proves to be useful in specific scenarios. We observe that this can lead to an interesting direction of research where many further optimizations can be made.

The last two papers deal with the novel Restricted Decoding Problem, specifically in its Oracle model. We believe that such a setting will open doors to many potential applications, similar to the case of the Learning with Errors problem. First, Paper V explores the idea of building an extremely efficient authentication protocol, diversifying the use of RSDP beyond previous work. Moreover, we also present the new, much-needed solving algorithms for RSDP that can utilize ad-

ditional samples and bridge the gap between combinatoric and algebraic solvers. With such a tool, cryptographers may have greater freedom and confidence in proposing other RSDP-based schemes.

4.2.1 Reflections

With the help of senior co-authors, particularly my supervisors, I have learned advanced and up-to-date specialized knowledge in Cryptography, Cryptanalysis, and Code-based Cryptography, as shown in our publications.

In the first stage of the Ph.D. program, I had the opportunity to focus on attacking code-based cryptosystems. This gave me hands-on experience in cryptanalysis. In addition to coding problems and state-of-the-art solving approaches, I broadened my knowledge of code-based applications and how to employ many creative techniques to exploit weaknesses in a particular design. As a result, I frequently investigated well-known cryptanalytic tools in code-based cryptography. A significant achievement was understanding how to navigate research. For example, in Paper I, Paper II, Paper III, and Paper IV, I learned about the analysis aspects of our attacks and algorithms that went beyond presenting our core ideas and pseudocodes. Notably, it was to answer a series of follow-up questions about the correctness, complexity, or supporting heuristic arguments. I exercised well-known methodologies, such as combinatorics and probability theory, to analyze our approaches. This practice allowed me to deliver quality results suitable for the research field, as well as fair comparisons with existing work.

In addition, I have been responsible for most of the manuscript writing. It was crucial for my formative research stage, which required me to consume a broad knowledge of previous research. Besides knowledge of code-based cryptography, I also got the gist of how to structure and present research papers in a conventional way in the community. In discussion, I gradually became bolder and more upfront when proposing my ideas and opinions on our work. The same can be said about time management. I showed more dependency in scheduling timelines and expectations for different stages of research with my co-authors.

One of the most important learning outcomes in the Ph.D. program is to show the capability to "demonstrate the ability to identify and formulate issues with scholarly precision critically, autonomously and creatively...". In other words, it means identifying the needs for certain research directions while standing more on one's legs, becoming more independent. During the projects of Paper V and Paper VI, it has been my compelling motivation. In Paper V, we explore the idea of having an analogy of the LPN-SDP relationship by formulating an Oracle model of RSDP, eventually showing its (remarkable) competitiveness in lightweight cryptography. Likewise, in Paper VI, we address the issue of cryptanalysis for the novel RSDP problem, especially in the new Oracle setting. With the suggestion of having more responsibility from my supervisors, I initiated the first research steps, including literature review, initial sketches/designs, and facilitating the basic

strategies for proofs/analysis. Having familiarized myself with the research area's methodology, I could lay the groundwork for our projects, getting it to a state where my co-authors can join in to optimize and propose how to achieve the results. Looking back, although the results left much to be desired, and progress was, at times, slow and stagnant, I believe it was a valuable experience as a junior researcher to push the work forward. Still, there could not have been papers without the feedback, experience, and valuable insights from my co-authors, which were essential for me to carry on with more confidence.

I have learned how to program and implement our algorithms in all our projects in C++ and Python. In most research disciplines, experiments for verification or refutation are as crucial as forming hypotheses. Not only are they necessary for ethical aspects and the reviewing processes (ease of reproducing results), but they are also excellent tools to guide our research correctly. There have been numerous occasions when we have discovered unexpected behaviors from our algorithms. Trying to understand and develop heuristic arguments that can explain the non-desirable aspects of our work, in addition to being the right thing to do (as opposed to burying it under the carpet), taught me much more than just cryptography. For example, in Paper IV, we encountered the problem of duplicates in our sieving-style ISD algorithm, which took us quite some time to figure out. However, we then had a good understanding of how the algorithm performed and, by extension, how it differed from previous approaches.

Peer-to-peer review is also an invaluable process throughout my program, especially with how the communication is conducted within the research field. Reviewers, having faced similar challenges, can provide me with different perspectives and insights to improve our work. Most importantly, they point out weaknesses and oversights that we did not take into account. They can validate or challenge our assumptions, foster a more comprehensive understanding, and ensure our work aligns with the research standards. The process not only enhances the quality of our research but also strengthens my critical thinking and communication skills. More often than not, we come out of the reviewing process with a far more comprehensive grasp of our work than before.

Although not evident in this dissertation, performing department duties as a teaching assistant has improved me as a Ph.D. student in unexpected ways. Teaching presented the challenge of explaining ideas and concepts concisely and effectively, translating tremendously into expounding my research ideas and motivations, especially during conferences and invited talks. In addition, it is also an opportunity to broaden my general knowledge in electrical engineering and cryptography, which have aided me in critical (and dry) times in my research. Through students' curious interrogations of the purpose of being a Ph.D. and what exactly is (post-quantum) cryptography, I have the chance to summarize and look back on my Ph.D. journey, speak a few (possibly wrong) things in the naive hope of motivation, about other fields of post-quantum cryptography or exciting projects that our research group is doing. Each time, it certainly solidified further my

understanding of the research fields.

References

[AB09] S. Arora and B. Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[AG11] S. Arora and R. Ge. "New algorithms for learning in presence of errors". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2011, pp. 403–415.

[Ara+23] N. Aragon et al. "BIKE: Bit-flipping key encapsulation". In: *NIST PQC* (2023).

[Bal+19] M. Baldi et al. "A Finite Regime Analysis of Information Set Decoding Algorithms". In: *Algorithms* 12.10 (2019), p. 209.

[Bal+20a] M. Baldi et al. "A New Path to Code-based Signatures via Identification Schemes with Restricted Errors". In: *CoRR* abs/2008.06403 (2020). arXiv: 2008.06403.

[Bal+24a] M. Baldi et al. "CROSS". In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2024).

[Bal+24b] M. Baldi et al. "LESS: Linear Equivalence Signature Scheme". In: *Round 2 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2024).

[Bal+24c] M. Baldi et al. "Zero Knowledge Protocols and Signatures from the Restricted Syndrome Decoding Problem". In: *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part II*. Ed. by Q. Tang and V. Teague. Vol. 14602. Lecture Notes in Computer Science. Springer, 2024, pp. 243–274.

[Ban+23] G. Banegas et al. “Wave”. In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2023).

[BC08] J. Bringer and H. Chabanne. “Trusted-HB: a low-cost version of HB+ secure against Man-in-The-Middle attacks”. In: *CoRR* abs/0802.0603 (2008). arXiv: 0802.0603.

[Bec+12a] A. Becker et al. “Decoding Random Binary Linear Codes in 2 n/20: How 1 + 1 = 0 Improves Information Set Decoding”. In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 520–536.

[Ber+20] D. J. Bernstein et al. “Classic McEliece: conservative code-based cryptography”. In: *NIST submissions* (2020).

[Ber09] D. J. Bernstein. “Introduction to post-quantum cryptography”. In: *Post-quantum cryptography*. Springer, 2009, pp. 1–14.

[Bet+24] S. Bettaieb et al. “PERK: compact signature scheme based on a new variant of the permuted kernel problem”. In: *Des. Codes Cryptogr.* 92.8 (2024), pp. 2131–2157.

[Bit+23] S. Bitzer et al. “Generic Decoding of Restricted Errors”. In: *IEEE International Symposium on Information Theory, ISIT 2023, Taipei, Taiwan, June 25-30, 2023*. IEEE, 2023, pp. 246–251.

[BKW03a] A. Blum, A. Kalai, and H. Wasserman. “Noise-tolerant learning, the parity problem, and the statistical query model”. In: *J. ACM* 50.4 (2003), pp. 506–519.

[BLP08] D. J. Bernstein, T. Lange, and C. Peters. “Attacking and Defending the McEliece Cryptosystem”. In: *Post-Quantum Cryptography, Second International Workshop, PQCrypto 2008, Cincinnati, OH, USA, October 17-19, 2008, Proceedings*. Ed. by J. Buchmann and J. Ding. Vol. 5299. Lecture Notes in Computer Science. Springer, 2008, pp. 31–46.

[BLP11] D. J. Bernstein, T. Lange, and C. Peters. “Smaller Decoding Exponents: Ball-Collision Decoding”. In: *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*. Ed. by P. Rogaway. Vol. 6841. Lecture Notes in Computer Science. Springer, 2011, pp. 743–760.

[BM17b] L. Both and A. May. “The Approximate k-List Problem”. In: *IACR Trans. Symmetric Cryptol.* 2017.1 (2017), pp. 380–397.

[BM18] L. Both and A. May. "Decoding Linear Codes with High Error Rate and Its Impact for LPN Security". In: *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*. Ed. by T. Lange and R. Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 25–46.

[BM24] W. Beullens and P. B. andIn contrast Morten Øygarden. "A Security Analysis of Restricted Syndrome Decoding Problems". In: *IACR Cryptol. ePrint Arch.* (2024), p. 611.

[BMT78b] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)" In: *IEEE Trans. Inf. Theory* 24.3 (1978), pp. 384–386.

[BØ23] P. Briaud and M. Øygarden. "A New Algebraic Approach to the Regular Syndrome Decoding Problem and Implications for PCG Constructions". In: *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V*. Ed. by C. Hazay and M. Stam. Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 391–422.

[Bog+21b] S. Bogos et al. "Towards Efficient LPN-Based Symmetric Encryption". In: *Applied Cryptography and Network Security - 19th International Conference, ACNS 2021, Kamakura, Japan, June 21-24, 2021, Proceedings, Part II*. Ed. by K. Sako and N. O. Tippenhauer. Vol. 12727. Lecture Notes in Computer Science. Springer, 2021, pp. 208–230.

[Bon+18] D. Boneh et al. "Exploring Crypto Dark Matter: - New Simple PRF Candidates and Their Applications". In: *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*. Ed. by A. Beimel and S. Dziembowski. Vol. 11240. Lecture Notes in Computer Science. Springer, 2018, pp. 699–729.

[BR17a] A. Bogdanov and A. Rosen. "Pseudorandom Functions: Three Decades Later". In: *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*. Ed. by Y. Lindell. Cham: Springer International Publishing, 2017, pp. 79–158.

[BV15] S. Bogos and S. Vaudenay. "How to Sequentialize Independent Parallel Attacks? - Biased Distributions Have a Phase Transition". In: *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 -*

December 3, 2015, Proceedings, Part II. Ed. by T. Iwata and J. H. Cheon. Vol. 9453. Lecture Notes in Computer Science. Springer, 2015, pp. 704–731.

[BV16] S. Bogos and S. Vaudenay. "Optimization of LPN Solving Algorithms". In: *Advances in Cryptology – ASIACRYPT 2016*. Ed. by J. H. Cheon and T. Takagi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 703–728.

[Car+22] K. Carrier et al. "Statistical Decoding 2.0: Reducing Decoding to LPN". In: *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV*. Ed. by S. Agrawal and D. Lin. Vol. 13794. Lecture Notes in Computer Science. Springer, 2022, pp. 477–507.

[Car+24] K. Carrier et al. "Reduction from Sparse LPN to LPN, Dual Attack 3.0". In: *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*. Ed. by M. Joye and G. Leander. Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 286–315.

[CC98] A. Canteaut and F. Chabaud. "A New Algorithm for Finding Minimum-Weight Words in a Linear Code: Application to McEliece's Cryptosystem and to Narrow-Sense BCH Codes of Length 511". In: *IEEE Trans. Inf. Theory* 44.1 (1998), pp. 367–378.

[Che+22] J. H. Cheon et al. "Adventures in crypto dark matter: attacks, fixes and analysis for weak pseudorandom functions". In: *Des. Codes Cryptogr.* 90.8 (2022), pp. 1735–1760.

[DH76] W. Diffie and M. Hellman. "New directions in cryptography". In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[Din+21] I. Dinur et al. "MPC-Friendly Symmetric Cryptography from Alternating Moduli: Candidates, Protocols, and Applications". In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*. Ed. by T. Malkin and C. Peikert. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 517–547.

[DK07] D. N. Duc and K. Kim. *Securing* HB+ *against GRS man-in-the-middle attack*. 2007.

[Dod+12] Y. Dodis et al. “Message Authentication, Revisited”. In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 355–374.

[DT17] T. Debris-Alazard and J.-P. Tillich. *Statistical Decoding*. 2017. arXiv: 1701.07416 [cs.CR].

[Duc+24] L. Ducas et al. “Asymptotics and Improvements of Sieving for Codes”. In: *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*. Ed. by M. Joye and G. Leander. Vol. 14656. Lecture Notes in Computer Science. Springer, 2024, pp. 151–180.

[Dum91] I. Dumer. “On minimum distance decoding of linear codes”. In: *Proc. 5th Joint Soviet-Swedish International Workshop Information Theory*. 1991, pp. 50–52.

[DV13a] A. Duc and S. Vaudenay. “HELEN: A Public-Key Cryptosystem Based on the LPN and the Decisional Minimal Distance Problems”. In: *Progress in Cryptology – AFRICACRYPT 2013*. Ed. by A. Youssef, A. Nitaj, and A. E. Hassanien. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 107–126.

[EB22c] A. Esser and E. Bellini. “Syndrome decoding estimator”. In: *IACR International Conference on Public-Key Cryptography*. Springer. 2022, pp. 112–141.

[EKZ21] A. Esser, R. Kübler, and F. Zweydinger. “A Faster Algorithm for Finding Closest Pairs in Hamming Metric”. In: *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2021, December 15-17, 2021, Virtual Conference*. Ed. by M. Bojanczyk and C. Chekuri. Vol. 213. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 20:1–20:21.

[Eli55] P. Elias. “Coding for noisy channels”. In: *IRE*. 1955.

[Ess23] A. Esser. “Revisiting Nearest-Neighbor-Based Information Set Decoding”. In: *Cryptography and Coding - 19th IMA International Conference, IMACC 2023, London, UK, December 12-14, 2023, Proceedings*. Ed. by E. A. Quaglia. Vol. 14421. Lecture Notes in Computer Science. Springer, 2023, pp. 34–54.

[Fey82] R. P. Feynman. “Simulating physics with computers”. In: *International journal of theoretical physics* 21.6/7 (1982), pp. 467–488.

[FS09] M. Finiasz and N. Sendrier. “Security Bounds for the Design of Code-Based Cryptosystems”. In: *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*. Ed. by M. Matsui. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 88–105.

[Gal62] R. Gallager. “Low-density parity-check codes”. In: *IRE Transactions on Information Theory* 8.1 (1962), pp. 21–28.

[GJL14] Q. Guo, T. Johansson, and C. Löndahl. “Solving LPN Using Covering Codes”. In: *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*. Ed. by P. Sarkar and T. Iwata. Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 1–20.

[Gop81] V. D. Goppa. “Codes on Algebraic Curves”. In: *Izvestiya Rossiiskoi Akademii Nauk. Seriya*. 1981.

[Gro96] L. K. Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC ’96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219.

[GRS05] H. Gilbert, M. Robshaw, and H. Sibert. *An Active Attack Against* HB+ *- A Provably Secure Lightweight Authentication Protocol*. `https://eprint.iacr.org/2005/237`. 2005.

[GRS08a] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. “Good Variants of HB + Are Hard to Find”. In: *Financial Cryptography and Data Security*. Ed. by G. Tsudik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 156–170.

[GRS08b] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. “HB$^{\#}$: Increasing the Security and Efficiency of HB^{+}”. In: *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*. Ed. by N. P. Smart. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[GRS08c] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "HB#: Increasing the Security and Efficiency of HB+". In: *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[GRS08d] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "How to Encrypt with the LPN Problem". In: *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*. Ed. by L. Aceto et al. Vol. 5126. Lecture Notes in Computer Science. Springer, 2008, pp. 679–690.

[HB01b] N. J. Hopper and M. Blum. "Secure Human Identification Protocols". In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. Ed. by C. Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 52–66.

[HJ10] N. Howgrave-Graham and A. Joux. "New Generic Algorithms for Hard Knapsacks". In: *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*. Ed. by H. Gilbert. Vol. 6110. Lecture Notes in Computer Science. Springer, 2010, pp. 235–256.

[IM98] P. Indyk and R. Motwani. "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality". In: *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*. Ed. by J. S. Vitter. ACM, 1998, pp. 604–613.

[Jab01] A. A. Jabri. "A Statistical Decoding Algorithm for General Linear Block Codes". In: *Cryptography and Coding*. Ed. by B. Honary. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 1–8.

[JW05b] A. Juels and S. A. Weis. "Authenticating Pervasive Devices with Human Protocols". In: *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*. Ed. by V. Shoup. Vol. 3621. Lecture Notes in Computer Science. Springer, 2005, pp. 293–308.

[Kil+11] E. Kiltz et al. “Efficient Authentication from Hard Learning Problems”. In: *Advances in Cryptology – EUROCRYPT 2011*. Ed. by K. G. Paterson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 7–26.

[KS06a] J. Katz and J. S. Shin. “Parallel and Concurrent Security of the HB and HB$^+$ Protocols”. In: *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*. Ed. by S. Vaudenay. Vol. 4004. Lecture Notes in Computer Science. Springer, 2006, pp. 73–87.

[Kuz+23] O. Kuznetsov et al. “Trade-offs in Post-Quantum Cryptography: A Comparative Assessment of BIKE, HQC, and Classic McEliece”. In: *Proceedings of the Workshop on Classic, Quantum, and Post-Quantum Cryptography (CQPC 2023) co-located with International Conference on Problems of Infocommunications. Science and Technology (PICST 2023), Kyiv, Ukraine, August 1, 2023 (online)*. Ed. by V. Sokolov, V. Ustimenko, and M. Nazarkevych. Vol. 3504. CEUR Workshop Proceedings. CEUR-WS.org, 2023, pp. 1–11.

[LB88] P. J. Lee and E. F. Brickell. “An Observation on the Security of McEliece’s Public-Key Cryptosystem”. In: *Advances in Cryptology - EUROCRYPT ’88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*. Ed. by C. G. Günther. Vol. 330. Lecture Notes in Computer Science. Springer, 1988, pp. 275–280.

[Leo88] J. S. Leon. “A probabilistic algorithm for computing minimum weights of large error-correcting codes”. In: *IEEE Trans. Inf. Theory* 34.5 (1988), pp. 1354–1359.

[LF06b] É. Levieil and P.-A. Fouque. “An improved LPN algorithm”. In: *Security and Cryptography for Networks: 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006. Proceedings 5*. Springer. 2006, pp. 348–359.

[LGQ13b] Z. Li, G. Gong, and Z. Qin. “Secure and efficient LCMQ entity authentication protocol”. In: *IEEE transactions on information theory* 59.6 (2013), pp. 4042–4054.

[LM13a] V. Lyubashevsky and D. Masny. “Man-in-the-Middle Secure Authentication Schemes from LPN and Weak PRFs”. In: *Advances in Cryptology – CRYPTO 2013*. Ed. by R. Canetti and J. A. Garay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 308–325.

[LM13b] V. Lyubashevsky and D. Masny. “Man-in-the-Middle Secure Authentication Schemes from LPN and Weak PRFs”. In: *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*. Ed. by R. Canetti and J. A. Garay. Vol. 8043. Lecture Notes in Computer Science. Springer, 2013, pp. 308–325.

[LMM08] X. Leng, K. Mayes, and K. Markantonakis. “HB-MP+ Protocol: An Improvement on the HB-MP Protocol”. In: *2008 IEEE International Conference on RFID*. 2008, pp. 118–124.

[Lön14] C. Löndahl. “Some Notes on Code-Based Cryptography”. PhD thesis. Dec. 2014.

[Lyu05] V. Lyubashevsky. “The Parity Problem in the Presence of Noise, Decoding Random Linear Codes, and the Subset Sum Problem”. In: *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*. Ed. by C. Chekuri et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 378–389.

[Man80] Y. Manin. “Computable and Non-computable”. In: *Sov. Radio*. 1980, pp. 13–15.

[Mat94] M. Matsui. “Linear Cryptanalysis Method for DES Cipher”. In: *Advances in Cryptology — EUROCRYPT ’93*. Ed. by T. Helleseth. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 386–397.

[McE78] R. J. McEliece. “A public-key cryptosystem based on algebraic coding theory”. In: *JPL DSN Progress Reports* 44 (1978), pp. 114–116.

[Mel+23a] C. A. Melchor et al. “Hamming quasi-cyclic (HQC)”. In: *NIST PQC* (2023).

[Meu13] A. Meurer. “A coding-theoretic approach to cryptanalysis”. doctoralthesis. Ruhr-Universität Bochum, Universitätsbibliothek, 2013.

[MMT11a] A. May, A. Meurer, and E. Thomae. “Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$”. In: *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*. Ed. by D. H. Lee and X. Wang. Vol. 7073. Lecture Notes in Computer Science. Springer, 2011, pp. 107–124.

[MO15] A. May and I. Ozerov. "On Computing Nearest Neighbors with Applications to Decoding of Binary Linear Codes". In: *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*. Ed. by E. Oswald and M. Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 203–228.

[MP07] J. Munilla and A. Peinado. "HB-MP: A further step in the HB-family of lightweight authentication protocols". In: *Computer Networks* 51.9 (2007). (1) Advances in Smart Cards and (2) Topics in Wireless Broadband Systems, pp. 2262–2267.

[Nie86] H. Niederreiter. "Knapsack-Type Cryptosystems and Algebraic Coding Theory". In: *Problems of Control and Information Theory* 15 (1986), pp. 159–166.

[OOV08] K. Ouafi, R. Overbeck, and S. Vaudenay. "On the Security of HB# against a Man-in-the-Middle Attack". In: *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*. Vol. 5350. Lecture Notes in Computer Science. Springer, 2008, pp. 108–124.

[Pet10] C. Peters. "Information-Set Decoding for Linear Codes over F_q". In: *Post-Quantum Cryptography, Third International Workshop, PQCrypto 2010, Darmstadt, Germany, May 25-28, 2010. Proceedings*. Ed. by N. Sendrier. Vol. 6061. Lecture Notes in Computer Science. Springer, 2010, pp. 81–94.

[Pra62] E. Prange. "The use of information sets in decoding cyclic codes". In: *IRE Trans. Inf. Theory* 8.5 (1962), pp. 5–9.

[RS60] I. S. Reed and G. Solomon. "Polynomial Codes Over Certain Finite Fields". In: *Journal of the Society for Industrial and Applied Mathematics* 8.2 (1960), pp. 300–304. eprint: `https://doi.org/10.1137/0108018`.

[RSA78] R. L. Rivest, A. Shamir, and L. M. Adleman. "A method for obtaining digital signatures and public-key cryptosystems". In: *Commun. ACM* 21 (1978), pp. 120–126.

[Sen11] N. Sendrier. "Decoding One Out of Many". In: *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*. Ed. by B. Yang. Vol. 7071. Lecture Notes in Computer Science. Springer, 2011, pp. 51–67.

[Sha79] A. Shamir. “How to share a secret”. In: *Commun. ACM* 22.11 (Nov. 1979), pp. 612–613.

[Sha89] A. Shamir. “An Efficient Identification Scheme Based on Permuted Kernels (Extended Abstract)”. In: *Advances in Cryptology - CRYPTO ’89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*. Ed. by G. Brassard. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 606–609.

[Sho94] P. W. Shor. “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”. In: *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*. IEEE Computer Society, 1994, pp. 124–134.

[Ste88] J. Stern. “A method for finding codewords of small weight”. In: *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*. Ed. by G. D. Cohen and J. Wolfmann. Vol. 388. Lecture Notes in Computer Science. Springer, 1988, pp. 106–113.

[Wag02] D. A. Wagner. “A Generalized Birthday Problem”. In: *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*. Ed. by M. Yung. Vol. 2442. Lecture Notes in Computer Science. Springer, 2002, pp. 288–303.

[Yu+18] Z. Yu et al. “A Practical Public Key Encryption Scheme Based on Learning Parity With Noise”. In: *IEEE Access* PP (May 2018).

Included Publications

Attacks on the Firekite Cipher

Abstract

Firekite is a synchronous stream cipher using a pseudo-random number generator (PRNG) whose security is conjectured to rely on the hardness of the *Learning Parity with Noise* (LPN) problem. It is one of a few LPN-based symmetric encryption schemes, and it can be very efficiently implemented on a low-end SoC FPGA. The designers, Bogos, Korolija, Locher and Vaudenay, demonstrated appealing properties of Firekite, such as requiring only one source of cryptographically strong bits, small key size, high attainable throughput, and an estimate for the bit level security depending on the selected practical parameters.

We propose distinguishing and key-recovery attacks on Firekite by exploiting the structural properties of its PRNG. We adopt several *birthday-paradox* techniques to show that a particular sum of Firekite's output has a low Hamming weight with higher probability than the random case. We achieve the best distinguishing attacks with complexities $2^{66.75}$ and $2^{106.75}$ for Firekite's parameters corresponding to 80-bit and 128-bit security, respectively. By applying the distinguishing attacks and an additional algorithm we describe, one can also recover the secret matrix used in the Firekite PRNG, which is built from the secret key bits. This key recovery attack works on most large instances of Firekite parameters and has slightly larger complexity, for instance, $2^{69.87}$ on the 80-bit security parameters $n = 16,384, m = 216, k = 216$.

Thomas Johansson, Willi Meier, Vu Nguyen. "Attacks on the Firekite Cipher". In *Fast Software Encryption (FSE) 2023, Beijing, China*. IACR Transactions on Symmetric Cryptology. 2022, 3, p. 191–216.

1 Introduction

Since Shor [Sho99] in his seminal work introduced quantum algorithms that efficiently break the *discrete-log* and *factoring* problems, researchers have set their sights to cryptographic alternatives that promise to be quantum-resistant such as *lattice-based* or *code-based* cryptography. In particular, cryptographic primitives whose security relies on *learning problems*, such as *Learning Parity with Noise* (LPN), *Learning with Errors* (LWE), and the closely related Ring-LPN, are receiving great attentions as they are built on supposedly hard problems.[1] Moreover, Impagliazzo and Levin showed that cryptography is only possible if efficient learning is not [IL90]. Besides the absence of an efficient LPN-solving quantum algorithm, LPN-based constructions are desired as they can be efficiently implemented using mainly XOR ('exclusive or') operations, thus achieving popularity in lightweight cryptography on constrained, low-powered devices. However, most LPN constructions are inclined towards asymmetric cryptography and they have their own disadvantages. These include the requirement to produce and extract randomness (cryptographically secure bits) from an entropy-limited source, causing a significant overhead cost [Hen+12; Sho99], and that they also often require large public keys.

Bogos, Korolija, Locher and Vaudenay [Bog+21b] proposed *Firekite*, a synchronous symmetric cipher, using an LPN-based PRNG which requires only one cryptographically strong bit vector to construct the secret matrix key. A small key size is attained by moving from an LPN problem to a Ring-LPN problem [Hey+07]. Their study conjectures that the corresponding Ring-LPN instance remains hard to solve when using said matrix instead of a fully random matrix. They demonstrated that using the Firekite noise distribution for an LPN instance is still secure and there is a 'partial' transformation to an LPN instance. Using the best BKW-style algorithm proposed by Levieil and Fouque [LF06a], Firekite's designers estimated the complexity to break the transformed LPN instances, thus derived concrete complexity results for attacking their cipher. The cipher's efficiency was tested in terms of the throughput, which is the number of bytes encrypted or decrypted per second using both desktop computers and FPGAs. They also showcased that, given dedicated hardware, the Firekite PRNG can be parallelized, hence throughput improved substantially for larger parameters.

One can draw many parallels between Firekite and the closely related LPN-C [GRS08d]; in particular, both involve computing a noisy product using a secret random matrix $\mathbf{M}$ and a random error vector $\mathbf{e}$. However, LPN-C further requires an error correcting code $\mathcal{C}$ and the error vectors are drawn from a Bernoulli distribution, as opposed to being bounded as in the Firekite PRNG. This could make the decrypting process fail once the error weight exceeds the code's error correcting capacity. This drawback could be amended by truncating the binomial distribution to make sure not too many bits are set in the error vectors. How-

[1] LPN with adversarial errors is $\mathcal{NP}$-hard.

ever, it is speculated that doing so may have a negative impact to the security of LPN-C [Bog+21b]. Furthermore, LPN-C inherently requires a large random secret matrix and samples two uniformly random vectors for every invocation of the encryption algorithm. Hence, it becomes infeasible to implement it efficiently when implemented in a constrained environment. Firekite, besides avoiding such undesirable features, surpasses LPN-C by not requiring fresh random bits for each output block.

Even more important than constructing schemes that are potentially quantum secure, it is crucial to try to attack them with the most suitable approaches to better understand their security.

1.1 Contributions

In this work, we propose both distinguishing and key-recovery attacks for Firekite. We observe that the secret matrix is fixed throughout every round of encryption. Hence, if the vector components in the internal states collide to the zero codeword, the outputs of Firekite, when combined together appropriately, result in unusually low weight sums and can be detected. In other words, finding such occurrences amounts to solving a birthday paradox problem with a specific target weight.

We then consider the secret matrix as the generator for a code and by carefully determining which positions in the above combinations are free of errors, we describe a key-recovery attack with a slightly higher complexity than that of the distinguishing attack.

As an example, we apply the distinguishing attacks on the Firekite cipher with specific parameters that target 80-bit and 128-bit to understand better Firekite's security. In particular, we launch both a distinguishing attack and a key recovery attack on parameters $n = 16384, m = 216, k = 216$ with complexity $2^{68.87}$ and $2^{69.97}$, respectively. As there are many choices of parameter sets for each security level, the complexity numbers vary a bit depending on selected parameter sets.

1.2 Related work

Due to their difficulty, either proved or conjectured, LPN and Ring-LPN have made their way into many cryptographic constructions, such as human identification protocols, which were firstly introduced by Hopper and Blum [HB01b], later modified and improved to HB^+ and $HB^\#$ [JW05b; KS06a; GRS08b]. Recent LPN-based encryption schemes that can be found are Helen by Duc and Vaudenay [DV13b], or LPN-C by Gilbert et al. [GRS08d]. Using the Ring-LPN variant, Heyse et al. [Hey+12] proposed an efficient two-round identification protocol in constrained environments, called Lapin. One can also find LPN useful in other applications, e.g., message authentication codes (MACs) [Kil+17; Dod+12], pseudo-random generators [App+09b; Blu+93], or CCA-secure public-key encryption schemes [YZ16].

Since its introduction, LPN has drawn a plethora of LPN-solving studies with different approaches. It is natural to see LPN as a decoding problem; hence, a generic decoding technique applies. Attacks on LPN can be categorized as *Information-set decoding* (ISD) or BKW-type algorithms, and they prove advantageous in different scenarios, namely, low noise-rate and constant noise. Information-set decoding was first introduced by Prange [Pra62], and further improved by Leon [Leo88], Lee and Brickell [LB88] and Stern [Ste93]. Recently, several methods have been proposed to achieve better attacks, to name a few, *Ball-collision technique* by Bernstein et al. [BLP11], *representation technique* by Becker, Joux, May, Meurer [Bec+12a], or the state-of-the-art algorithm [BV15]. On the other hand, BKW began with the foundation laid by Blum, Kalai, and Wasserman [BKW03a]. Besides the improvement by Levieil and Fouque who used Walsh-Hadamard transform to recover several bits of the secret vector, using a limited number of queries, notable advancements can be found such as the use of covering codes by Guo et al. [GJL14], or on the use of the *Generalized birthday attack* (GBA) [Wag02] as in [Kir11a].

Generalized birthday attack is one of the most pertinent generic attacks in cryptology, in particular, analyzing the security of an LPN-based cryptographic scheme. There have been many notable works related to the generalized birthday problem. Our study is inspired by the seminal works of Wagner [Wag02] and May et al.'s approximate k-list algorithm [BM17b].

1.3 Organization

The paper is organized as follows. Section 2 presents preliminary and background knowledge regarding the LPN problem and its variants such as Ring-LPN. A brief review of the LPN-based Firekite PRNG, and how it gave rise to the Firekite synchronous stream cipher follows. We then describe our idea, and formally analyze our attack for Firekite in Section 3. In Section 4, we attack different parameters proposed for Firekite and verify our approach by a simulation with smaller parameters. We describe our key-recovery attack in Section 5 and discussions on how to improve Firekite finally concludes our work.

2 Background

Whereas the LPN problem usually finds its cryptographic applications in the public-key domain, we will be interested in its application in symmetric cryptography. In particular, we have seen constructions of a few synchronous stream ciphers [GRS08d; Bog+21b] based on LPN.

A synchronous stream cipher is a symmetric cipher, in which a stream of pseudorandom bits is generated independently of the plaintext and ciphertext messages, and then bitwise XOR-ed to the plaintext, to encrypt, or to the ciphertext, in order to decrypt. Cryptanalytic attacks either aim to distinguish the output

of the pseudorandom bit generator from random source, recover the state of the pseudorandom generator, or recover the key. As known plaintext for a segment of ciphertext implies knowledge of the keystream for the same segment, a known plaintext attack of a synchronous stream cipher assumes that a large part of the keystream is available to an attacker which is only limited by the maximum number of keystream bits allowed to be output for a same key. Distinguishing attacks [HJB09] on the (known) keystream are relevant to the security of stream ciphers as well: depending on the nonranomness detected, some information on the plaintext may be leaked. For some stream ciphers, a distinguishing feature can even be elaborated to a key recovery attack, as is the case for the distinguishing property we shall derive for Firekite.

2.1 The LPN problem

LPN is an important problem in cryptography. It appears as one of main problems on which we base post-quantum cryptography. Due to the existence of fast algorithms for quantum computers that can solve the factorization and the discrete logarithm problems [Sho99], the LPN problem (and the related LWE problem), including its different versions, are of great interest. No fast quantum algorithm that solves the LPN problem is known. Although current omnipresent symmetric encryption schemes such as AES will likely not be rendered obsolete in the near future, studies in post-quantum cryptography, namely the aforementioned works, are of absolute necessity. We need post-quantum cryptographic primitives to have *efficiency, confidence,* and *usability* [Ber09].

Cryptographic constructions based on LPN are also appealing, since only simple operations such as bit-wise addition (XOR) and scalar products are used. This can give rise to efficient algorithms or protocols.

The LPN problem can informally be described as the problem of solving a noisy binary system of equations. We formally define it below.

Let Ber_η be the Bernoulli distribution with parameter $\eta \in (0, \frac{1}{2})$ and a bit $e \leftarrow \mathsf{Ber}_\eta$ be such that $\mathbf{Pr}[e = 1] = \eta$, $\mathbf{Pr}[e = 0] = 1 - \eta$. Denote by $\mathbf{x} \xleftarrow{U} \{0,1\}^m$ the event that a vector $\mathbf{x}$ is drawn uniformly from $\{0,1\}^m$.

Definition 9. *(LPN oracle). Let* $\mathbf{x} \xleftarrow{U} \{0,1\}^m$ *and* $\eta \in (0, \frac{1}{2})$. *An LPN oracle* Π_{LPN} *for* $\mathbf{x}$ *and* η *returns pairs of the form*

$$\left(\mathbf{g} \xleftarrow{U} \{0,1\}^m, \langle \mathbf{x}, \mathbf{g} \rangle \oplus e\right),$$

where $e \leftarrow \mathsf{Ber}_\eta$, *and* $\langle \mathbf{x}, \mathbf{g} \rangle$ *denotes the scalar product of vectors* $\mathbf{x}$ *and* $\mathbf{g}$.

Definition 10. *(LPN problem). Given an LPN oracle* Π_{LPN} *with parameters* m *and* η. *The* (m, η)*-LPN problem is finding the secret vector* $\mathbf{x}$ *and is said to be* (T, N, δ)*-solvable if there exists an algorithm* $\mathfrak{A}$ *asking for at most* N *oracle queries, using time*

at most T and

$$\mathbf{Pr}\left[\mathfrak{A}(\Pi_{\text{LPN}}) = \mathbf{x} : \mathbf{x} \xleftarrow{U} \{0,1\}^m\right] \geq \delta.$$

The definition above is known as the search version of the LPN problem. In the decisional version of the LPN problem, the objective is to distinguish pairs from Π_{LPN} from uniformly random samples. The search and decisional versions are proved to be computationally equivalent [KS06a].

We briefly look at a subclass of LPN problems called RING-LPN which proves to be useful in general and specifically used in the Firekite PRNG. Let f be a polynomial over $\mathbb{Z}_2$ and $R = \mathbb{Z}_2[x]/(f)$ denote the quotient ring. Hence R consists of all polynomials over $\mathbb{Z}_2$ of degree less than that of f. We say $r \leftarrow \mathsf{Ber}_\eta^R$ if the coefficients of the ring element $r \in R$ are assigned independently following the distribution Ber_η. If r is drawn uniformly from R, we write $r \xleftarrow{U} R$. The RING-LPN problem can be defined similarly to the standard LPN problem.

Definition 11. *(Ring-LPN oracle). Let $s \xleftarrow{U} R$ and $\eta \in (0, \frac{1}{2})$. A Ring-LPN oracle $\Pi_{\text{RING-LPN}}$ for s and η returns pairs of the form*

$$(r \xleftarrow{U} R, r \cdot s + e),$$

where $e \leftarrow \mathsf{Ber}_\eta^R$.

Definition 12. *(Ring-LPN problem). Given a Ring-LPN oracle $\Pi_{\text{RING-LPN}}$ with parameters η and a polynomial ring R. The Ring-LPN problem is finding the secret polynomial $s \in R$ and is said to be (T, N, δ)-solvable if there exists an algorithm $\mathfrak{A}$ asking for at most N oracle queries, using time at most T and*

$$\mathbf{Pr}\left[\mathfrak{A}(\Pi_{\text{RING-LPN}}) = s\right] \geq \delta.$$

It is worth pointing out the essential difference between LPN and RING-LPN. If we query the LPN oracle N times, then we can collect an $m \times N$ matrix $\mathbf{G} = \left(\mathbf{g}_1^{\mathrm{T}} \dots \mathbf{g}_N^{\mathrm{T}}\right)$ and each column is generated independently. In the case of RING-LPN, only one polynomial r is generated uniformly random in R. If we consider a polynomial as its coefficient vector, only the first column $\mathbf{r}$ is drawn uniformly random . The other columns are obtained via shifting $\mathbf{r}$ [Hey+12]. While the LPN problem has been shown to be $\mathcal{NP}$-hard in the worst case [BMT78b], the hardness of RING-LPN is not known. However, there is a reduction from RING-LPN to LPN and the assumption is that RING-LPN is also hard.

2.2 Firekite's PRNG and Firekite construction

We recall that the decisional version of the LPN assumption can be interpreted as one can not efficiently distinguish an LPN oracle from a source providing random bit vectors of length $m + 1$. Naturally, it can be extended into stating that distinguishing a noisy product of an $m \times n$ matrix M and a secret vector $\mathbf{v}$, i.e., $\mathbf{vM} + \mathbf{e}$

from a random n-bit vector in $\mathbb{Z}_2$, where $\mathbf{e}$ is a n-bit noise vector is hard. As an example, LPN-C further used a $[k, n]$ error correcting code $\mathcal{C}$ with a generator matrix $\mathbf{G}$ to encode a plaintext $\mathbf{x}$ to a ciphertext $\mathbf{c}$ through

$$\mathbf{c} = \mathbf{xG} + \mathbf{vM} + \mathbf{e}.$$

However, this construction inherently asks the source to produce random $\mathbf{v}$ and $\mathbf{e}$ for encrypting a single plaintext. The Firekite PRNG circumvents this problem by extracting both $\mathbf{v}$ and $\mathbf{e}$ from the noisy product and feeding them iteratively into the next encryption invocations. Out of n bits, one can spare $m + k \cdot \log n$ bits to initialize the next round of Firekite.[2] Let $||$ denote the usual concatenation of vectors. We write

$$\mathbf{vM} + \mathbf{e} = (\mathbf{g}||\mathbf{v}'||\mathbf{c}_e). \tag{1}$$

Assuming $n \gg m$, one can split the noisy product into three components as in (1), then m bits are used for producing the next vector $\mathbf{v}'$. Since $\mathbf{e}$ is only required to be a sparse n-bit vector, we can have a compact representation of the next noise vector, called $\mathbf{c}_e$. Then, the remaining bits, forming $\mathbf{g}$, are the PRNG's output. We are now in the position to describe the Firekite PRNG formally.

Let m, n, and k be some integer parameters, where $n \gg m$ and n is a power of 2. A secret key $\mathbf{M}$ is a binary matrix of size $m \times n$, and $\mathbf{w}$ is a vector of length $m + k \log n < n$. Together, they form a pair $(\mathbf{M}, \mathbf{w})$, the *state* of the PRNG. We define $\mathbf{w} = \mathbf{v}||\mathbf{c}_e$, where $\mathbf{v}$ and $\mathbf{c}_e$ are of length m and $k \log n$, respectively. As stated above, $\mathbf{M}$ is fixed, and $\mathbf{w}$ is updated for every iteration. It is straightforward to assign $\mathbf{v} = \mathbf{v}'$. To get the next error vector $\mathbf{e}$, we further parse $\mathbf{c}_e = \mathbf{c}_1||\mathbf{c}_2|| \ldots ||\mathbf{c}_k$ where $\mathbf{c}_i$ is of length $\log n$. Hence, each $\mathbf{c}_i$ can be seen as the binary presentation of a non-negative integer less than n. Therefore, $\mathbf{c}_e$ encodes an n-bit error vector of weight at most k. In particular, let $\mathbf{b}_{\mathbf{c}_j}$ be the unit vector of length n, where the bit at the position represented by $\mathbf{c}_j$ is 1. Then the error vector $\mathbf{e}$ is defined as $\mathbf{e} = \bigvee_{j=1}^{k} \mathbf{b}_{\mathbf{c}_j}$. Note that this construction implies $\mathbf{e}$ is not a Bernoulli distributed error. The execution of the Firekite PRNG is described by Algorithm I.1.

At each iteration, the PRNG's input is its state $(\mathbf{M}, \mathbf{w})$, where the first m bits and the remaining $k \log n$ bits of $\mathbf{w}$ are set to be $\mathbf{v}$ and $\mathbf{c}_e$ respectively. Then the error vector $\mathbf{e}$ is derived from its concise representation $\mathbf{c}_e$ and the noisy n-bit product is computed as $\mathbf{vM} + \mathbf{e}$. This vector is again parsed into $\mathbf{g}$ and $\mathbf{w}'$ of length $d = n - m - k \log n$ and $m + k \log n$, respectively. The internal state is then updated to $(\mathbf{M}, \mathbf{w}')$ and $\mathbf{g}$ is the output of the PRNG. The number r of randomization rounds is needed to guarantee that $\mathbf{v}$ is free from significant biases when Firekite begins to output its keystream [Bog+21b].

Firekite is a synchronous stream cipher that makes use of this PRNG to produce the d-bit keystream $\mathbf{g}$ directly. Therefore, for each invocation, d-bit data of a plaintext is encrypted, and the next output of Firekite depends on the up-

[2] Throughout the paper, $\log(.)$ denotes logarithm to base 2.

Algorithm I.1: Firekite PRNG

Input: An $m \times n$ secret matrix M and a nonce $\mathbf{w}$, $r > 0$: randomization rounds.

while $r \neq 0$ **do**
  Parse: $\mathbf{w} = \mathbf{v}||\mathbf{c}_1||\mathbf{c}_2||\ldots||\mathbf{c}_k$;
  $\mathbf{e} \leftarrow \bigvee_{j=1}^{k} \mathbf{b}_{\mathbf{c}_j}$;
  $\mathbf{g}||\mathbf{w}' := \mathbf{v}\mathbf{M} + \mathbf{e}$; `// Randomization`
  $\mathbf{w} \leftarrow \mathbf{w}'$;
  $r \leftarrow r - 1$;
while *true* **do**
  Parse: $\mathbf{w} = \mathbf{v}||\mathbf{c}_1||\mathbf{c}_2||\ldots||\mathbf{c}_k$;
  $\mathbf{e} \leftarrow \bigvee_{j=1}^{k} \mathbf{b}_{\mathbf{c}_j}$;
  $\mathbf{g}||\mathbf{w}' := \mathbf{v}\mathbf{M} + \mathbf{e}$; `// Generating keystream`
  $\mathbf{w} \leftarrow \mathbf{w}'$;
  Return: g

dated internal state. The designers pointed out that, for practicality, the parameters m, n, and k need to be large which in turn makes the secret key $\mathbf{M}$ big. In order to solve this problem, they proposed the following technique, which turns the LPN instance into a Ring-LPN instance. Consider $R = \mathbb{Z}_2[X]/(X^b - 1)$, i.e, the polynomial ring with binary coefficients reduced modulo $X^b - 1$ such that $(X^b - 1)/(X - 1)$ is irreducible. It is well known that every polynomial in R can be represented by its coefficient vectors in $\mathbb{Z}_2^b$. Pick $\mathbf{q}_1 \xleftarrow{U} \mathbb{Z}_2^b$ and define $\mathbf{q}_i := X^{i-1}\mathbf{q}_1, i = 1, ...b$, meaning that we shift the entries in the coefficient vectors $\mathbf{q}_1$ by $i - 1$ times. Hence, we can construct a $b \times b$ matrix $\mathbf{Q}$ by shifting the first row to the left consecutively $b - 1$ times. The secret matrix $\mathbf{M}$ is obtained by generating the first m rows, then dropping the last $b - n$ columns of $\mathbf{Q}$. Therefore, the secret key of Firekite PRNG is, in fact, the random b-bit vector $\mathbf{q}_1$ rather than an $m \times n$ matrix $\mathbf{M}$. The designers conjectured that using such $\mathbf{M}$ does not substantially reduce the security compared to a fully random matrix $\mathbf{M}$.

Table 1 shows a few sets of suggested parameters for Firekite that correspond to 80 and 128 security bit levels. Other proposed parameter sets can be found in [Bog+21b].

To derive an estimation of the concrete security of Firekite, one faces two problems: first, the noise vectors from Firekite has weight at most k and the noise distribution is not binomial, as opposed to a standard LPN instance. Second, an adversary only sees a part of the noisy product. Therefore, it is necessary to

Table 1: Firekite's parameters for 80 and 128-bit security with the properties in terms of key size b, and number of randomization rounds r.

Parameters			Properties		
m	n	k	Key size (b)	r	Security level
216	1024	16	1061	18	82.76
216	2048	32	2053	18	82.76
216	16,384	216	16,421	21	80.68
352	2048	32	4099	18	129.07
352	8192	120	8219	18	128.99
352	16,384	228	16,421	18	128.93

prove that using Firekite noise distribution for an LPN variant is still hard, and the underlying problem of solving Firekite is as hard as LPN.

The first problem is solved as follows. Let $\Delta(\mathbf{e})$ denote the Hamming weight of an n-bit vector and e_j the j-th bit of $\mathbf{e}$. If $\mathbf{e}$ comes from Firekite and assume each $\mathbf{c}_e$ is uniformly distributed, then $\mathbf{Pr}[e_j = 0] = \left(\frac{n-1}{n}\right)^k$. Therefore, the expected Hamming weight of the Firekite noise (denoted by $\Delta^{\text{Firekite}}(\mathbf{e})$) is

$$\mathbb{E}[\Delta^{\text{Firekite}}(\mathbf{e})] = n\left(1 - \left(\frac{n-1}{n}\right)^k\right),$$

and one can show that

$$\frac{2}{3}k < \mathbb{E}[\Delta^{\text{Firekite}}(\mathbf{e})] < k.$$

In a standard LPN problem with parameters η and m, $\mathbb{E}[\Delta^{\text{LPN}}(\mathbf{e})] = \eta m$ and $\mathbf{Pr}[\Delta^{\text{LPN}}(\mathbf{e}) = \lfloor \mathbb{E}[\Delta^{\text{LPN}}(\mathbf{e})] \rfloor] \in \Omega(1/n)$. Therefore, given such an LPN instance, we set k such that $\eta m \leq k$, e.g., $k := \frac{3}{2}\eta m$. Then the noise of this LPN instance could come from the Firekite noise distribution with probability at least $\Omega(1/n)$. In other words, if the LPN instance with the Firekite noise distribution can be broken efficiently, any standard LPN instance can also be broken with $O(n)$ more work.

As for the underlying problem of solving Firekite, Firekite's designers were able to show that it is *at most* as hard as the LPN problem, and they also conjectured that the reverse is also true [Bog+21b]. Using this transformation to attack Firekite with the most efficient LPN-solving algorithm, namely the one by Levieil and Fouque [LF06a], they were able to derive the concrete proposed parameters for the different security levels.

2.3 The problem of observing noisy codewords from an unknown code

The task of recovering *partially* the secret matrix $\mathbf{M}$ (by observing vectors $\mathbf{g}_i$) can be seen as identifying an unknown code by observing noisy codewords. The problem often arises in different contexts [MGB12], especially in analyzing cryptosystems where encryption involves error-correcting codes and the transmission is carried over a noisy channel (e.g., a binary symmetric channel). General approaches consist of three steps: first, arranging noisy codewords as rows of a matrix, then running the Gaussian elimination, and finally from the non-echelon part finding sums of vectors that are candidates to construct dual codewords (i.e, parity-check equations). Instead of looking at only columns that sum to 0, Sicot, Houcke and Barbier argued that sparse sums of columns can also be candidates for being dual codewords [BSH06; SHB09]. Therefore, the last step can be reduced to an instance of the well known *close neighbors search problem*. Beside the *projection method* proposed by Cluzeau and Finiasz [CF09], which aimed to find sparse sums of p columns (with complexity of order $\Omega(n^{p/2})$ when p is even) that are equal in some positions using birthday paradox and hashtables, there have been many improvements and extensive studies to the close neighbors search problem recently. In particular, one being the *Dubiner method* which later was applied by Carrier and Tillich in their generalized approach [CT19]. Their algorithm only performed a *partial* Gaussian elimination for the second steps. The argument is that the Gaussian elimination increases the noise by combining noisy codewords, hence it is more likely to obtain sparse sums in the early stage of the Gaussian elimination and minimizing the dual codewords that might have been undetected by Sicot-Houcke-Barbier algorithm [CT19]. Moreover it also allowed them to find dual codewords of much larger weight (compared to the full Gaussian elimination) with reasonable complexities.

In practice, the recovery of an unknown code by observing noisy codewords concerns useful families of codes, such as cyclic codes, convolutional codes, turbo codes, or the ubiquitous LDPC, which is important as finding low-weight dual codewords is essential in determining communication components such as unknown interleaver [BSH06; Tix15] or reconstructing other families of codes.

In the next section we introduce a new method, namely finding a small number of noisy codewords summing to the zero codeword through a generalized birthday type of algorithm.

3 The proposed distinguishing algorithm

In this section, we aim to give a brief description of the idea used in our distinguishing attacks on Firekite. We firstly observe that the secret key matrix $\mathbf{M}$ is fixed throughout the rounds of Firekite; hence, the keystream output by Firekite PRNG is subjected to accumulating non-randomness. Let us look at the Firekite

PRNG, fulfilling

$$\mathbf{vM} + \mathbf{e} = (\mathbf{g}||\mathbf{v}'||\mathbf{c}_e),$$

where $\mathbf{v}'$ and $\mathbf{c}_e$ are used in the next iteration by assigning $\mathbf{v}' = \mathbf{v}$ and $\mathbf{e} = \bigvee_{j=1}^{k} \mathbf{b}_{\mathbf{c}_j}$, and $\mathbf{g}$ is the PRNG's output. In the initial part of the attack, we concentrate on assuming the knowledge of

$$\mathbf{g} = \mathbf{vM}' + \mathbf{e}',$$

where $\mathbf{g}$ is a known d-bit vector, $\mathbf{M}'$ is now considered as an $m \times d$ secret binary matrix (obtained from the first d columns of the original $\mathbf{M}$ matrix) and $\mathbf{e}'$ is a secret d-bit noise vector, being the first d positions of $\mathbf{e}$. It is known that $\Delta(\mathbf{e}) \leq k$ (which is small); hence, the weight of $\mathbf{e}'$ is also small. The expected weight of $\mathbf{e}'$ denoted by $\hat{k}$, where $\hat{k} = \frac{k \cdot d}{n}$ since it is assumed that the ones in $\mathbf{e}$ are uniformly distributed among all d positions.

In a synchronous stream cipher attack, we assume that an adversary has access to a long output stream, which means access to a large number of d-bit vectors $\mathbf{g}$. The set of these vectors is written as $\{\mathbf{g}_i, i = 1, \ldots, S\}$, where now $\mathbf{g}_i = \mathbf{v}_i\mathbf{M}' + \mathbf{e}'_i$ and for some S to be addressed in the following subsections. We first sketch the ideas behind our distinguishing attack, i.e., given an aforementioned set of vectors, decide whether they originate from Firekite or if they are random vectors.

Our goal is to find a subset of $\mathbf{g}_{i_j}$ vectors, $j = 1, \ldots, \ell$, such that the corresponding $\sum_{j=1}^{\ell} \mathbf{v}_{i_j} = \mathbf{0}$, i.e., we find a set of noisy codewords such that the underlying information vectors sum to zero. If ℓ vectors $\mathbf{v}_{i_j}$, $j = 1, \ldots, \ell$, sum to zero, then the sum of the corresponding $\mathbf{g}_{i_j}$ is expected to be of weight $c_\omega = \lceil \ell \cdot \hat{k} \rceil$ with nonzero contributions coming only from the errors $\mathbf{e}'_{i_j}$. Indeed, we then have

$$\sum_{j=1}^{\ell} \mathbf{g}_{i_j} = \sum_{j=1}^{\ell} \mathbf{v}_{i_j}\mathbf{M}' + \sum_{j=1}^{\ell} \mathbf{e}'_{i_j} = \sum_{j=1}^{\ell} \mathbf{e}'_{i_j}.$$

Therefore, when ℓ is not too large, e.g. $\ell = 4$ or $\ell = 8$, the expected weight in $\sum_{j=1}^{\ell} \mathbf{g}_{i_j}$ will be low if $\sum_{j=1}^{\ell} \mathbf{v}_{i_j} = 0$. Since d is much larger than c_ω (with proposed parameters for Firekite), such a weight is very unlikely if the vectors $\mathbf{g}_{i_j}$ are random vectors. In the Firekite PRNG, such a collision of vectors of length m (i.e, with probability proportional to 2^{-m}) guarantees a low weight vector of length d. It is only intuitive to deduce that we can detect such occurrences more frequently than what is expected in the random case.

3.1 A basic algorithm for finding noisy codewords summing to the zero codeword

Recall that we want to find ℓ different $\mathbf{g}_{i_j}$ vectors, $j = 1, \ldots, \ell$, such that the associated unknown vectors $\mathbf{v}_{i_j}$ sum to zero. Our approach is built from ideas from the generalized birthday attack [Wag02] and the BKW algorithm [BKW03a]. A different but related approach is also May et al.'s *Match-and-Filter* algorithm [BM17b].

In a simplified description following [Wag02], we set up ℓ ($\ell = 2^t$ is a power of 2) lists of size 2^c filled by $\mathbf{g}_i$ vectors. We then combine the lists pairwise, resulting in a new list containing vectors created as a sum of two vectors, one from each initial list, such that some c predetermined positions are all zero. The expected number of vectors in the new list is 2^c. After the first step, we have $\ell/2$ lists. We then perform the same procedure again, reducing another c positions to zero until one single list remains, i.e, after t steps. In the remaining list, we will finally examine whether there are vectors $\sum_{j=1}^{\ell} \mathbf{g}_{i_j}$ that are candidates to satisfy $\sum_{j=1}^{\ell} \mathbf{v}_{i_j} = 0$. In fact, they are quite easily detected, since if this is the case then $\sum_{j=1}^{\ell} \mathbf{g}_{i_j} = \sum_{j=1}^{\ell} \mathbf{e'}_{i_j}$, which has very low weight.

As in the BKW algorithm framework, one may use the same list for $\mathbf{g}_i$ vectors and we increase the list size to roughly $3 \cdot 2^c$. Starting with a list $L^{(0)}$, we can write up a sequence of updated lists $L^{(0)} \rightarrow L^{(1)} \rightarrow L^{(2)} \cdots \rightarrow L^{(t)}$, where in each step we reduce another c positions. This means that $L^{(i)}$ have vectors where the first $i \cdot c$ positions are all zero. On average, there are three vectors that collide in given c positions. Therefore, we can have three combinations for such vectors and the size of $L^{(i)}$ can be kept (hence, the motivation for the factor 3). We formally describe this approach in Algorithm I.2.

Figure 1 and Figure 2 are visualizations of the COMBINE step for the $(i-1)$-th list $L^{(i-1)}$ and the filtering for the last list, respectively.[3]

We need to consider complexity and memory of the algorithm. Let this computational complexity measured in simple operations be denoted C and the used memory in bits be denoted Mem. Its main parts are the $L^{(i)} = \text{COMBINE}(L^{(i-1)})$ steps in the loop. We assume that the vectors in the list $L^{(i-1)}$ are organized in a hash table. We have that the first $(i-1) \cdot c$ positions are all zero in all vectors in $L^{(i-1)}$, and they are again sorted in different buckets in the hash table according to the value of the next c positions, i.e., position $(i-1) \cdot c$ to $i \cdot c - 1$, for $i = 1, \ldots, t$. The COMBINE step now creates new vectors for the new list $L^{(i)}$ by adding together all possible pairs that are stored in the same bucket. This will cancel out another c positions so that vectors in $L^{(i)}$ start with $i \cdot c$ zeros. New vectors are created until the list $L^{(i)}$ has cardinality $3 \cdot 2^c$ and the sorting procedure

[3] Inspired by Erik Mårtensson's poster "Coded-BKW with Sieving"[Mår18].

Algorithm I.2: t-step Distinguisher

Input: A list $L^{(0)}$, with $\mathbf{g}_i \in \mathbb{Z}_2^n$, $i = 1, \ldots, 3 \cdot 2^c$ ($|L^{(0)}| = 3 \cdot 2^c$), parameters $k, d, \ell, t = \log l$, $c_\omega = \lceil \frac{\ell \cdot k \cdot d}{n} \rceil$.

1 **for** $i = 1, \ldots, t$ **do**
2 $\quad L^{(i)} = \textsc{Combine}(L^{(i-1)})$; `// Combine list, cancelling` c `bits.`
3 minweight $= k \cdot \ell$;
4 **for** $\mathbf{g}'$ **in** $\mathbf{L^{(t)}}$ **do**
5 $\quad$ **if** HammingWt($\mathbf{g}'$) $\leq$ minweight **then**
6 $\quad\quad$ minweight = HammingWt($\mathbf{g}'$) ; `// Filtering low weight sums.`
7 **if** minweight $\leq c_\omega$ **then**
$\quad$ **Return:** 'Firekite'
8 **else**
$\quad$ **Return:** 'Random'

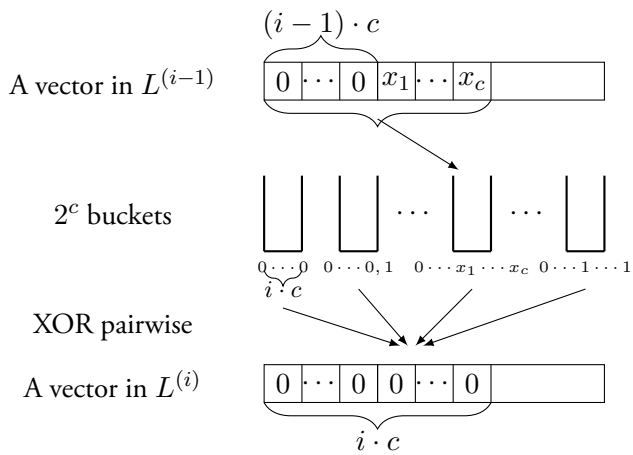


Figure 1: Combine for $L^{(i-1)}$.

is repeated for the next iteration.[4] The complexity of one Combine step is then $3 \cdot 2^c$ bit-wise additions of vectors of length at most d and storing the result in memory. We adopt Firekite's designers' notation by letting p be the word-length of

[4] The input could be not uniformly distributed and we might have more combinations than $3 \cdot 2^c$. If we obtain significantly fewer vectors after Combine (unlikely) due to input non-uniformity, the list size gradually decreases and Algorithm I.2 might fail.

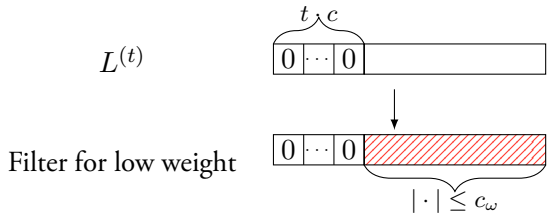


Figure 2: Filter $L^{(t)}$ with c_ω.

a bit-wise addition operation, i.e, the number of bits for which an XOR operation can be computed.[5]. We write the cost of one d-bit XOR operation as $(1 + \lfloor d/p \rfloor)$ This procedure is repeated t times in Algorithm I.2. The final check for low weight vectors actually does not need to go through all buckets, but only those with a low weight (for instance, one can sort the vectors in $L^{(t)}$ by their next c positions). This cost is then much smaller than the previous steps and can be disregarded. The complexity can thus be estimated as

$$C = t \cdot (3 \cdot 2^c) \cdot (1 + \lfloor d/p \rfloor). \tag{2}$$

The required memory M is the storage of two lists, altogether at most $M = 2 \cdot 3 \cdot 2^c \cdot d$ in bits. In the next subsection, we investigate the success probability of the distinguisher.

3.2 Parameter choices and the success probability of the proposed algorithm

Algorithmic steps

Since the added noise in the $\sum_{j=1}^{\ell} \mathbf{g}_{i_j} = \sum_{j=1}^{\ell} \mathbf{v}_{i_j}\mathbf{M}' + \sum_{j=1}^{\ell} \mathbf{e}'_{i_j} = \sum_{j=1}^{\ell} \mathbf{e}'_{i_j}$ expression becomes significant as ℓ grows large, a low-weight sum from Firekite will become hard to distinguish as ℓ grows (from the random case). We hence fix the number of algorithmic steps t to 2 or 3, corresponding to $\ell = 4$ and $\ell = 8$ in the proposed algorithm, respectively.

The required Firekite output observations

A vector formed as $\sum_{j=1}^{\ell} \mathbf{g}_{i_j} = \sum_{j=1}^{\ell} \mathbf{v}_{i_j}\mathbf{M}' + \sum_{j=1}^{\ell} \mathbf{e}'_{i_j} = \sum_{j=1}^{\ell} \mathbf{e}'_{i_j}$ will be called a *zero sum vector*. Furthermore, considering a sum of error vectors, e.g., $\sum_{j=1}^{\ell} \mathbf{e}'_{i_j}$, we say that a position is *error free* mod 2 if $\sum_{j=1}^{\ell} \mathbf{e}'_{i_j}$ is zero in that position; we say that a position is simply *error free* if all $\mathbf{e}'_{i_j}$ are zero in the position.

[5]For example, the *Advance Vector Extension* AVX-512 allows XOR to have 512 bits computed per cycle.

It can happen that a *double error* event occurs, i.e., two ones in the same position and $1 + 1 = 0$. The required Firekite output observations (i.e., $|L^{(0)}| = 3 \cdot 2^c$) has to be chosen such that zero sum vectors can be found after Algorithm I.2. Moreover, they must be error free mod 2 in the first $t \cdot c$ positions. This probability is denoted P_{nf} (noise-free), and we investigate this probability for both case $\ell = 4$ and $\ell = 8$.

The case $\ell = 4$ Starting with $\ell = 4$, we are interested in knowing if a zero sum vector can be found in the final list. The expected number of zero sum vectors in the final list is denoted by N. We have

$$N = \binom{3 \cdot 2^c}{4} \cdot 2^{-m} \cdot 3 \cdot 2^{-c} \cdot P_{\text{nf}} \tag{3}$$

such zero sum vectors, which can be roughly explained as follows: there are $\binom{3 \cdot 2^c}{4}$ possible combinations from the initial list. Among all such 4-sums, only a fraction 2^{-m} will correspond to a zero sum in $\mathbf{v}_{i_j}, j = 1, \ldots, 4$. Then, there are 3 ways to choose 2 pairs as in Algorithm I.2. We consider two particular pairs $\{\mathbf{g}_{i_1}, \mathbf{g}_{i_2}\}$ and $\{\mathbf{g}_{i_3}, \mathbf{g}_{i_4}\}$ summing to a zero sum, we further condition $\mathbf{g}_{i_1}$ and $\mathbf{g}_{i_2}$ to cancel in the first c bits with probability 2^{-c} (the other pair automatically follows). Finally, we assume that $\mathbf{e}'_{i_1} + \mathbf{e}'_{i_2} + \mathbf{e}'_{i_3} + \mathbf{e}'_{i_4}$ is zero in the first $2c$ positions, i.e, error free mod 2.

P_{nf} can be bounded by the probability that the first $2c$ positions are error free. For each $\mathbf{e}'_{i_1}$, there are at most k bits set, uniformly distributed among n positions,[6] so the probability of one error vector, being error free in the first $2c$ position, is roughly $((n - 2c)/n)^k$. Therefore, we have

$$P_{\text{nf}} \geq ((n - 2c)/n)^{4k}.$$

Lemma 1. *When $\ell = 4$, we expect to have*

$$N > \binom{3 \cdot 2^c}{4} \cdot 2^{-m-c} \cdot 3 \cdot \left(\frac{n - 2c}{n}\right)^{4k} \tag{4}$$

zero sum vectors in the final list in Algorithm I.2.

The case $\ell = 8$ Next, we investigate $\ell = 8$. Similar to the case $\ell = 4$, we have:

$$N = \binom{3 \cdot 2^c}{8} \cdot 2^{-m} \cdot 105 \cdot 2^{-4c} \cdot P_{\text{nf}}. \tag{5}$$

[6] The expected weight of $\mathbf{e}'$ from Firekite is smaller than k, but we can compute probabilities from error assignment in $\mathbf{e}$.

The explanation is again as follows: the number of different sums of 8 vectors that can be constructed is $\binom{3\cdot 2^c}{8}$. Among them, we expect a fraction of 2^{-m} summing to the zero case. There are $7 \cdot 5 \cdot 3 = 105$ ways to form 4 pairs of 8 vectors. Consider the particular pairing $\{\mathbf{g}_{i_1}, \mathbf{g}_{i_2}\}$, $\{\mathbf{g}_{i_3}, \mathbf{g}_{i_4}\}$, $\{\mathbf{g}_{i_5}, \mathbf{g}_{i_6}\}$, and $\{\mathbf{g}_{i_7}, \mathbf{g}_{i_8}\}$. A sum constructed from this pairing will be in the final list of Algorithm I.2 if $\mathbf{g}_{i_1} + \mathbf{g}_{i_2}$, $\mathbf{g}_{i_3} + \mathbf{g}_{i_4}$ and $\mathbf{g}_{i_5} + \mathbf{g}_{i_6}$ are all zero in the first c positions. Then $\mathbf{g}_{i_7} + \mathbf{g}_{i_8}$ has to be zero in the first c positions. The probability of this event for each choice of fixed indices is 2^{-3c}. Similarly to the 4-sum, now $(\mathbf{g}_{i_1} + \mathbf{g}_{i_2}) + (\mathbf{g}_{i_3} + \mathbf{g}_{i_4})$ must sum to zero in the next c positions, with probability 2^{-c}. Finally, we also need the sum of error vectors to be error free mod 2 in $3c$ positions.

As before, P_{nf} can be bounded by the probability that no errors occur in the first $3c$ positions. The probability of such a distribution for a single error vector is then roughly $((n-3c)/n)^k$ and for all eight of them we have

$$P_{\text{nf}} \geq ((n-3c)/n)^{8k}.$$

Lemma 2. *When $\ell = 8$, we expect to have*

$$N > \binom{3\cdot 2^c}{8} \cdot 2^{-m-4c} \cdot 105 \cdot \left(\frac{n-3c}{n}\right)^{8k} \tag{6}$$

zero sum vectors in the final list in Algorithm I.2.

For $\ell = 8$ there are more errors in general, meaning that P_{nf} is much smaller compared to $\ell = 4$. This gives a stronger motivation for examining other error patterns such as the double errors canceling out. In particular, the sums from our algorithm can have $1 + 1 = 0$ in the first $3c$ bits. More specifically, if two error vectors have a one in the same position, their combination still survives the Combine step in Algorithm I.2. For some parameters proposed by Firekite's designers, certain double error events are even more likely than having no error at all in the first $3c$ positions and thus should not be neglected. Since the error vectors are sparse (e.g., $k = 16 \ll n = 1024$), if a double error occurs at a position, it most likely happens only **once**, i.e, coming from one pair of $\mathbf{g}_{i_j}$ (or equivalently, $\mathbf{e}'_{i_j}$). Having four ones in the same position is exceedingly rare for interested parameters (see Appendix, Example 1). Therefore, we can have a lower bound of P_{nf} by considering only non-repeating double errors. Let us look at the simple case where errors from Firekite have exactly k bits set.[7]

Assume we have $\epsilon \leq k$ double errors, and the probability is denoted by P_ϵ. Then P_ϵ is equal the sum of all possible error patterns/combinations of $\mathbf{g}_i$ vectors,

[7] The expected weight of errors from Firekite can be smaller than k. Hence using binomial expressions, while not entirely correct, gives a good approximation.

provided they result in ϵ collisions. One writes

$$\epsilon = \sum_{i,j>i} \epsilon_{ij},$$

where ϵ_{ij} denotes the number of double errors between $\mathbf{g}_i$ and $\mathbf{g}_j$. The total number of errors in $3 \cdot c$ positions of $\mathbf{g}_i$ is $\epsilon_i = \sum_j \epsilon_{ij}$. Hence

$$P_\epsilon = \sum_{\{(\epsilon_{ij})\}} P_{\epsilon,\{(\epsilon_{ij})\}},$$

where $\{(\epsilon_{ij})\}$ is an eligible error colliding pattern of $\mathbf{g}_i$ vectors with the corresponding probability $P_{\epsilon,\{(\epsilon_{ij})\}}$.

Lemma 3. *Let $\mathbf{g}_i, i = 1, \ldots, 8$ be binary vectors. Then, the noise-free probability of $\sum_{i=1}^{8} \mathbf{g}_i$ in the first $3 \cdot c$ bits is*

$$P_{\text{nf}} = \sum_{\epsilon=0,\ldots,k} P_\epsilon = \sum_{\substack{\epsilon=0,\ldots,k \\ \{(\epsilon_{ij})\}}} P_{\epsilon,\{(\epsilon_{ij})\}}, \tag{7}$$

where

$$P_{\epsilon,\{(\epsilon_{ij})\}} \approx \prod_{i=1}^{8} \binom{k}{\epsilon_i} \left(\frac{3c}{n}\right)^{\epsilon_i} \left(\frac{n-3c}{n}\right)^{k-\epsilon_i} \frac{\binom{\epsilon_1}{\epsilon_{1i}}\binom{\epsilon_2-\epsilon_{12}}{\epsilon_{2i}} \cdots \binom{3c-\epsilon_{1i}-\ldots-\epsilon_{i-1i}}{\epsilon_i-\epsilon_{1i}-\ldots\epsilon_{i-1i}}\binom{\epsilon_{i-1}-\epsilon_{1i-1}-\ldots\epsilon_{i-2i-1}}{\epsilon_{i-1i}}}{\binom{3c}{\epsilon_i}}.$$

Proof. Assume ϵ double errors and a fixed error colliding pattern $\{(\epsilon_{ij})\}$. Without loss of generality, we further assume that $\epsilon_i \geq \epsilon_j$ for $i < j$, i.e., $\mathbf{g}_1$ has the most errors in the first $3c$ bits. The probability of $\mathbf{g}_i$ having ϵ_i errors in the first $3c$ bits is

$$\binom{k}{\epsilon_i} \left(\frac{3c}{n}\right)^{\epsilon_i} \left(\frac{n-3c}{n}\right)^{k-\epsilon_i}.$$

We also requires $\mathbf{g}_2$ to have ϵ_{12} colliding positions out of ϵ_2. This probability is

$$\frac{\binom{\epsilon_1}{\epsilon_{12}}\binom{3c-\epsilon_{12}}{\epsilon_2-\epsilon_{12}}}{\binom{3c}{\epsilon_2}}.$$

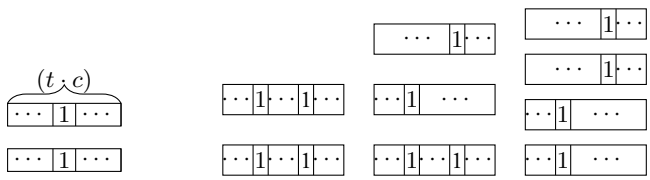
Similarly, for vector $\mathbf{g}_3$, the colliding probability is

$$\frac{\binom{\epsilon_1}{\epsilon_{13}}\binom{\epsilon_2-\epsilon_{12}}{\epsilon_{23}}\binom{3c-\epsilon_{13}-\epsilon_{23}}{\epsilon_3-\epsilon_{13}-\epsilon_{23}}}{\binom{3c}{\epsilon_3}}.$$

Generalizing for $\mathbf{g}_i$ and the lemma follows. □

However, it is not practical to take into account all possible double error events. For instance, if the expected numbers of 1's in the first $t \cdot c$ positions for each error vector is small, e.g., fewer than 2, multiple double error events occur with decreasingly small probability (from a certain point). Moreover, they do not contribute substantially to our estimate (Appendix, Example 2). Moreover, an improvement in estimating P_{nf} only suggests that we need less input for Algorithm I.2, i.e., the bigger P_{nf} is, the smaller c to satisfy (5). A reasonable approximation suffices for us to deduce the necessary initial list size $|L^{(0)}|$ so that the expected number of zero sums is $N > 1$.

To illustrate this argument, we consider the case where we have the relative Hamming weight of the error vectors $\mathbf{e}_i$'s in the first $3 \cdot c$ positions to be slightly larger than 2 ($\ell = 8$). Hence, we focus only on the scenarios of up to 2 double errors in Figure 3.



(a) 1 double error **(b)** 2 double errors

Figure 3: Illustration for the colliding patterns

One can find the inspiration from Wagner ℓ-tree algorithm [Wag02] in Algorithm I.2, namely, by consecutively canceling out c bits. Wagner argued that one needs lists of size $O(2^{\frac{m}{1+\log \ell}})$ to have a solution in the exact ℓ-*list birthday problem*. In our algorithm, we need slightly more,[8] i.e, $O(2^{\frac{m}{1+\log \ell}+a})$ where a depends on P_{nf}. Note that P_{nf} remains relatively the same if $c \approx \frac{m}{1+\log \ell}$. Therefore, we initially set $c = \lfloor \frac{m}{1+\log \ell} \rfloor$, then raising until we get $N > 1$. Finally, we verify $N > 1$ again with P_{nf} estimated by said c.

The success probability

Previously, we have seen that if we choose parameters suitably, we can detect zero sum vectors in the final list. We now need to check whether low weight sums can stem from random vectors. In other words, the zero sums must be easily distinguished from those coming from the random case.

Assume Algorithm I.2 outputs 'Firekite' for the random case. This means that it has found a vector in the final list of weight at most c_ω. It is thus of interest to derive the likelihood of such a vector in the random case. Recall $\Delta(\mathbf{g})$ as the

[8] Wagner showed that one can find $\alpha^{1+\log \ell}$ more solutions at the expense of α times more work, provided $\alpha \leq 2^{m/(\log \ell \cdot (1+\log \ell))}$ [Wag02].

Hamming weight of a binary vector $\mathbf{g}$, and let $\mathbf{g}_{[i]}$ be the i-bit truncated $\mathbf{g}$ (first i positions). A vector in the final list will have the first $t \cdot c$ positions all zero, but the remaining positions $d - t \cdot c$ positions are just formed by XOR-ing ℓ random bit values; thus, they are independent and uniformly distributed on $\{0, 1\}$. The probability of such a vector having Hamming weight at most c_ω is

$$\mathbf{Pr}\left[\Delta(\mathbf{g}) \leq c_\omega : \mathbf{g} \in \mathbb{Z}_2^d, \mathbf{g}_{[t\cdot c]} = 0\right] = \frac{\sum_{i=0}^{c_\omega} \binom{d-t\cdot c}{i}}{2^{d-t\cdot c}},$$

and the expected number of vectors of weight at most c_ω, denoted by N_{random}, in the final list is

$$N_{\text{random}} = 3 \cdot 2^c \cdot \frac{\sum_{i=0}^{c_\omega} \binom{d-t\cdot c}{i}}{2^{d-t\cdot c}}. \tag{8}$$

Information theoretically, we have an approximation[9] as

$$N_{\text{random}} = 3 \cdot 2^c \cdot \frac{\sum_{i=0}^{c_\omega} \binom{d-t\cdot c}{i}}{2^{d-t\cdot c}} \approx \sum_{i=0}^{c_\omega} 2^{-\left(1-H(\frac{i}{d-t\cdot c})\right)(d-t\cdot c)+c} \approx 2^{-\left(1-H(\frac{c_\omega}{d-t\cdot c})\right)(d-t\cdot c)+c},$$

where H is the binary entropy function and $H(p) = -p\log(p) - (1-p)\log(1-p)$ with $p \in (0, 1)$. Therefore, if there exists a low weight sum in the final list $L^{(t)}$ and N_{random} is 'vanishingly small' (i.e., $N_{\text{random}} \ll 1$), we have shown that the Firekite's output vectors are indeed not random. Different values of N_{random} for various parameters can be found in Table 2.

4 Results for the distinguisher

In this section, we give the results for our distinguishing attack as described in Section 3 when it is applied to the suggested parameter sets for Firekite.

4.1 Theoretical complexity estimation for the proposed parameters of Firekite

We investigate the results for Firekite's proposed parameters. As an example, we explain our distinguishing attack for the case $n = 1024, m = 216, d = 648, k = 16$, where the claimed security level is 82.

1. For $\ell = 4$ we derive the following: we pick $c = 76$, $c_w = \lceil 4 \cdot \hat{k} \rceil = 41$, where $\hat{k} = \frac{k\cdot d}{n}$. Let P_0 denote the probability of the first $2\cdot c$ position being error free. By simply setting $P_{\text{nf}} \approx P_0 = \left(\frac{n-2\cdot c}{n}\right)^{4k} \approx 2^{-14.83}$, we get $N > 1.42$ and

[9] We use $2^n \binom{n}{i} \approx 2^{(1+H(i/n))n}$. The final approximation is due to overwhelming contribution of $2^{-(1-H(c_\omega/(d-t\cdot c)))(d-t\cdot c)+c}$.

$$N_{\text{random}} \approx 2^{-215.76}.$$

2. For $\ell = 8$ we derive the following: picking $c = 62$ and similarly, $c_\omega = \lceil \ell \cdot k \cdot \frac{d}{n} \rceil = 81$, we have N_{random} very close to zero.

   As an example, we approximate P_{nf} by the sum of probabilities of no double errors P_0, one double errors P_1, and two double errors P_2 ($\epsilon = 0, 1, 2$). As discussed, many double errors are improbable and we focus on the most likely cases.

   - If there is no double error, $P_0 = \left(\frac{n-3c}{n}\right)^{8k} \approx 2^{-37.02}$.
   - Assume there is one double error occurring. For the colliding pair of vectors, the probability of having exactly one 1 in the same position $\frac{k^2}{3c}\left(\frac{3c}{n}\right)^2\left(\frac{n-3c}{n}\right)^{2(k-1)}$, and there are $\binom{8}{2}$ ways to select a pair/colliding pattern $\{(\epsilon_{ij})\}$, hence

   $$P_1 \approx \binom{8}{2}\frac{k^2}{3c}\left(\frac{3c}{n}\right)^2\left(\frac{n-3c}{n}\right)^{2(k-1)}\left(\frac{n-3c}{n}\right)^{6k} \approx 2^{-36.09}.$$

   - If there are two double errors, then there are two cases: the double errors happen in one pair or two pairs (note that a vector in the first pair can appear in the second pair). Let P_{21} and P_{22} denoted such events, respectively, then

   $$P_2 \approx P_{21} + P_{22},$$

   where

   $$P_{21} \approx \binom{8}{2}\frac{\binom{k}{2}^2}{\binom{3c}{2}}\left(\frac{3c}{n}\right)^4\left(\frac{n-3c}{n}\right)^{2(k-2)}\left(\frac{n-3c}{n}\right)^{6k},$$

   and

   $$P_{22} \approx 2\binom{8}{3}\binom{k}{2}\left(\frac{3c}{n}\right)^2\left(\frac{n-3c}{n}\right)^{k-2}\left[\frac{k}{3c}\left(\frac{3c}{n}\right)\left(\frac{n-3c}{n}\right)^{(k-1)}\right]^2\left(\frac{n-3c}{n}\right)^{5k} + \binom{8}{2}\binom{6}{2}\left(\frac{k^2}{3c}\right)^2\left(\frac{3c}{n}\right)^4\left(\frac{n-3c}{n}\right)^{4(k-1)}\left(\frac{n-3c}{n}\right)^{4k}.$$

   Therefore, $P_2 \approx 2^{-35.86}$, and $P_{\text{nf}} > P_0 + P_1 + P_2 \approx 2^{-34.65} \approx 4P_0$ which gives $N > 2.7$. The failure probability for this attack when $c_\omega = 81$ can be indicated by

   $$N_{\text{random}} \approx 2^{-90.23}.$$

Table 2 shows our attack's complexity and the corresponding N_{random} for a few sets of parameters suggested by the Firekite's designers. The number of required Firekite output observations is indicated by the parameter c. Recall that the theoretical complexity is

$$C = t \cdot (3 \cdot 2^c) \cdot (1 + \lfloor d/p \rfloor).$$

In their implementation, beside several optimization flags, they also use a compilation flag `-mavx2` that allows XOR operations to apply on 256 bits per cycle. Therefore, in our complexity estimates, we set $p = 256$.

Table 2: Our distinguishing attack complexity corresponding to a few selected sets of parameters for 80-bit and 128-bit security of Firekite's stream cipher.

Parameters				c		Attacks(log)		N_{random}(log)	
m	n	k	Security	4-sum	8-sum	4-sum	8-sum	4-sum	8-sum
216	1024	16	82.76	76	62	80.17	66.75	-215.76	-90.23
216	2048	32	82.76	76	62	81.17	67.75	-765.79	-465.74
216	16,384	216	80.68	75	60	83.28	68.87	-9011.62	-6541.71
352	2048	32	129.07	125	101	130.16	106.75	-541.40	-275.94
352	4096	58	128.95	124	99	130.17	105.75	-1739.41	-1150.69
352	16,384	228	128.93	123	99	131.26	107.84	-8510.19	-6023.39

With 4-sum attacks and for small parameters of 80-bit secured Firekite, we can only refine Firekite's designer estimates marginally. However, our 8-sum distinguisher manages to break Firekite for all parameters, except for the smallest 128-bit secure instance, which is $n = 1024, m = 352$, and $k = 16$. In particular, we can find a zero sum with the cost $2^{107.75}$ but $\log(N_{\text{random}}) \approx 83$. Therefore, we were unable to claim that the Firekite's output is not randomly distributed as the low weight sums found could easily come from random vectors. The explanation is that $d = n - m - k \log n$ is not so large compared to $8 \cdot \hat{k}$ in this case; hence, it is impossible to distinguish from the case of random vectors $\mathbf{g}_i$. In general, 8-sum attack performs slightly better when the parameters n and k grows (with the same factor, as suggested by Firekite's designers). This is owing to the fact that d grows bigger while m remains relatively unchanged; hence we have even smaller failure probability and bigger error free probability P_{nf}. In fact, we need a smaller initial list ($3 \cdot 2^{60}$ compared to $3 \cdot 2^{62}$) when attacking Firekite instance with $n = 16384, m = 216, k = 216$.

These theoretical results above can be improved; larger Firekite parameters make the double error events more probable. For instance, attacking the parameters $n = 16384, m = 352, k = 228$ with 8-sum distinguisher, we find that two double errors (P_2) are twice as likely as no error (P_0). Therefore, P_{nf} should be better approximated by taking, e.g., P_3 and P_4 into consideration.

4.2 Simulation results for smaller parameters

We verify our approach and formulas by performing simulations.[10] As a toy example, we set up a mini version of Firekite with small parameters (where the ratios $\frac{n}{k}$ are kept constant as in Table 1 and m is also reduced by a similar factor) and run Algorithm I.2.

Our parameters are $m = 52, n = 256, k = 4, b = 269$ and $r = 15$. Recall that b is the secret key's length used to generate the first row of Firekite's secret matrix $\mathbf{M}$ such that $(X^b - 1)/(X - 1)$ is irreducible in $\mathbb{Z}_2[x]$ and r is the number of randomization rounds before Firekite generates its actual output.

- For the 4-sum distinguisher, the filter weight is $c_\omega = 11$. The parameter c is chosen so that

$$N = \binom{3 \cdot 2^c}{4} \cdot 2^{-m} \cdot 3 \cdot 2^{-c} \cdot P_{\text{nf}} > 1.$$

  One has $P_0 = \left(\frac{n-2c}{n}\right)^{4k} \approx 2^{-3.5}$. If we choose $c = 18$, i.e $|L^{(0)}| = 3 \cdot 2^{18}$, there are, on average, less than 1 bit set of the error vectors in the first $2c$ bits. One can safely assume $P_{\text{nf}} \approx P_0$, and it gives $N \approx 3.6$. The simulation returns, on average, 1.65 low weight vectors after 10^2 tests. The discrepancy can be explained as follows: the assumption that we can keep the list's size $|L^{(i)}| = 3 \cdot 2^c$ is often violated as there are **more** vectors after every COMBINE step owing to vectors being not evenly distributed among buckets. Therefore, 'good' combinations that are present in zero sums might be discarded by chance. Keeping all combinations from COMBINE, we obtain more low weight sums after filtering with c_ω (at the cost of higher complexity) and the simulation is more consistent with the theoretical estimate. We now look at the probability of 4 random vectors summing to such sums:

$$\mathbf{Pr}\left[\Delta(\mathbf{g}) \leq c_\omega = 11 : \mathbf{g} \in L^{(2)}, \mathbf{g}_{[2\cdot 18]} = 0\right] = \frac{\sum_{i=0}^{c_\omega} \binom{d-2\cdot c}{i}}{2^{d-2\cdot c}} \approx 2^{-83.76},$$

  therefore,

$$N_{\text{random}} \approx 2^{-64.18}.$$

- For the 8-sum distinguisher, the filter weight is chosen to be $c_\omega = 21$. Again, c must fulfill

$$N = \binom{3 \cdot 2^c}{8} \cdot 2^{-m-4c} \cdot 105 \cdot P_{\text{nf}} > 1,$$

[10] Our simple implementation can be found at https://anonymous.4open.science/r/FirekiteDistinguisher-553B/README.md

where $P_{\text{nf}} \approx P_0 + P_1 + P_2 \approx 2^{-7.67}$. Setting $c = 14$, meaning $|L^{(0)}| = 3 \cdot 2^{14}$, suffices and gives $N > 1.35$. It needs to be clarified that in the 8-sum attack's implementation, the effect of keeping $|L^{(i)}| = 3 \cdot 2^{14}$ is more visible. In particular, we might discard all 'good combinations' when N is very close to 1. We adapt by allowing $|L^{(1)}|$ and $|L^{(2)}|$ to be at most $2 \cdot |L^{(0)}|$, then directly filter combinations from $|L^{(2)}|$ with c_ω. Therefore, the complexity is slightly higher than the theoretical estimate provided in the previous section. We suppose said negative impact can be mitigated when c is large as the vectors in $|L^{(i)}|$ might be more evenly distributed among buckets. The simulation returns 1.52 low weight vector on average after 10^2 tests.

In the random case, the probability of having a vector having Hamming weight up to c_ω is:

$$\mathbf{Pr}\left[\Delta(\mathbf{g}) \leq c_\omega = 21 : \mathbf{g} \in L^{(3)}, \mathbf{g}_{[3\cdot 14]} = 0\right] = \frac{\sum_{i=0}^{c_\omega} \binom{d-3\cdot c}{i}}{2^{d-3\cdot c}} \approx 2^{-50.16},$$

which yields

$$N_{\text{random}} \approx 2^{-34.58}.$$

5 A key-recovery attack on Firekite

In this section we show a possible way to turn the distinguishing attack into a key-recovery attack with slightly higher complexity. We focus on the recovery of the secret $m \times d$ matrix $\mathbf{M}'$. First, we recall that the secret matrix $\mathbf{M}$ in Firekite is constructed by choosing a part of the bigger $b \times b$ matrix $\mathbf{Q}$ as described in Section 2. We pick $\mathbf{q}_1 \xleftarrow{U} \mathbb{Z}_2^b$ and defined rows in the matrix $\mathbf{Q}$ as $\mathbf{q}_i = X^{i-1}\mathbf{q}_1, i = 1, \ldots, b$, i.e., by shifting the first row to the left consecutively $b-1$ times. The secret matrix $\mathbf{M}$ is obtained by dropping the last $b-n$ columns of $\mathbf{Q}$ and keeping only the first m rows. The secret key is only the random b-bit vector $\mathbf{q}_1$ rather than a $m \times n$ matrix $\mathbf{M}$. Let the unknown bits in $\mathbf{q}_1$ be written as $\mathbf{q}_1 = (k_1, k_2, \ldots, k_b)$.

More specifically, we can now see that if $\mathbf{M} = [m_{ij}]_{m\times n}$ then every entry in the matrix $\mathbf{M}$ corresponds to an unknown key bit. As $\mathbf{M}'$ is the first part of $\mathbf{M}$, the same holds for $\mathbf{M}'$.

We can view $\mathbf{M}'$ as a generator matrix that spans a code $\mathcal{C}$. But there are many generator matrices spanning the same code. One particular case is when $\mathbf{M}'$ is transformed to the systematic form, that is $\mathbf{M}'' = [\mathbf{I}\ \mathbf{J}]$, where $\mathbf{I}$ is the $m \times m$ identity matrix and $\mathbf{M}'' = \mathbf{S}\mathbf{M}'$ for some $m \times m$ unknown matrix $\mathbf{S}$. We assume $\mathcal{C} = \{\mathbf{v}\mathbf{M}', \mathbf{v} \in \mathbb{Z}_2^m\} = \{\mathbf{v}[\mathbf{I}\ \mathbf{J}], \mathbf{v} \in \mathbb{Z}_2^m\}$. In this case, entries in $\mathbf{J}$ are linear combinations of the secret key bits. Therefore, we consider all entries in $\mathbf{J}$ as

unknown. There is an assumption here that the fist m columns of $\mathbf{M}'$ are linearly independent, which is adopted.

The key-recovery attack consists of running the aforementioned distinguisher, finding several zero sum vectors, and then deducing $\mathbf{M}'$. We first show how to derive the secret key from such zero sum vectors if we assume that the first m positions are all error free (no double error). Again, a zero sum vector fulfills

$$\sum_{j=1}^{\ell} \mathbf{g}_{i_j} = \sum_{j=1}^{\ell} \mathbf{v}_{i_j}\mathbf{M}' + \sum_{j=1}^{\ell} \mathbf{e}_{i_j} = \sum_{j=1}^{\ell} \mathbf{e}_{i_j}.$$

Therefore, finding a zero sum amounts to knowing the corresponding $\sum_{j=1}^{\ell} \mathbf{e}_{i_j}$. We now consider a single $\mathbf{g}_{i_j}$ vector. Its positions can be split in two parts, namely those for which we know that they are (most likely) error free mod 2 (since the error vector is zero in this position) and those for which we do not have knowledge of, since one of the ℓ involved vectors has an error. Note that the first $t \cdot c$ positions are error free, but there are, on average, c_ω positions where at least one of the eight vectors will have an error.

If the first m positions are all of the error free type, one can write

$$\mathbf{g}_{i_j} = \mathbf{v}_{i_j}[\mathbf{I}\ \mathbf{J}] + \mathbf{e}_{i_j}. \tag{9}$$

We further have roughly $d - t \cdot c - c_\omega$ additional positions to be error free. For each such position, we can form a linear equation. Denote by $\mathbf{J}_q$ the q-th column of $\mathbf{J}$. Assume a position $q > m$ is error free. Then

$$\mathbf{g}_{i_j}(q) = \sum_{i'=1}^{m} \mathbf{v}_{i_j}(i')\mathbf{J}_q(i'). \tag{10}$$

Here $\mathbf{g}_{i_j}(q)$ denotes position q in vector $\mathbf{g}_{i_j}$, etc. Since $\mathbf{g}_{i_j}(q)$ and $\mathbf{v}_{i_j}$ are known, it gives a linear equation in the unknowns of vector $\mathbf{J}_q$. Collecting many such equations will enable us to derive $\mathbf{J}_q$ and eventually, the full $\mathbf{J}$ matrix. However, such approach is not adequate as we have seen in Section 4 that, for interested parameters, double error probability can not be disregarded. Hence, we need to consider a more complicated approach where we try to detect columns with double errors.

Assume we have found N zero sum vectors with the 8-sum distinguisher. Let the first seven vectors in the first zero sum vector $\sum_{j=1}^{\ell} \mathbf{g}_{i_j}$ be denoted $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_7$, the first seven vectors in the next zero sum vector be denoted $\mathbf{g}_8, \mathbf{g}_9, \ldots, \mathbf{g}_{14}$, and so forth. We construct a matrix

$$\mathbf{G} = \left(\mathbf{g}_i\right)_{i=1,\ldots,7N}$$

where $\mathbf{g}_i$ are row vectors.

Now we will examine the columns and the related known error vector for the corresponding zero sum vector. Recall that the first $t \cdot c$ columns are error free mod 2, by virtue of Algorithm I.2. For the remaining columns, if $\sum_{j=1}^{\ell} \mathbf{g}_{i_j}$ is zero in the q-th position for **all** detected zero sum vectors, the corresponding column $\mathbf{G}_q$ is free of 'direct errors' and is kept. If it is not true, then we discard the column. After this process, we have a new matrix $\mathbf{G}'$ of length $t \cdot c + U$, where U is the number of columns kept in the previous step. If $t \cdot c + U > 7N > m$ there will be low-weight codewords in the code spanned by $\mathbf{G}'$. Namely, if $7N > m$ then there are linear combinations of rows that correspond to a zero codeword plus error terms. As we have removed all 'direct errors', the only error contribution in the code must come from double errors.

Each double error will give either a 0 or a 1 as contribution in that position. Assume there are D double errors in the columns in $\mathbf{G}'$, then we expect to find codewords in the code spanned by $\mathbf{G}'$ that have weight around $D/2$. More, because we can form 2^{P-m} different combinations of rows that sum to zero in the underlying code, we have 2^{P-m} low weight codewords of weight around $D/2$. Finally, every column in $\mathbf{G}'$ can be found to contain a double error or not as follows. If this position is zero in all (or almost all) low-weight codewords, there is no double error. Otherwise, we detected a double error in that position. From this information, it is possible to do a full recovery through additional steps. A description of this key recovery attack is given in Appendix, Algorithm I.3.

There arises a problem of finding enough columns free of direct errors. When the length n is small, there will be very few columns free of direct errors and the length of $\mathbf{G}'$ is not enough to have low-weight codewords. Therefore, in the following example, we choose a large n instance of Firekite to illustrate our attack.

5.1 Example of the key recovery attack on Firekite with $n = 16384$

Consider the Firekite parameters choice $m = 216$, $n = 16384$ and $k = 216$. The attack works as follows. First we run the 8-sum distinguisher to obtain zero sum vectors. In this case, we need to have slight more than $m/7$, e.g., 32 zero sum vectors. Applying (5), we choose $c = 60$ generating $N \approx 3.5$ zero sum vectors using the 8-sum distinguisher. Instead of repeating the distinguishing attacks 32 times or, equivalently 2^5 more work, we can instead increase the initial list size to $3 \cdot 2^{61}$ to obtain sufficient low-weight sum ($N \approx 41$), with an affordable complexity of $2^{69.87}$. We denote this cost by $C_{\text{distinguishing}}$.

We now consider a matrix $\mathbf{G}$ of dimension $P = 32 \cdot 7 = 224$ consisting of the $\mathbf{g}_i$ vectors as its rows and we then remove columns with direct associated errors. In a zero sum vector, there are $216 \cdot 8$ errors inserted, so a positon is error free with probability $(1 - 1/(16384 - 61 \cdot 3))^{216 \cdot 8} \approx 0.899$. In our case, we want the position to be error free in all 32 zero sum vectors, which brings the probability to about 0.033. Since $d = 13144$, we can have about 432 columns

error free. We then form the matrix $\mathbf{G}'$ which is of dimension $P = 224$ and length $61 \cdot 3 + 432 = 615$. There will be $2^{224-216} = 2^8$ codewords in the code spanned by $\mathbf{G}'$ with support corresponding to the double errors.

By computing the likelihood of double errors, we find that a column in $\mathbf{G}'$ is error free with probability at most $0.995^{32} = 0.85$. For instance, consider a simple case where there is no non-repeating double error at position j-th of a zero sum. Then the probability is

$$1 - \binom{8}{2} \cdot \left(1 - \left(\frac{16383}{16384}\right)^{216}\right)^2 \cdot \left(\frac{16383}{16384}\right)^{216 \cdot 6} \approx 0.995$$

One can expect $615 \cdot 0.15 \approx 93$ columns to have double errors. In conclusion, the code spanned by $\mathbf{G}'$ will contain 2^8 codewords where the weight is distributed around 47. Finding low-weight codewords in a random binary linear code is a well-known problem that has been studied extensively. One can use ISD algorithms to complete the task. For our example, an improved Stern's ISD algorithm[11] yields the bit-complexity estimate, denoted C_{ISD}, to be $2^{44.6}$, which is small compared to the distinguishing step.

A random linear code with dimension 224 and length 615 will have an expected minimum distance of about 100 according to the Varshamov-Gilbert bound, so the low weight codewords would come from the observation above. Finally, generating say 16 such low weight codewords, we look for the positions where all the 16 of these codewords are zero. This would be the case for more than 500 positions and in this way we have identified 500 columns that are completely error free. Using a selection of them as the information set of the code we can recover remaining parts of the code $\mathbf{M}'$. The total complexity is therefore

$$C = C_{\text{distinguishing}} + C_{\text{ISD}} \approx 2^{69.87} + 16 \cdot 2^{44.6} \approx 2^{69.87}.$$

6 Discussion and Conclusions

Having seen how Firekite is vulnerable to our distinguisher, especially the 8-sum distinguishing attack, it is natural to ask how we can make Firekite and other similar ciphers resilient to a generic birthday problem solving algorithm. From the result and performance of our attacks, there are certain approaches one can consider. First, we observed that N_{random}, or in other words, the failure probability inflates when the filtering weight c_ω grows. That is to say, unless c_ω is very small compared to d, it is difficult to distinguish Firekite's zero sum vectors from those that could stem from random vectors $\mathbf{g}_i$. Therefore, instantiating Firekite with

[11] The estimate is obtained in a recent work by Andre Esser and Emanuele Bellini [EB22a], where they unify ISD-algorithm variants (Prange,Stern,MMT,BJMM) in a *Nearest-Neighbor* framework. They also provided a complexity estimator for independent parameters.

larger k can be beneficial. Second, we have discussed that the attack complexity depends on the parameter c which is solely determined by m (if we fix ℓ), the number of rows in $\mathbf{M}$. Therefore, if the security level is close to $m/(1+\log \ell)$, our attack becomes infeasible. As a contribution to Firekite's design criteria, we propose a few modifications as follows.

For small Firekite's parameters, one can increase k slightly, which yields an LPN instance with a higher noise rate; therefore more difficult to solve in general. In our estimate, larger k suggests a drastic decrease in P_{nf} and an increase in N_{random}. It is now exceedingly unlikely to have no error in the first $t \cdot c$ bits and d becomes smaller, which makes it more difficult to distinguish the zero sum found by Algorithm I.2 from those stemming from the random case. As an example, by setting $k = 24$ for the instance $n = 1024, m = 216$, our attack was rendered useless as N_{random} is always larger than 1. This comes at the cost of decreasing the number of bits encrypted per invocation; hence more instructions need to be executed *per bit*. However, larger parameter instances of Firekite are less affected by this 'fix' as d becomes large relatively to k. For instance, our 8-sum attack still succeeds with $n = 16384, m = 216$ despite raising k from its original $k = 216$ to $k = 400$. We only need to slightly increase $|L^{(0)}| = 3 \cdot 2^{67}$ and we still obtain a good failure probability as $N_{\text{random}} \approx 2^{-2835}$. An extreme adjustment such as $k = 600$ gives Firekite resistance to our attack. We apply this idea to our simulation with toy parameters to verify the countermeasure (see Appendix, Example 3).

Finally, we may discuss possible future improvements to the proposed attack. We believe that there can be a possibility to gain some small amount in terms of decreased complexity by smaller changes in the distinguishing algorithm. One idea could be to not only allow sums of vectors that sum to zero in c positions, but also those that have weight 1 in these c positions. Still, it would not change the complexity significantly, and with modified parameters as suggested above, the Firekite should meet the intended security level. Moreover, since our approach relies heavily on GBA, it would be amenable in principle to the quantum search approach, e.g., using the result in [NS20].

Acknowledgments

The authors are grateful to anonymous reviewers from ToSC 2022 and CRYPTO 2022 for their helpful comments, which improve greatly both the editorial and technical quality of this work.

The work presented in this paper has been partly funded by the Swedish Research Council (Grant No. 2019-04166) and the Swedish Foundation for Strategic Research (Grant No. RIT17-0005).

Appendix

Algorithm I.3: Firekite key recovery algorithm

Input: A list $L^{(0)}$, with $\mathbf{g}_i \in \mathbb{Z}_2^n$, parameters k, m, d, $c_\omega = \lceil \frac{8 \cdot k \cdot d}{n} \rceil, D/2, G = \emptyset, E = \emptyset$.

1 **for** $i = 1, \ldots, 3$ **do**
2 $L^{(i)} = \textsc{Combine}(L^{(i-1)})$;
3 **for** $\mathbf{s}_i = \sum_{j=1}^{8} \mathbf{g}_{i_j}$ **in** $L^{(3)}$ **do**
4 **if** $\mathsf{HammingWt}(\mathbf{s}_i) \leq c_\omega$ **then**
5 $G \leftarrow G \cup \{\mathbf{g}_{i_1}, \ldots, \mathbf{g}_{i_7}\}$;
6 $\mathbf{G} = (\mathbf{g}_j)_{j=1\ldots|G|}$; `// constructing G`
7 **for** $i = 1, \ldots, d$ **do**
8 **if** $\exists \mathbf{s} \in L^{(3)}, \mathbf{s}_i = 1$ **then**
9 $\mathbf{G} \leftarrow \mathbf{G} \setminus \mathbf{G}_i$; `// removing columns with direct error`
10 Finding all vectors $\mathbf{e}_i$ spanned by $\mathbf{G}$ that are of Hamming weight less than $D/2$;
11 Keep columns of $\mathbf{G}$ if $\mathbf{e}_i$ are all 0 in that position;
12 Recover $\mathbf{M}'$ from $\mathbf{G}$;
13 **return** $\mathbf{M}'$;

Example 1. *(Probability of repeating double error.)*

To convince readers that P_{nf} can be reasonably approximated by taking into account only non-repeating double error, we give an example where there is 1 double-error. We consider the probabilities of this event in both cases: the double error comes from only 2 or 4 vectors out of 8. Our parameters are $n = 1024, k = 16, c = 62$ (see Section 4 for the choice of c).

The probability of 4 error vectors having a one in the same position in the first $3 \cdot c$ bits is:

$$P_1 \approx \binom{8}{4} \frac{k^4}{(3c)^3} \left(\frac{3c}{n}\right)^4 \left(\frac{n-3c}{n}\right)^{4(k-1)} \left(\frac{n-3c}{n}\right)^{4k} \approx 2^{-46.19}.$$

The probability of 2 error vectors having a one in the same position in the first $3 \cdot c$ bits is:

$$P_1' \approx \binom{8}{2} \frac{k^2}{3c} \left(\frac{3c}{n}\right)^2 \left(\frac{n-3c}{n}\right)^{2(k-1)} \left(\frac{n-3c}{n}\right)^{6k} \approx 2^{-36.09} \gg P.$$

Table 3: Repeating (P_1) and non-repeating double errors (P_1') probabilities for some parameters of Firekite.

Parameters			Probabilities	
m	n	k	$\log(P_1)$	$\log(P_1')$
216	1024	16	-46.19	-36.09
216	2048	32	-44.95	-34.54
216	16,384	216	-33.37	-27.70

Example 2. *(Multiple double errors)*

Let us compare the probabilities of multiple double errors in the first $t \cdot c$ bits to justify our assumption that P_{nf} can be 'practically' approximated. We continue with Firekite parameters in Example 1, with $c = 76$ (4-sum distinguisher).

The probability of no double error in the first $2 \cdot 76$ bits is:

$$P_0 = \left(\frac{n - 2 \cdot c}{n}\right)^{4k} \approx 2^{-14.83}.$$

The probability of one double error in the first $2 \cdot 76$ bits is:

$$P_1 \approx \binom{4}{2} \frac{k^2}{(2c)} \left(\frac{2c}{n}\right)^2 \left(\frac{n - 2c}{n}\right)^{2(k-1)} \left(\frac{n - 2c}{n}\right)^{2k} \approx 2^{-17.13}.$$

Therefore, $P_0 + P_1 \approx 2^{-14.57}$, and more importantly, this 'better approximation' of P_{nf} does not affect our algorithm significantly.

Another example being P_{nf} in our simulation (Section 4) where $n = 256, m = 52, k = 4$. For the 8-sum distinguisher, the double error probabilities are:

$$P_0 \approx 2^{-8.27},$$

$$P_1 \approx 2^{-9.55},$$

$$P_2 \approx 2^{-11.58}.$$

Example 3. *(Improve Firekite by increasing noise level)*

Recall in our simulation, the parameters are $n = 256, m = 52, k = 4$. We increase k to $k = 7$ and apply the 8-sum distinguisher again with $c = 14$. One can verify, with the new parameters, that

$$N_{random} \approx 3.15.$$

$$N \approx 0.03.$$

As discussed in Section 4, due to non-uniformity of the input, we exhaust all combinations in the last Combine step. On average, there are 2^{18} *vectors in* $L^{(3)}$ *which raises* N_{random} *(in our simulation) to* $\frac{2^{18}}{3 \cdot 2^{14}} \cdot 3.15 \approx 18.9$. *After* 10^2 *tests, we observe on average,* 18.8 *low weight vectors. Therefore our distinguishing attack does not work.*

References

[App+09b] B. Applebaum et al. "Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems". In: *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*. Ed. by S. Halevi. Vol. 5677. Lecture Notes in Computer Science. Springer, 2009, pp. 595–618.

[Bec+12a] A. Becker et al. "Decoding Random Binary Linear Codes in 2 n/20: How 1 + 1 = 0 Improves Information Set Decoding". In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 520–536.

[Ber09] D. J. Bernstein. "Introduction to post-quantum cryptography". In: *Post-quantum cryptography*. Springer, 2009, pp. 1–14.

[BKW03a] A. Blum, A. Kalai, and H. Wasserman. "Noise-tolerant learning, the parity problem, and the statistical query model". In: *J. ACM* 50.4 (2003), pp. 506–519.

[BLP11] D. J. Bernstein, T. Lange, and C. Peters. "Smaller Decoding Exponents: Ball-Collision Decoding". In: *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*. Ed. by P. Rogaway. Vol. 6841. Lecture Notes in Computer Science. Springer, 2011, pp. 743–760.

[Blu+93] A. Blum et al. "Cryptographic Primitives Based on Hard Learning Problems". In: *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*. Ed. by D. R. Stinson. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 278–291.

[BM17b] L. Both and A. May. "The Approximate k-List Problem". In: *IACR Trans. Symmetric Cryptol.* 2017.1 (2017), pp. 380–397.

[BMT78b] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)" In: *IEEE Trans. Inf. Theory* 24.3 (1978), pp. 384–386.

[Bog+21b] S. Bogos et al. "Towards Efficient LPN-Based Symmetric Encryption". In: *Applied Cryptography and Network Security - 19th International Conference, ACNS 2021, Kamakura, Japan, June 21-24, 2021, Proceedings, Part II*. Ed. by K. Sako and N. O. Tippenhauer. Vol. 12727. Lecture Notes in Computer Science. Springer, 2021, pp. 208–230.

[BSH06] J. Barbier, G. Sicot, and S. Houcke. "Algebraic approach for the reconstruction of linear and convolutional error correcting codes". In: *International Journal of Applied Mathematics and Computer Science* 2.3 (2006), pp. 113–118.

[BV15] S. Bogos and S. Vaudenay. "How to Sequentialize Independent Parallel Attacks? - Biased Distributions Have a Phase Transition". In: *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*. Ed. by T. Iwata and J. H. Cheon. Vol. 9453. Lecture Notes in Computer Science. Springer, 2015, pp. 704–731.

[CF09] M. Cluzeau and M. Finiasz. "Recovering a code's length and synchronization from a noisy intercepted bitstream". In: *IEEE International Symposium on Information Theory, ISIT 2009, June 28 - July 3, 2009, Seoul, Korea, Proceedings*. IEEE, 2009, pp. 2737–2741.

[CT19] K. Carrier and J. Tillich. "Identifying an unknown code by partial Gaussian elimination". In: *Des. Codes Cryptogr.* 87.2-3 (2019), pp. 685–713.

[Dod+12] Y. Dodis et al. "Message Authentication, Revisited". In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 355–374.

[DV13b] A. Duc and S. Vaudenay. "HELEN: A Public-Key Cryptosystem Based on the LPN and the Decisional Minimal Distance Problems". In: *Progress in Cryptology - AFRICACRYPT 2013, 6th International Conference on Cryptology in Africa, Cairo, Egypt, June 22-24, 2013. Proceedings*. Ed. by A. M. Youssef, A. Nitaj, and

A. E. Hassanien. Vol. 7918. Lecture Notes in Computer Science. Springer, 2013, pp. 107–126.

[EB22a] A. Esser and E. Bellini. "Syndrome Decoding Estimator". In: *Public-Key Cryptography – PKC 2022*. Ed. by G. Hanaoka, J. Shikata, and Y. Watanabe. Cham: Springer International Publishing, 2022, pp. 112–141.

[GJL14] Q. Guo, T. Johansson, and C. Löndahl. "Solving LPN Using Covering Codes". In: *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*. Ed. by P. Sarkar and T. Iwata. Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 1–20.

[GRS08b] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "HB$^{\#}$: Increasing the Security and Efficiency of HB^{+}". In: *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*. Ed. by N. P. Smart. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[GRS08d] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "How to Encrypt with the LPN Problem". In: *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*. Ed. by L. Aceto et al. Vol. 5126. Lecture Notes in Computer Science. Springer, 2008, pp. 679–690.

[HB01b] N. J. Hopper and M. Blum. "Secure Human Identification Protocols". In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. Ed. by C. Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 52–66.

[Hen+12] N. Heninger et al. "Mining your Ps and Qs: Detection of widespread weak keys in network devices". In: *21st {USENIX} Security Symposium ({USENIX} Security 12)*. 2012, pp. 205–220.

[Hey+07] S. Heyse et al. "An efficient authentication protocol based on Ring-LPN". In: *ECRYPT Workshop on Lightweight Cryptography*. Vol. 2011. Citeseer. 2007.

[Hey+12] S. Heyse et al. “Lapin: An Efficient Authentication Protocol Based on Ring-LPN”. In: *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*. Ed. by A. Canteaut. Vol. 7549. Lecture Notes in Computer Science. Springer, 2012, pp. 346–365.

[HJB09] M. Hell, T. Johansson, and L. Brynielsson. “An overview of distinguishing attacks on stream ciphers”. In: *Cryptogr. Commun.* 1.1 (2009), pp. 71–94.

[IL90] R. Impagliazzo and L. A. Levin. “No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random”. In: *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*. IEEE Computer Society, 1990, pp. 812–821.

[JW05b] A. Juels and S. A. Weis. “Authenticating Pervasive Devices with Human Protocols”. In: *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*. Ed. by V. Shoup. Vol. 3621. Lecture Notes in Computer Science. Springer, 2005, pp. 293–308.

[Kil+17] E. Kiltz et al. “Efficient Authentication from Hard Learning Problems”. In: *J. Cryptol.* 30.4 (2017), pp. 1238–1275.

[Kir11a] P. Kirchner. “Improved Generalized Birthday Attack”. In: *IACR Cryptol. ePrint Arch.* (2011), p. 377.

[KS06a] J. Katz and J. S. Shin. “Parallel and Concurrent Security of the HB and HB^+ Protocols”. In: *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*. Ed. by S. Vaudenay. Vol. 4004. Lecture Notes in Computer Science. Springer, 2006, pp. 73–87.

[LB88] P. J. Lee and E. F. Brickell. “An Observation on the Security of McEliece’s Public-Key Cryptosystem”. In: *Advances in Cryptology - EUROCRYPT ’88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*. Ed. by C. G. Günther. Vol. 330. Lecture Notes in Computer Science. Springer, 1988, pp. 275–280.

[Leo88] J. S. Leon. “A probabilistic algorithm for computing minimum weights of large error-correcting codes”. In: *IEEE Trans. Inf. Theory* 34.5 (1988), pp. 1354–1359.

[LF06a] É. Levieil and P. Fouque. "An Improved LPN Algorithm". In: *Security and Cryptography for Networks, 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006, Proceedings*. Ed. by R. D. Prisco and M. Yung. Vol. 4116. Lecture Notes in Computer Science. Springer, 2006, pp. 348–359.

[Mår18] E. Mårtensson. "Poster: Coded-BKW with Sieving". In: *Training School on Cryptanalysis of Ubiquitous Computing Systems*. Ponta Delgada, Azores, Portugal, Apr. 2018.

[MGB12] M. Marazin, R. Gautier, and G. Burel. "Algebraic method for blind recovery of punctured convolutional encoders from an erroneous bitstream". In: *IET Signal Process.* 6.2 (2012), pp. 122–131.

[NS20] M. Naya-Plasencia and A. Schrottenloher. "Optimal Merging in Quantum k-xor and k-sum Algorithms". In: *Advances in Cryptology – EUROCRYPT 2020*. Ed. by A. Canteaut and Y. Ishai. Cham: Springer International Publishing, 2020, pp. 311–340.

[Pra62] E. Prange. "The use of information sets in decoding cyclic codes". In: *IRE Trans. Inf. Theory* 8.5 (1962), pp. 5–9.

[SHB09] G. Sicot, S. Houcke, and J. Barbier. "Blind detection of interleaver parameters". In: *Signal Process.* 89.4 (2009), pp. 450–462.

[Sho99] P. W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Rev.* 41.2 (1999), pp. 303–332.

[Ste93] J. Stern. "A New Identification Scheme Based on Syndrome Decoding". In: *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*. Ed. by D. R. Stinson. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 13–21.

[Tix15] A. Tixier. "Blind identification of an unknown interleaved convolutional code". In: *IEEE International Symposium on Information Theory, ISIT 2015, Hong Kong, China, June 14-19, 2015*. IEEE, 2015, pp. 71–75.

[Wag02] D. A. Wagner. "A Generalized Birthday Problem". In: *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*. Ed. by M. Yung. Vol. 2442. Lecture Notes in Computer Science. Springer, 2002, pp. 288–303.

[YZ16] Y. Yu and J. Zhang. “Cryptography with Auxiliary Input and Trapdoor from Constant-Noise LPN”. In: *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*. Ed. by M. Robshaw and J. Katz. Vol. 9814. Lecture Notes in Computer Science. Springer, 2016, pp. 214–243.

Differential cryptanalysis of Mod-2/Mod-3 constructions of binary weak PRFs

Abstract

Pseudo-random functions are a fundamental building block in many cryptographic applications. In certain scenarios, a weaker notion (where security is restricted to uniformly random input), but more computationally efficient, called weak pseudo-random functions, is sufficient. In this work, we present new differential attacks on the main binary weak pseudo-random function constructions, namely the so-called Alternative Mod-2/Mod-3. For the Alternative Mod-2/Mod-3 wPRF, the best distinguisher proposed by Cheon et al. achieves $O(2^{0.21n})$ complexity, where n is the input length. We show that our attack asymptotically outperforms this and requires far fewer samples that can be applied in restricted oracle settings. By minimizing computational complexity, we can achieve $O(2^{0.166n})$ complexity. Additionally, in a small experiment, we indicate that their proposed fix of using keys with large Hamming weight is even more vulnerable to our attack.

1 Introduction

A pseudo-random function (PRF), first conceptualized in 1984 by Goldreich et al. [GGM84], is a keyed function that behaves in all aspects like a true random function. PRFs are widely used as building blocks to construct different cryptographic

Thomas Johansson, Willi Meier, Vu Nguyen. "Differential cryptanalysis of Mod-2/Mod-3 constructions of binary weak PRFs". In *International Symposium on Information Theory 2023, Taipei, Taiwan*. 2023 IEEE International Symposium on Information Theory (ISIT), pp. 477-482, IEEE.

schemes. Examples of such schemes are message authentication, keyed hash functions, digital signatures, and indistinguishability obfuscation, [App14; Ana+15; Bel15; BCK96; BR17b; Gol86]. The standard (or strong) pseudo-random function definition enjoys a strong security requirement: not only does it have to be resistant to all classes of distinguishers, but also secure against adaptive attacks, i.e., adversaries can observe the input-output of their choices. In certain settings, the full power of PRF is not required, and it is natural to restrict adversaries to perform only on random input. This motivates non-adaptive or weak PRF. For a thorough survey, we refer the readers to [BR17b].

The notion of weak PRFs (wPRF), despite satisfying a weaker security definition, achieves higher efficiency than PRFs and opens the door to many applications. A wPRF is, briefly explained, a keyed function that is indistinguishable from a random function when the adversary is limited to only observing outputs that come from randomly sampled inputs.

Many cryptographic primitives and applications are built from weak PRFs because of their efficiency [AA17; Bal+20b; Dod+12; DN02; LM13b; MS07; Pie09]. In [Din+21], new candidates for symmetric cryptographic primitives are proposed that use the idea of alternation between linear functions modulo 2 and modulo 3 to support fast protocols for secure multiparty computation (MPC). Their motivations are to construct cryptographic primitives that have concrete efficiency, are easy to describe, and are well-suited for different applications. This continues the study of weak pseudo-random functions of this kind initiated by Boneh et al., [Bon+18], and Cheon et al. [Che+22]. Notably, in [Bon+18], the author proposed a new kind of primitive, called 'encoded-input' PRF, constructed from their weak PRFs, as a potential replacement for strong PRFs in a scenario where the latter are not known to exist.

In [Bon+18], two types of weak PRF candidates are introduced, coined basic Mod-2/Mod-3 and alternative Mod2/Mod-3 weak PRF. Both use a mixture of linear computations defined on different small moduli to satisfy conceptual simplicity, low complexity, and MPC friendliness. These wPRFs candidates were conjectured to be exponentially secure against any adversary that allows exponentially many samples. In [Che+22], these wPRFs are investigated regarding attacks and necessary fixes.

In particular, for the alternative Mod-2/Mod-3 wPRF, it is proved that the adversary's advantage is at least $2^{-0.105n}$, where n is the size of the input space of the weak PRF. This gives distinguisher and key-recovery attacks of asymptotic complexity $O(2^{0.21n})$. It is worth noting that, as pointed out in [Din+21; Che+22], the attack in [Che+22] requires a large number of evaluations from the wPRF. Further, a simple method of using keys with large Hamming weight is suggested for repairing the wPRFs affected by these attacks while preserving the parameters.

It appears, therefore, of interest, to increase the understanding of constructions based on mixing linear computations defined on different small moduli.

1.1 Contributions

The paper aims to describe a differential attack on the binary Alternative Mod-2/Mod-3 wPRF in [Bon+18]. Rather than the attack in [Che+22], our attack investigates and exploits reduced weight input differences. Our attack significantly outperforms the one in [Che+22] regarding data complexity. Minimizing computational complexity, we achieve $O(2^{0.166n})$ complexity, an asymptotic improvement compared to [Che+22]. Moreover, it is demonstrated that the fix does not achieve the desired security level as proposed in [Che+22].

2 Preliminaries

Throughout the paper, we denoted by

- bold uppercase letters, e.g., $\mathbf{A}$, matrices and bold lowercase letters, e.g., $\mathbf{x}$, row vectors.
- $\mathbf{A}_i$ and $\mathbf{A}^j$ the i-th row and the j-th columns of $\mathbf{A}$, respectively.
- $\mathbf{x}$ mod p, for p prime, as the modulo operation by each element of $\mathbf{x}$.
- $\mathbf{x} + \mathbf{y}$ the bit-by-bit XOR of vectors $\mathbf{x}$ and $\mathbf{y}$.
- $\omega_H(\mathbf{x})$ the Hamming weight of a binary vector $\mathbf{x}$.
- log the logarithm base 2 and $H(p) = -(p \log p + (1-p)\log(1-p))$ the binary entropy function with $p \in [0, 1]$.
- $[i] = 1, \ldots, i$ for an integer $i \in \mathbb{N}$.
- $\{0,1\}^n$ binary vector of length n.
- Funs$[\mathcal{X}, \mathcal{Y}]$ by the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$.

Definition 1 (Pseudo random functions (PRF) in [Bon+18]). *Let $\lambda \in \mathbb{N}$ be a security parameter, and $\mathcal{K} = \{\mathcal{K}_\lambda\}_{\lambda\in\mathbb{N}}$,$\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$, $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ be ensembles of finite sets indexed by λ. Let $\{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be an efficiently-computable family of functions $F_\lambda : \mathcal{K}_\lambda \times \mathcal{X}_\lambda \to \mathcal{Y}_\lambda$. The function family $\{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ is said to be a (t, ϵ)-strong pseudo-random function if for all adversaries $\mathcal{A}$ running in time $t(\lambda)$, and taking $k \in \mathcal{K}_\lambda$, $f_\lambda \in$ Funs$[\mathcal{X}_\lambda, \mathcal{Y}_\lambda]$ uniformly at random, we have that*

$$\left| \Pr\left[\mathcal{A}^{F_\lambda(k,\cdot)}(1^\lambda) = 1\right] - \Pr\left[\mathcal{A}^{f_\lambda(\cdot)}(1^\lambda) = 1\right] \right| \leq \epsilon(\lambda).$$

A weaker notion of PRF is the weak PRF: the family $\{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ is said to be an (ℓ, t, ϵ)-weak pseudo-random function if for all adversaries $\mathcal{A}$ running in time $t(\lambda)$, taking

$k \in \mathcal{K}_\lambda, f_\lambda \in \text{Funs}[\mathcal{X}_\lambda, \mathcal{Y}_\lambda]$, and $x_1, \ldots, x_\ell \in \mathcal{X}_\lambda$ uniformly at random, we have

$$\left| \Pr\left[\mathcal{A}\left(1^\lambda, \{x_i, F_\lambda(k, x_i)\}_{i\in[\ell]}\right)\right] - \Pr\left[\mathcal{A}\left(1^\lambda, \{x_i, f_\lambda(k, x_i)\}_{i\in[\ell]}\right)\right] \right| \leq \epsilon(\lambda).$$

Informally speaking, for a family of functions to be strong (resp. weak) pseudorandom, an adversary can not distinguish them from a uniformly random function when observing adaptively chosen (resp. uniformly random) pairs of input-output. We briefly visit the binary weak PRF constructions that are of interest to our work.

Construction 1 (Alternative Mod-2/Mod-3 candidate in [Bon+18]). *Let λ be a security parameter, and $n = n(\lambda)$ be the key and input length. The weak PRF candidate is a family of functions $\mathcal{F}_\lambda : \mathbb{Z}_2^n \times \mathbb{Z}_2^n \to \mathbb{Z}_2$ with key-space $\mathcal{K}_\lambda = \mathbb{Z}_2^n$, input space $\mathcal{X}_\lambda = \mathbb{Z}_2^n$ and output space $\mathcal{Y}_\lambda = \mathbb{Z}_2$. Let $\mathbf{k} = (k_1, \ldots, k_n) \in \mathcal{K}_\lambda$ and an input $\mathbf{x} = (x_1, \cdots, x_n)$, a function $F_\lambda(\mathbf{k}, \mathbf{x})$, denoted by $F_\mathbf{k}(\mathbf{x})$, in $\mathcal{F}_\lambda$ is defined as*

$$F_\mathbf{k}(\mathbf{x}) = \sum_{i\in[n]} k_i x_i \bmod 2 + \sum_{i\in[n]} k_i x_i \bmod 3 \bmod 2.$$

The construction can be viewed as a deterministic *Learning Parity with Noise* (LPN) instance with noise rate $\frac{1}{3}$. Indeed, $k_i x_i \bmod 3 = 1$ with probability $\frac{1}{3}$ [Bon+18]. However, in this construction, the noise is correlated to both the input and the secret key rather than being independently sampled. The authors pointed out that although the BKW attack applies to this candidate, it requires a large number of samples and does not pose a major threat in practice [Bon+18]. The motivation behind the construction is that it is very efficient to compute in different MPC settings. For background in the LPN problem and its state-of-the-art attack, we refer the readers to [GJL14].

In a recent work [Che+22], Cheon et al. analyzed the security of several weak PRF candidates, including the Alternative Mod-2/Mod-3. They showed a distinguishing attack for the candidate in time $O(2^{0.21n})$. As a fix for the candidates, they proposed raising the Hamming weight of $\mathbf{k}$ as the complexity of their attack depends on $\omega_H(\mathbf{k})$ (Theorem 1, [Che+22]).

The idea of our attack for this wPRF is to find reduced weight input differences. In brevity, we look for pairs $(\mathbf{x}, \mathbf{x}')$ where $\mathbf{x} + \mathbf{x}'$ has 'small' weight. We then derive the particular bias arising from such pairs. Since our attack relies on finding pairs of vectors with low Hamming weight in the bit-wise XOR sum, any dedicated algorithm benefits our attack. In particular, our problem can be solved through the *Nearest-Neighbor Problem*, which we briefly introduce.

Definition 2 (Nearest Neighbor (NN) Problem in [MO15]). *Let $n \in \mathbb{N}, 0 < \alpha < 1/2$, and $0 < \lambda < 1$. In the (n, α, λ)-NN problem, we are given two lists $\mathcal{L}, \mathcal{R}$ of equal size $2^{\lambda n}$ with uniform and pairwise independent vectors. If there exists a pair $(\mathbf{x}, \mathbf{x}') \in \mathcal{L} \times \mathcal{R}$ with Hamming distance $\omega_H(\mathbf{x}, \mathbf{x}') = \alpha n$, we are asked to output a list that contains $(\mathbf{x}, \mathbf{x}')$.*

Examining this problem, there have been numerous studies [IM98; Dub10; MO15; EKZ21], to name a few. In 2015, May and Ozerov [MO15] proposed their algorithm, called NearestNeighbor. This has led to many improvements in cryptanalysis, particularly in the *Information-Set Decoding* algorithms towards code-based cryptography [MO15; BM18; EB22b]. The following theorem is helpful to our work.

Theorem 1 (Theorem 1, [MO15]). *For any constant $\epsilon > 0$ and any $\lambda < 1 - H\left(\frac{\alpha}{2}\right)$,* NearestNeighbor *solves the (n, α, λ)-NN problem with overwhelming probability in time*

$$O\left(2^{(y+\epsilon)n}\right), \text{with } y := (1-\alpha)\left(1 - H\left(\frac{H^{-1}(1-\lambda) - \frac{\alpha}{2}}{1-\alpha}\right)\right).$$

For details of the algorithm, we refer the readers to the original work [MO15].

To provide stronger candidate constructions that can work on a smaller input space, a set of different primitives was proposed in [Din+21], among them a binary weak PRF that was named LPN-wPRF.

Construction 2 (LPN-wPRF candidate in [Din+21]). *Let λ be the security parameter, m, n, t be functions of λ, and $\mathbf{B} \in \mathbb{Z}_2^{t \times m}$ be fixed public matrix chosen uniformly at random. The LPN-wPRF candidate is a family of functions $\mathcal{F}_\lambda : \mathbb{Z}_2^{m \times n} \times \mathbb{Z}_2^n \to \mathbb{Z}_2^t$ with key-space $\mathcal{K}_\lambda = \mathbb{Z}_2^{m \times n}$, input space $\mathcal{X}_\lambda = \mathbb{Z}_2^n$ and output space $\mathcal{Y}_\lambda = \mathbb{Z}_2^t$. Let $\mathbf{K} \in \mathcal{K}_\lambda$, a function $F_\mathbf{K}$ in $\mathcal{F}_\lambda$ is defined as*

- *For input $\mathbf{x} \in \mathbb{Z}_2^n$, compute*

$$\mathbf{u} = \mathbf{K}\mathbf{x}^T \bmod 2.$$

- *The noise is added by interpreting $\mathbf{K}$ and $\mathbf{x}$ over $\mathbb{Z}_3$. In particular, we compute*

$$\mathbf{v} = \left(\mathbf{K}\mathbf{x}^T \bmod 3\right) \bmod 2,$$

and output

$$\mathbf{y} = \mathbf{B}(\mathbf{u} + \mathbf{v})^T.$$

To ease the notation, from now on, we omit the transpose symbol as it should be clear from the context. A private key $\mathbf{K}$ in the above construction has the size mn, which can be expensive to communicate within distributed protocols. Hence, the authors in [Din+21] proposed *structured keys*, i.e., those that have a concise presentation (linear in n and m). In particular, *Toeplitz matrices* and *generalized circulant matrices* were introduced.

3 Attack on the Alternative Mod-2/Mod-3 wPRF

3.1 Description of the attack

In this section, we describe another attack on the Construction 1. In particular, we still consider the key to be chosen uniformly at random as opposed to Cheon et al.'s suggestion to increase the Hamming weight (for instance, $\omega_H(\mathbf{k}) = 310$ when $n = 384$). We shall look into the attack performance later when applied to keys with unusually large Hamming weights. Our idea is to distinguish between $k_i = 0$ and $k_i = 1$. Without loss of generality, suppose we are looking at the first key bit k_1.

Assume we can ask for the wPRF evaluations until we get pairs of input $(\mathbf{x}, \mathbf{x}')$ such that $\omega_H(\hat{\mathbf{x}}, \hat{\mathbf{x}}') = \alpha n$, where $\hat{\mathbf{x}} = (x_2, \ldots, x_n)$ (correspondingly $\hat{\mathbf{x}}'$) is the shortened vector $\mathbf{x}$ without the first position. Moreover, α is an important parameter to be specified later. In addition, we are only interested in pairs that fulfill the following:

$$|I := \{i : x_i = 0, x'_i = 1\}| = |J := \{i : x_i = 1, x'_i = 0\}| = \frac{\alpha n}{2}. \tag{1}$$

In other words, we are looking for 'small'-weight sums whose Hamming weights are evenly contributed from the corresponding vectors. Let us consider the probability

$$\Pr[x_1 = x'_1, F_{\mathbf{k}}(\mathbf{x}) = F_{\mathbf{k}}(\mathbf{x}') = 0]. \tag{2}$$

If $k_1 = 0$, then the first bits of both inputs do not contribute to the evaluation of the wPRF. Hence $k_1 = 0$ yields $\Pr[x_1 = x'_1, F_{\mathbf{k}}(\mathbf{x}) = F_{\mathbf{k}}(\mathbf{x}') = 0] = \frac{1}{8}$, since we assume the inputs are drawn uniformly at random. We can rewrite the equation (2) as

$$\Pr[x_1 = x'_1, F_{\mathbf{k}}(\mathbf{x}) = 0] \cdot \Pr[F_{\mathbf{k}}(\mathbf{x}') = 0 | x_1 = x'_1, F_{\mathbf{k}}(\mathbf{x}) = 0]. \tag{3}$$

For an input $\mathbf{x}$, let us denote $G(\mathbf{k}, \mathbf{x}) = G_{\mathbf{k}}(\mathbf{x}) = \sum_{i=1}^{n} k_i x_i \bmod 6$. We recall that from the definition of the wPRF, $F_{\mathbf{k}}(\mathbf{x}) = 0$ if and only if $G_{\mathbf{k}}(\mathbf{x}) \in \{0, 1, 2\} \bmod 6$. Besides the set of indices I and J in equation (1), we further define $K := \{i : x_i = x'_i = 1\}$ and three sums

$$A = \sum_{i \in K} k_i x_i \bmod 6 = \sum_{i \in K} k_i x'_i \bmod 6,$$

$$X = \sum_{i \in I} k_i x_i \bmod 6, \quad X' = \sum_{i \in J} k_i x'_i \bmod 6.$$

To ease the notation, from now on we omit the mod 6, unless otherwise mentioned. We can rewrite $G_{\mathbf{k}}(\mathbf{x}) = A + X$ and $G_{\mathbf{k}}(\mathbf{x}') = (A + X) + (X' - X) = G_{\mathbf{k}}(\mathbf{x}) + X - X'$. With the assumption that $\Pr[G_{\mathbf{k}}(\mathbf{x}) = a | G_{\mathbf{k}}(\mathbf{x}) \in \{0, 1, 2\}] =$

$\frac{1}{3}$, $a = 0, 1, 2$, the second term in equation (3) becomes

$$\begin{aligned}&\Pr[G_{\mathbf{k}}(\mathbf{x}) + (X' - X) \in \{0,1,2\} | x_1 = x_1', G_{\mathbf{k}}(\mathbf{x}) \in \{0,1,2\}] \\ =&\frac{1}{3}\sum_{a=0}^{2} \Pr[X' - X \in \{-a, 1-a, 2-a\} | G_{\mathbf{k}}(\mathbf{x}) = a].\end{aligned}$$

By definition, the first equality follows from $(X' - X)$ is independent of the event $x_1 = x_1'$. We claim the following theorem.

Theorem 2. *Let* $\mathbf{k} = (k_1, \ldots, k_n)$, $\mathbf{x} = (x_1, \ldots, x_n)$, *and* $\mathbf{x}' = (x_1', \ldots, x_n')$ *be length-*n *binary vectors, such that* $\omega_H(\mathbf{x} + \mathbf{x}') = \alpha n = 2\ell$ *and* $|I| = |J| = \ell$, *where* $I := \{i : x_i = 0, x_i' = 1\}$, *and* $J := \{i : x_i = 1, x_i' = 0\}$. *Assume that* $\mathbf{k}$ *is uniformly random. Then the followings hold*

$$\sum_{a=0}^{2} \Pr[X' - X \in \{-a, 1-a, 2-a\} | G_{\mathbf{k}}(\mathbf{x}) = a] = \frac{3}{2} + \left(\frac{3}{4}\right)^{l-1}, \tag{4}$$

$$\left| \Pr[F_{\mathbf{k}}(\mathbf{x}') = 0 | x_1 = x_1', F_{\mathbf{k}}(\mathbf{x}) = 0] - \frac{1}{2} \right| \approx \frac{1}{2^{0.41\ell + 1.17}}. \tag{5}$$

Proof. To prove Theorem 2, we make use of Lemma 4.1 in Cheon et al.'s work [Che+22].

Lemma 4. *Let* n *be a positive integer. For all* $0 \leq a \leq 5$, *the following equations hold*

$$\sum_{a+6k \leq n} \binom{n}{a+6k} = \frac{1}{6}\left(\sum_{j=0} \left(\omega^j\right)^{6-a} \cdot \left(1 + \omega^j\right)^n\right),$$

where $\omega = \dfrac{1 + \sqrt{3}i}{2}$ *is the* 6*-th root of unity.*

The proof consists of long and repetitive computations; hence, we will only prove for the case $\ell = 6k$, $k \geq 0$ is odd. Recall that $X = \sum_{i \in I} k_i x_i = \sum_{i \in I} k_i$ where $|I| = \ell$. Therefore, for $a = 0, \ldots 5$,

$$\Pr[X = a] = \frac{\sum_{6k+a \leq \ell} \binom{\ell}{a+6k}}{2^\ell}.$$

Using Lemma 4, we obtain

$$\begin{aligned}\Pr[X = a] &= \frac{\sum_{j=0}^{5} \left(\omega^j\right)^{6-a} \cdot \left(1 + \omega^j\right)^\ell}{6 \cdot 2^\ell} \\ &= \frac{1}{6} + \frac{\sum_{j=1}^{5} \left(\omega^j\right)^{6-a} \cdot \left(1 + \omega^j\right)^\ell}{6 \cdot 2^\ell}\end{aligned}$$

$$= \frac{1}{6} + \epsilon_a.$$

We also note that, since $\sum_{a=0}^{5} \Pr[X - a] = 1$, then $\sum_{a=0}^{5} \epsilon_a = 0$. We have

$$\begin{aligned}\Pr[X' - X = a] &= \sum_{m=0}^{5} \Pr[X' = m + a]\Pr[X' = m] \\ &= \sum_{m=0}^{5} (1/6 + \epsilon_{m+a})(1/6 + \epsilon_m) \\ &= 1/6 + \sum_{m=0}^{5} \epsilon_{m+a}\epsilon_m.\end{aligned}$$

To compute ϵ_a, $a = 0, \ldots 5$, we use the following identities

$$1 + \omega = \omega^5\sqrt{3}i, \quad 1 + \omega^2 = \omega,$$

$$1 + \omega^3 = 0, \quad 1 + \omega^4 = \omega^5,$$

$$1 + \omega^5 = -\omega\sqrt{3}i.$$

Then, from $\ell = 6k$, k is odd, we derive the followings

$$6 \cdot 2^l \cdot \epsilon_0 = 2 - 2(\sqrt{3})^l, \quad 6 \cdot 2^l \cdot \epsilon_1 = -\sqrt{3}^l - 1$$

$$6 \cdot 2^l \cdot \epsilon_2 = \sqrt{3}^l - 1, \quad 6 \cdot 2^l \cdot \epsilon_3 = 2\sqrt{3}^l + 2$$

$$6 \cdot 2^l \cdot \epsilon_4 = \sqrt{3}^l - 1, \quad 6 \cdot 2^l \cdot \epsilon_5 = -\sqrt{3}^l - 1.$$

Now, we compute $\Pr[X - X' = a], a = 0, \ldots, 5$.

$$\Pr\left[X - X' = i\right] = \begin{cases} \frac{1}{6} + \frac{1 + 3^l}{3 \cdot 4^l}, i = 0. \\ \frac{1}{6} + \frac{-1 + 3^l}{6 \cdot 4^l}, i = 1. \\ \frac{1}{6} + \frac{-1 - 3^l}{6 \cdot 4^l}, i = 2. \\ \frac{1}{6} + \frac{1 - 3^l}{3 \cdot 4^l}, i = 3. \\ \frac{1}{6} + \frac{-1 - 3^l}{6 \cdot 4^l}, i = 4. \\ \frac{1}{6} + \frac{-1 + 3^l}{6 \cdot 4^l}, i = 5. \end{cases}$$

Putting everything together, we conclude Theorem 2. □

On the one hand, Theorem 2 provides us with a distinguishing attack for the Alternative Mod-2/Mod-3 wPRF. In details, one can collect $O(2^{2\cdot(0.41\ell+1.17)})$ 'particular' (as described in Theorem 2) pairs of $(\mathbf{x}, \mathbf{x}')$ such that $F_{\mathbf{k}}(\mathbf{x}) = 0$. Then by computing the deviation of $\Pr[F_{\mathbf{k}}(\mathbf{x}') = 0]$ from $\frac{1}{2}$, one can conclude if the evaluations were done using the aforementioned wPRF.

On the other hand, Equation (2) helps us conceive a key recovery attack in the same manner.

To compare with Cheon et al.'s work [Che+22], our distinguishing attack relies on the assumption that $\mathbf{k}$ was chosen at uniformly random and the later-to-be-specified parameter ℓ, rather than $\omega_H(\mathbf{k})$. Moreover, we stress that our distinguisher complexity is not $O(2^{2\cdot(0.41\ell+1.17)})$. We recall that the attack requires producing desired pairs $(\mathbf{x}, \mathbf{x}')$ before estimating the deviation. We will discuss how we choose our parameters α and ℓ, then derive the attack complexity in the next section.

3.2 Different moduli

In [Che+22], Cheon et al. generalized their attacks into Alternative Mod-p/Mod-q wPRF. They showed that the adversary's advantage, in this scenario, is larger than $c_h \cdot \left|\frac{\omega_{pq}+1}{2}\right|^h$.[1] Our approach to finding the bias is similar to the original work. Therefore, in principle, we can also extend to different moduli scenarios.

3.3 Heuristic arguments for the attack complexity

Data complexity

Recall that, for the distinguishing attack, one needs $O(2^{2\cdot(0.41\ell+1.17)})$ pairs of $(\mathbf{x}, \mathbf{x}')$. Suppose we query the wPRF to evaluate $N = 2^c$ evaluations, then we can form up to 2^{2c-1} pairs. Among such pairs, we only look at pairs where Hamming weight of the difference is αn. Since the inputs are assumed to be uniformly random, so is their bit-wise XOR. The probability of a random vector having weight αn is

$$\Pr\left[\omega_H\left(\mathbf{x}+\mathbf{x}'\right) = \alpha n\right] = \frac{\binom{n}{\alpha n}}{2^n}. \tag{6}$$

In addition, among pairs whose Hamming weight is αn, we only examine those that fulfill $|I| = |J| = \frac{\alpha n}{2}$. Heuristically, this probability is

$$\Pr\left[|I| = |J| = \frac{\alpha n}{2}\right] = \frac{\binom{\alpha n}{\alpha n/2}}{2^{\alpha n}}. \tag{7}$$

[1] where $h = \omega_H(\mathbf{k})$ and ω_{pq} is the pq-th root of unity.

In conclusion, we need to evaluate $N = 2^c$ samples, such that the following holds

$$2^{2c-1} \cdot \frac{\binom{n}{\alpha n}}{2^n} \cdot \frac{\binom{\alpha n}{\alpha n/2}}{2^{\alpha n}} \approx 2^{0.41\alpha n + 2.34},$$

or, in other words, the attack's data complexity is 2^c, where

$$c \approx \frac{n\,(1.41\alpha + 1) + 3.34 - \log\binom{\alpha n}{\alpha n/2} - \log\binom{n}{\alpha n}}{2}. \tag{8}$$

Asymptotic computational complexity of our attack

As opposed to Cheon et al.'s work, our attack requires an additional processing phase after the wPRF's evaluation. Suppose, for parameter α, that we observe 2^c evaluations, where $\lambda = c/n$ satisfies the prerequisite in Theorem 1, i.e., $\lambda < 1 - H(\frac{\alpha}{2})$. The cost of finding low-weight sums is then $O\left(2^{(y+\epsilon)n}\right)$, where y is defined as in Theorem 1. Note that the NearestNeighbor allows one to find low-weight sums with a smaller size of input lists. In our case, we have a somewhat fixed size since we need to observe the calculated bias in the previous section.

After the processing phase, we have a list of size $O(2^{2\cdot(0.41\ell+1.17)})$. We then go through the list to find $\left|\Pr[F_{\mathbf{k}}(\mathbf{x}') = 0 | x_1 = x_1', F_{\mathbf{k}}(\mathbf{x}) = 0] - \frac{1}{2}\right|$. Hence, the complexity of this step is $O(2^{0.41\alpha n + 2.34})$. Let us denote by $C_{\text{Mod-2/Mod-3}}$ the computational complexity of our attack, then

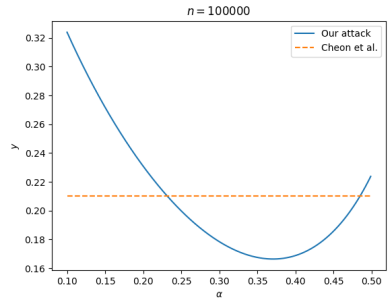
$$C_{\text{Mod-2/Mod-3}} = O\left(2^{(y+\epsilon)n}\right) + O(2^{0.41\alpha n + 2.34}) \approx O\left(2^{(y+\epsilon)n}\right). \tag{9}$$

We note that by fixing n, we can write the complexity exponent y as a function of only α. Therefore, it is important to understand how large y becomes for $\alpha \in (0, \frac{1}{2})$, compared to 0.21 (as in Cheon et al.'s attack). To this end, we rely on simulations with increasingly large input length n to observe the behavior of y (Figure 1a).[2] We see that our attack achieves better asymptotic complexity for $\alpha \in (0.23 + \epsilon(n), 0.49 - \epsilon(n))$, with $\epsilon(n)$ decreases in n. Notably, the smallest value of y is roughly 0.1664. Figure 1b shows the asymptotic data complexity of our attack as the ratio c/n, using Equation (8).

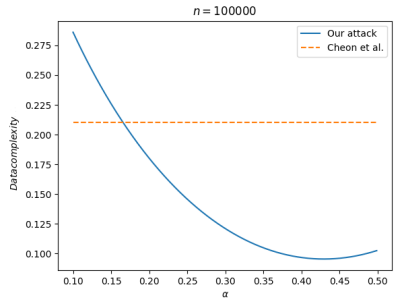
Example 1. *The (conservative) parameter of the Alternative Mod-2/Mod-3 wPRF to achieve 128-bit security is $n = 384$. The optimal parameter for our attack is $\alpha \approx 0.37$, which yields $c \approx 44.5$ and $C_{\text{Mod-2/Mod-3}} \approx 74.27$, if we ignore the polynomial overhead in (9). We show our attack in comparison with the one in [Che+22].*

While it remains unclear how the NearestNeighbor polynomial overhead, which has been addressed in [EKZ21; MO15], will affect our attack for concrete

[2] Since our attack heavily depends on the NearestNeighbor search, we exclude the region of α where the condition $\lambda < 1 - H(\frac{\alpha}{2})$ does not hold.

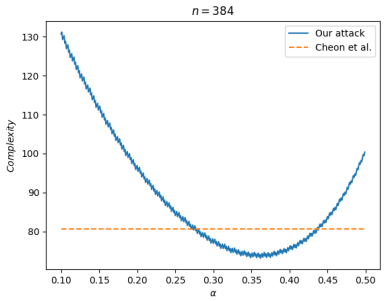


(a) The exponent $y = y(\alpha)$.

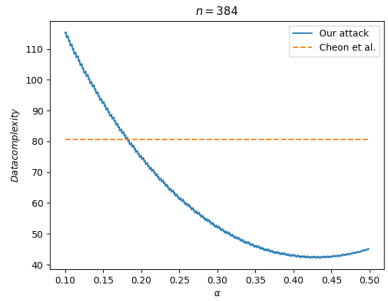


(b) The ratio $\lambda(\alpha) = c(\alpha)/n$.

Figure 1: The data and computational complexity for $n = 10^5$.



(a) The computational complexity.



(b) The data complexity.

Figure 2: The data and computational complexity for $n = 384$.

parameters, we can see that our distinguisher is advantageous. In particular, it requires far fewer samples/evaluations, compared to [Che+22], and is suitable in certain restricted oracle settings where the amount of queries to the oracle is limited.

3.4 Applying our approach to Cheon et al. fix

In [Che+22], the authors proposed a simple fix to preserve the wPRF security. Since their attack relies heavily on the Hamming weight of the key $\mathbf{k}$, it is interesting to observe the bias we get from our attack when applying their fix.

Experiment 1. *For $n = 384$, we first generate keys at random. Using Theorem 2, we obtain the predicted bias $\epsilon(\ell)$ where $\ell = \alpha n/2$. Then, we randomly generate $c \cdot \frac{1}{\epsilon(\ell)^2}$ pairs $(\mathbf{x}, \mathbf{x}')$, as in Theorem 2, to observe the bias. Finally, we rerun the experiment with $\omega_H(\mathbf{k}) \gg n/2$, for example, by setting $\Pr[k_i = 1] = \frac{310}{384}$. The result is shown*

in Table 1.[3]

It can be seen from the experiment (albeit not comprehensive), Cheon et al.'s fix results in an even bigger bias in our attack.

Table 1: Experiments for our distinguisher with Cheon et al. fix.

$\ell = \alpha n/2$	6	8	12	15	18
Bias observed for $\omega_H(\mathbf{k}) \gg n/2$	0.161	0.114	0.056	0.0363	0.0211
Bias observed for random $\mathbf{k}$	0.079	0.044	0.013	0.0060	0.0025
Bias predicted from Theorem 2	0.079	0.044	0.014	0.0059	0.0025

4 Conclusion

In this paper, we have examined a differential attack on the binary Alternative Mod-2/Mod-3 wPRF candidate in [Bon+18], improving the state of the art for these constructions. The complexity of the proposed attacks is still exponential in the input and key size, but the results show that proposed parameters for constructions to meet a specific security level need to change. It is likely that further improvements can be made.

5 Future works

An interesting idea to build a distinguisher on the LPN-wPRF would be to use the ideas from Section 3 and generate a large number of low weight differences, and then examine the distribution for the vector $\mathbf{y}$, where $\mathbf{y} = \mathbf{B}(\mathbf{u}+\mathbf{v})$. As was shown in Section 3, $\mathbf{u}+\mathbf{v}$ will be biased if it is created from a low weight difference, which in turn gives a (smaller) bias on $\mathbf{y}$. This bias will depend on properties of the $\mathbf{B}$ matrix. Computations of such large distributions can use methods like those in [MJ05].

6 Acknowledgement

The work presented in this paper has been partly funded by the Swedish Research Council (Grant No. 2019-04166) and the Swedish Foundation for Strategic Research (Grant No. RIT17-0005). We also appreciate comments from ISIT 2023 reviewers for the editorial and technical quality of the paper.

[3] Since $\omega_H(\mathbf{k})$ can vary, the bias, in particular when $\omega_H(\mathbf{k}) \gg n/2$, might slightly fluctuate when replicating the experiment.

References

[AA17] J. Alperin-Sheriff and D. Apon. “Weak is Better: Tightly Secure Short Signatures from Weak PRFs”. In: *IACR Cryptol. ePrint Arch.* (2017), p. 563.

[Ana+15] P. Ananth et al. “From Selective to Adaptive Security in Functional Encryption”. In: *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*. Ed. by R. Gennaro and M. Robshaw. Vol. 9216. Lecture Notes in Computer Science. Springer, 2015, pp. 657–677.

[App14] B. Applebaum. “Bootstrapping Obfuscators via Fast Pseudorandom Functions”. In: *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*. Ed. by P. Sarkar and T. Iwata. Vol. 8874. Lecture Notes in Computer Science. Springer, 2014, pp. 162–172.

[Bal+20b] M. Ball et al. “On the Complexity of Decomposable Randomized Encodings, Or: How Friendly Can a Garbling-Friendly PRF Be?” In: *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*. Ed. by T. Vidick. Vol. 151. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 86:1–86:22.

[BCK96] M. Bellare, R. Canetti, and H. Krawczyk. “Keying Hash Functions for Message Authentication”. In: *Advances in Cryptology - CRYPTO ’96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*. Ed. by N. Koblitz. Vol. 1109. Lecture Notes in Computer Science. Springer, 1996, pp. 1–15.

[Bel15] M. Bellare. “New Proofs for NMAC and HMAC: Security without Collision Resistance”. In: *J. Cryptol.* 28.4 (2015), pp. 844–878.

[BM18] L. Both and A. May. “Decoding Linear Codes with High Error Rate and Its Impact for LPN Security”. In: *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*. Ed. by T. Lange and R. Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 25–46.

[Bon+18] D. Boneh et al. “Exploring Crypto Dark Matter: - New Simple PRF Candidates and Their Applications”. In: *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*. Ed. by A. Beimel and S. Dziembowski. Vol. 11240. Lecture Notes in Computer Science. Springer, 2018, pp. 699–729.

[BR17b] A. Bogdanov and A. Rosen. “Pseudorandom Functions: Three Decades Later”. In: *IACR Cryptol. ePrint Arch.* (2017), p. 652.

[Che+22] J. H. Cheon et al. “Adventures in crypto dark matter: attacks, fixes and analysis for weak pseudorandom functions”. In: *Des. Codes Cryptogr.* 90.8 (2022), pp. 1735–1760.

[Din+21] I. Dinur et al. “MPC-Friendly Symmetric Cryptography from Alternating Moduli: Candidates, Protocols, and Applications”. In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*. Ed. by T. Malkin and C. Peikert. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 517–547.

[DN02] I. Damgård and J. B. Nielsen. “Expanding Pseudorandom Functions; or: From Known-Plaintext Security to Chosen-Plaintext Security”. In: *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*. Ed. by M. Yung. Vol. 2442. Lecture Notes in Computer Science. Springer, 2002, pp. 449–464.

[Dod+12] Y. Dodis et al. “Message Authentication, Revisited”. In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 355–374.

[Dub10] M. Dubiner. “Bucketing coding and information theory for the statistical high-dimensional nearest-neighbor problem”. In: *IEEE Trans. Inf. Theory* 56.8 (2010), pp. 4166–4179.

[EB22b] A. Esser and E. Bellini. “Syndrome Decoding Estimator”. In: *Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part I*. Ed. by G. Hanaoka, J. Shikata, and Y. Watanabe. Vol. 13177. Lecture Notes in Computer Science. Springer, 2022, pp. 112–141.

[EKZ21] A. Esser, R. Kübler, and F. Zweydinger. "A Faster Algorithm for Finding Closest Pairs in Hamming Metric". In: *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2021, December 15-17, 2021, Virtual Conference*. Ed. by M. Bojanczyk and C. Chekuri. Vol. 213. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 20:1–20:21.

[GGM84] O. Goldreich, S. Goldwasser, and S. Micali. "How to Construct Random Functions (Extended Abstract)". In: *25th Annual Symposium on Foundations of Computer Science, West Palm Beach, Florida, USA, 24-26 October 1984*. IEEE Computer Society, 1984, pp. 464–479.

[GJL14] Q. Guo, T. Johansson, and C. Löndahl. "Solving LPN Using Covering Codes". In: *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*. Ed. by P. Sarkar and T. Iwata. Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 1–20.

[Gol86] O. Goldreich. "Two Remarks Concerning the Goldwasser-Micali-Rivest Signature Scheme". In: *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*. Ed. by A. M. Odlyzko. Vol. 263. Lecture Notes in Computer Science. Springer, 1986, pp. 104–110.

[IM98] P. Indyk and R. Motwani. "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality". In: *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*. Ed. by J. S. Vitter. ACM, 1998, pp. 604–613.

[LM13b] V. Lyubashevsky and D. Masny. "Man-in-the-Middle Secure Authentication Schemes from LPN and Weak PRFs". In: *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*. Ed. by R. Canetti and J. A. Garay. Vol. 8043. Lecture Notes in Computer Science. Springer, 2013, pp. 308–325.

[MJ05] A. Maximov and T. Johansson. "Fast Computation of Large Distributions and Its Cryptographic Applications". In: *Advances in Cryptology - ASIACRYPT 2005*. Ed. by B. Roy. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 313–332.

[MO15] A. May and I. Ozerov. "On Computing Nearest Neighbors with Applications to Decoding of Binary Linear Codes". In: *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*. Ed. by E. Oswald and M. Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 203–228.

[MS07] U. M. Maurer and J. Sjödin. "A Fast and Key-Efficient Reduction of Chosen-Ciphertext to Known-Plaintext Security". In: *Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings*. Ed. by M. Naor. Vol. 4515. Lecture Notes in Computer Science. Springer, 2007, pp. 498–516.

[Pie09] K. Pietrzak. "A Leakage-Resilient Mode of Operation". In: *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*. Ed. by A. Joux. Vol. 5479. Lecture Notes in Computer Science. Springer, 2009, pp. 462–482.

A Key-Recovery Attack on the LCMQ Authentication Protocol

Abstract

We present a simple key-recovery attack on the LCMQ Authentication Protocol, an RFID authentication protocol proposed by Li, Gong, and Qin in 2013. We show that a successful attack is performed by solving a *Learning Parity with Noise* instance in a not-too-large dimension. For the proposed LCMQ parameters, the attack requires only a few invocations with the tag under attack. When there is no restriction on the number of invocations, state-of-the-art LPN solvers recover the keys with complexity below 2^{51} and 2^{86}, when attacking LCMQ parameters for security levels 80-bit and 128-bit, respectively. To the best of our knowledge, this is the first attack on LCMQ with complexity below exhaustive key search.

1 Introduction

In recent years, the cryptography and security communities have been focusing on designing secure entity authentication schemes for radio frequency identification (RFID) systems [JW05a]. This is due to the practical demands and theoretical challenges of creating extremely lightweight schemes. RFID systems are systems for automated identification of physical entities using radio frequency transmissions. They consist of very simple and low-cost tags that are attached to physical

Vu Nguyen, Thomas Johansson, Qian Guo. "A Key-Recovery Attack on the LCMQ Authentication Protocol". In *IEEE International Symposium on Information Theory (ISIT) 2024, Athens, Greece*. 2024 IEEE International Symposium on Information Theory (ISIT), pp. 1824-1829, IEEE.

objects. On the other hand, there are powerful readers that get data from the tags. RFID systems are to be found in many different applications, including supply chain management, payments, door locks, and other electronic identification. It is widely expected that RFID systems will dominate the physical identification mechanism market in the future.

The extremely low production cost of RFID tags is critical. For most applications, the price of a tag must be a few cents to be considered affordable. This implies that the low-cost RFID tags lack the necessary computation, communication, storage, and energy capability, posing security and privacy challenges in low-cost RFID systems. Secure and efficient entity authentication are vital aspects of preventing counterfeiting, which is a major attack on identification devices. Therefore, the cryptographic community has developed extremely lightweight authentication schemes to address these challenges.

One of the more interesting directions is the HB-like authentication protocols, most notably [HB01a; JW05a; GRS08c] that have gained much attention in the field of RFID systems. These protocols use only bitwise operations for the protocol participants and have a solid security foundation based on the learning parity with noise (LPN) problem. Including also security proofs [KS06b] makes them very attractive for entity authentication in very resource-constrained devices. A long line of patches for HB-like protocols has appeared in literature [DK07; MP07; LMM08; BC08; GRS08c] to resist man-in-the-middle attack (MIM) that are outside of the original security model [GRS05]. However, they are eventually vulnerable to more advanced methods such as [GRS08a; OOV08].

In response to the attacks on HB-like authentication protocols, Li, Gong, and Qin proposed in 2013 a lightweight authentication protocol named LCMQ [LGQ13b]. Arguments in the paper proved it secure in a general man-in-the-middle model. The scheme uses a special type of circulant matrix and efficient algorithms using binary matrix operations. By combining LPN and a multivariate quadratic problem, the LCMQ protocol appears to be secure against all probabilistic polynomial-time adversaries, and still, it resembles HB-like protocols in terms of tag computations, storage expenses, and communication costs. No attack better than an exhaustive key search has been presented since its introduction.

In this paper, we present a simple key-recovery attack on the LCMQ Authentication Protocol. We show that a successful attack is performed by solving an LPN instance in a not-too-large dimension, by-passing the multivariate quadratic problem. For the proposed LCMQ parameters for 80-bit and 128-bit security levels, the attack complexity is 2^{51} and 2^{86}, respectively, using LPN-solver.

2 Preliminaries

Notation We adopt most of the notation used by the authors in [LGQ13b] throughout the paper.

$\mathbb{F}_2$	The finite field $GF(2)$.
$\mathbf{a}, \omega_H(\mathbf{a})$	Bit vector and its Hamming weight.
$\cdot$	Inner product of vectors.
$\mathbf{M}^{n\times m}$	A $n \times m$ matrix $\mathbf{M}$ over $\mathbb{F}_2$.
$\circ$	Matrix multiplication.
$a(x)$	Polynomial in $\mathbb{F}_2[x]$.
$*$	Polynomial multiplication in $\mathbb{F}_2$.
$\mathbf{0}_n$, $\mathbf{1}_n$	n-bit vectors of all 0 and all 1, respectively.
$\mathbb{S}_n$	Set of all n-bit vectors excluding $\mathbf{0}_n$ and $\mathbf{1}_n$.
$\mathbb{S}_n^e$,$\mathbb{S}_n^o$	Set of vectors in $\mathbb{S}_n$ with even and odd Hamming weight, respectively.

Let Ber_η be the Bernoulli distribution with parameter $\eta \in (0, \frac{1}{2})$. We write $\mathbf{x} \xleftarrow{\$} \{0,1\}^n$ for the event that a vector $\mathbf{x}$ is drawn uniformly from $\{0,1\}^n$.

Definition 3 (LPN Problem). *Let* $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$ *and* $\eta \in (0, \frac{1}{2})$. *An LPN oracle* $\mathcal{O}_{\mathrm{LPN},\mathbf{s},\eta}$ *gives samples*

$$\left(\mathbf{g} \xleftarrow{\$} \{0,1\}^n, \mathbf{s}\cdot\mathbf{g} \oplus e\right),$$

where $e \leftarrow \mathsf{Ber}_\eta$. *The LPN problem asks for the recovery of the secret* $\mathbf{s}$ *given* q *LPN oracle samples.*

As the LPN problem is proven to be NP-hard [BMT78a], it has enjoyed much attention as a key ingredient in constructing post-quantum cryptographic schemes. One can find no shortage of its presence in cryptography. Especially relevant to our work is a field of research that utilizes LPN in crafting lightweight cryptoschemes such as Firekite stream cipher [Bog+21a], weak pseudo-random functions [Bon+18], the aforementioned HB-family of authentication protocols, and the LCMQ authentication protocol [LGQ13b] - the target of this paper.

Despite many of those constructions have been attacked and subsequently forced to use larger parameters [JMN22; Joh23] or severely compromised (such as the HB-like protocols), the LPN problem still stands promising and valuable for cryptographic primitives. In particular, as the authors of the LCMQ protocol demonstrated in their paper, coupled with a neat encryption scheme that involves circulant matrix multiplication, the LPN problem still delivers a secure yet low-cost authentication protocol. More importantly, it was shown to resist known attacks against the HB-family protocols while using significantly smaller parameters.

To understand the security of LPN-based primitives, it is crucial to consider state-of-the-art LPN-solving algorithms. Most notable are the two families of algorithms called *Information-Set Decoding* (ISD), e.g. to name a few, [Pra62; Ste88; MMT11b; Bec+12b; BM18] and *BKW* [BKW03b]. The prior is the main solver for the well-known *Syndrome Decoding Problem* (equivalent to the LPN problem when the amount of samples is fixed), while the latter proves useful when an ar-

bitrary number of samples is allowed. The family of BKW algorithms contains many versions, among them: [BKW03b; LF06b; Kir11b; GJL14]. A Rust implementation for practically solving LPN is found in [WS21].

Before we investigate the LCMQ protocol, we review crucial mathematical backgrounds selectively. For a more comprehensive understanding, we advise the readers to study the LCMQ paper [LGQ13b]. Furthermore, without introducing them in our work, we will point to theorems and lemmas in [LGQ13b] if needed.

Definition 4 (Circulant matrix). *Let* $\mathbf{a} = (a_0, \ldots, a_{m-1})$ *be a length-*m *vectors. A circulant matrix generated by* $\mathbf{a}$ *is a square matrix, denoted by* $\mathbf{C}_{\mathbf{a}}^{m\times m}$ *of which the* i*-th row is the* i*-th right cyclic shift of* $\mathbf{a}$

$$\mathbf{C}_{\mathbf{a}}^{m\times m} = \begin{pmatrix} a_0 & a_1 & \ldots & a_{m-1} \\ a_{m-1} & a_0 & \ldots & a_{m-2} \\ \vdots & \vdots & \vdots & \vdots \\ a_1 & a_2 & \ldots & a_0 \end{pmatrix}.$$

This notion can be extended to non-square circulant matrices: let $n \in [1, m-1]$ *be an integer, then* $\mathbf{C}_{\mathbf{a}}^{n\times m}$ *(resp.* $\mathbf{C}_{\mathbf{a}}^{m\times n}$*) is obtained by truncating* $m-n$ *rows (resp. columns) of* $\mathbf{C}_{\mathbf{a}}^{m\times m}$*.*

Definition 5 (Circulant-P2 Matrix). *Let* $\mathbf{a} \in \mathbb{F}_2^m$ *where* m *is a prime. We call* m *a* $P2$ *number if* $\mathbb{F}_m = \langle 2 \rangle$*, i.e., if* 2 *is a primitive element in* $\mathbb{F}_m$*. Moreover,* $\mathbf{C}_{\mathbf{a}}^{m\times m}$ *is a circulant-P2 matrix if neither* $\mathbf{0}_m$ *nor* $\mathbf{1}_m$ *is one of its rows. Similarly, We extend this notion to non-square circulant-P2 matrices.*

Lemma 5 (Lemma 2 in [LGQ13b]). *If* m *is a P2-number then the polynomial* $x^{m-1} + x^{m-2} + \ldots 1$ *is irreducible over* $\mathbb{F}_2$*.*

Lemma 6 (Lemma 4 in [LGQ13b]). *Let* $\mathbf{a} \in \mathbb{F}_2^m$ *where* m *is a P2-number. Then* $\mathbf{C}_{\mathbf{a}}^{m\times m}$ *is invertible if and only if* $\omega_H(\mathbf{a})$ *is odd.*

Lemma 7 (Lemma 5 in [LGQ13b]). *The matrix* $\mathbf{C}_{\mathbf{a}}^{m\times n}$ *(resp.* $\mathbf{C}_{\mathbf{a}}^{n\times m}$*) always has a right inverse (resp. left inverse).*

Let $f_m(x) = x^m + 1$ and consider the quotient ring $R = \mathbb{F}_2[x]/\langle f_m(x) \rangle$. It then becomes convenient to compute the polynomial multiplication using circulant matrices. Let $\mathbf{a}, \mathbf{b} \in \mathbb{F}_2^m$ be vectorial representation of polynomials $a(x), b(x) \in R$. The multiplication $z(x) = a(x) * b(x)$ in R has representation

$$\mathbf{z} = \mathbf{a} \circ \mathbf{C}_{\mathbf{b}}^{m\times m} = \mathbf{b} \circ \mathbf{C}_{\mathbf{a}}^{m\times m}.$$

3 The LCMQ authentication protocol

The LCMQ protocol combines a *ciphertext-only* secure encryption scheme and LPN, resulting in an efficient and lightweight authentication protocol. The prior

can be seen from the following algorithm (Algorithm 1 [LGQ13b], for correctness).

Algorithm III.1: Inverse of circulant-P2 matrix multiplication

Input: $\mathbf{a} \in \mathbb{S}_m, \mathbf{z} = \mathbf{b} \circ \mathbf{C}_{\mathbf{a}}^{m \times n}, f_m(x) = x^m + 1$.
Output: $\mathbf{b} \in \mathbb{S}_n$.
$g(x) \leftarrow \gcd(a(x), f_m(x))$;
Compute $\mathbf{a}'$, such that $a'(x) * a(x) = g(x) \bmod f_m(x)$;
$t(x) = z(x) * a'(x)$. ;
if $\mathbf{a} \in \mathbb{S}_m^o$ **then**
    $\mathbf{b} \leftarrow$ leftmost n-bit of $\mathbf{t}$;
else
    $b_0 \leftarrow t_0, i \leftarrow 1$;
    while $i < n$ **do**
        $b_i = b_{i-1} \oplus t_i$;
        $i \leftarrow i + 1$;

The encryption is then simply takes a vector $\mathbf{k}$ as the key, the plaintext as $\mathbf{m}$ and output $\mathbf{c} = \mathsf{Enc}(\mathbf{m}, \mathbf{k}) = \mathbf{m} \circ \mathbf{C}_{\mathbf{k}}^{m \times n}$. On the other hand, the decryption $\mathsf{Dec}(\mathbf{c}, \mathbf{k})$ is precisely Algorithm III.1.

After the mathematic background, we can study the LCMQ protocol, which is captured in Figure 1. In this protocol, we denote two entities, Tag and Reader, by $\mathcal{T}_{\mathbf{k}_1,\mathbf{k}_2,\eta,n}$ and $\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}$, where they share the secret keys $\mathbf{k_1}$ and $\mathbf{k}_2$. The Reader sends a "challenge" $\mathbf{a}$. Unless $\mathbf{a} \notin \mathbb{S}_m^e$, in which case the protocol is terminated, the Tag then responds with $(\mathbf{b}, \mathbf{z})$ as described in Figure 1. The $\mathbf{b}$ is randomly selected and then $\mathbf{z} = (\mathbf{y}||\mathbf{r}) \circ \mathbf{C}_{\mathbf{k}_2}^{n \times m}$, where $||$ means concatenation of vectors. Here $\mathbf{y}$ has been constructed by $\mathbf{y} = \mathbf{b} \circ \mathbf{C}_{\mathbf{k}_1}^{m \times n} \oplus \mathbf{v}$, where $\mathbf{v}$ is the low-weight noise vector. On the reader's side, the response is verified by first computing $\mathbf{y}||\mathbf{r} = \mathsf{Dec}(\mathbf{z}, \mathbf{k}_2 \oplus \mathbf{a})$; as described before, to get $\mathbf{y}$. Then it is checked that $\omega_H(\mathbf{b} \circ \mathbf{C}_{\mathbf{k}_1}^{m \times n} \oplus \mathbf{y})$ is small. The verifying threshold τ is set in the interval $(n\eta, \frac{n}{2})$.

3.1 Adversary models

Similarly to its predecessor protocols, LCMQ security was investigated under different models of adversary power, namely the *DET* (detection-based) and *MIM* (man-in-the-middle). We briefly review the main differences in the behaviour of an adversary $\mathcal{A}$ when considered in the aforementioned models.

DET Model The adversary $\mathcal{A}$ operates in two phases:

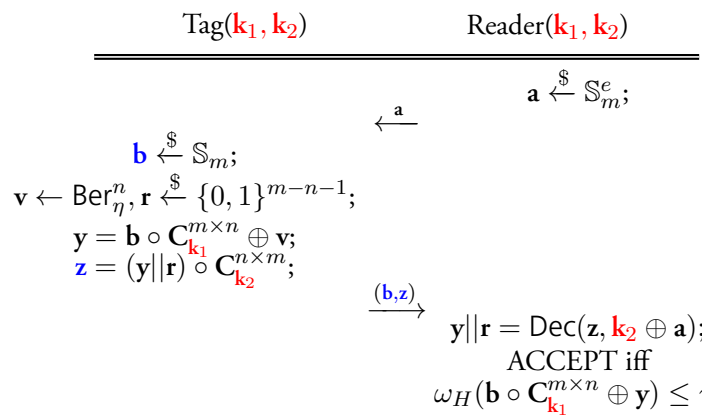


Figure 1: The LCMQ authentication protocol with parameters m as a P2 number, $n < m, \eta \in (0, \frac{1}{2}), \tau \in (n\eta, \frac{n}{2})$, and secret $\mathbf{k}_1, \mathbf{k}_2$ shared by the Tag and the Reader.

During Phase 1, $\mathcal{A}$ interacts q times with the legitimate tag $\mathcal{T}_{\mathbf{k}_1,\mathbf{k}_2,\eta,n}$ by sending challenges $\mathbf{a}_i$ of its choice. During the i-th interaction, the tag carries out computations as described in Figure 1 and responds $\mathcal{A}$ with $(\mathbf{b}_i, \mathbf{z}_i)$.

In Phase 2, $\mathcal{A}$ interacts with the actual LCMQ reader. It receives a challenge $\widehat{\mathbf{a}}$ from $\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}$ and responds with $(\widehat{\mathbf{b}}, \widehat{\mathbf{z}})$ to pass the authentication protocol.

MIM Model The adversary, $\mathcal{A}$ in this model, also carries two phases.

During Phase 1, $\mathcal{A}$ can intercept the communications between $\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}$ and $\mathcal{T}_{\mathbf{k}_1,\mathbf{k}_2,\eta,n}$ q times and observe their behaviour. In particular, it can replace the challenge $\mathbf{a}_i$ from the Reader with $\mathbf{a}_i \oplus \mathbf{a}'_i$ and the response $(\mathbf{b}_i, \mathbf{z}_i)$ from the Tag with $(\mathbf{b}_i \oplus \mathbf{b}'_i, \mathbf{z}_i \oplus \mathbf{z}'_i)$.

Similarly in Phase 2, $\mathcal{A}$ receives a challenge $\widehat{\mathbf{a}}$ from the Reader and outputs $(\widehat{\mathbf{b}}, \widehat{\mathbf{z}})$ correspondingly. One can see that the security in the MIM model is bounded by the security in the DET model as $\mathcal{A}$ is a stronger adversary. Regardless of the power $\mathcal{A}$ possesses, its ultimate goal is to pass the authentication with a non-negligible advantage. Note that the LCMQ protocol inherently has a false-positive probability P_{FP} (i.e., the probability that $\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}$ outputs ACCEPT for a random $(\mathbf{b}, \mathbf{z})$).[1] Therefore, we define the advantages of $\mathcal{A}$ in the above models as

$$\mathsf{Adv}_{\mathcal{A}}^{\text{DET}} := \Pr[\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}(\widehat{\mathbf{b}}, \widehat{\mathbf{z}}) = \text{ACCEPT}] - P_{\text{FP}}, \tag{1}$$

$$\mathsf{Adv}_{\mathcal{A}}^{\text{MIM}} := \Pr[\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}(\widehat{\mathbf{b}}, \widehat{\mathbf{z}}) = \text{ACCEPT}] - P_{\text{FP}}. \tag{2}$$

[1] $P_{\text{FP}} = \sum_{i=0}^{\tau} \binom{n}{i} 2^{-n}$.

Table 1: Parameters recommendation for the LCMQ protocol with the Upper-Bounded Bernoulli Noise Mode in [LGQ13b]

Security Level	80	112	128
η	0.08	0.09	0.10
m	163	269	317
n	162	268	316
τ	18	34	45

3.2 Security and parameters

In this part, we first briefly summarize the strategy the authors in [LGQ13b] employed to show that LCMQ is secure in the MIM model. For the full-version proof, we refer the readers to the original paper. In a nutshell, the arguments go as follows

- Show a reduction from DET-model to MIM-model.
- Assuming the intractability of the LPN problem, the LCMQ protocol is secure in the DET model.
- Therefore, LCMQ is provably secure in the MIM model.

The security proof of LCMQ in the DET model shows that the parameter $m \geq d+1$ is sufficient for d-bit security, and the noise rate is insignificant as long as noises are present. However, the security proof (Theorem 1 [LGQ13b]) requires m to be sufficiently big, and n is set to $n = m - 1$ to minimize the *False positive* probability. Furthermore, the authors in [LGQ13b] showed that the noise-free LCMQ protocol can be seen as an instance of the so-called *MQ Problem* [GJ79], which is defined as finding solutions of m variables with n quadratic equations. This problem is NP-hard for general values of m and n and has been employed in various cryptosystems. Therefore, introducing LPN noise into the MQ problem is assumed only to make it harder to break. The proposed parameters are shown in Table 1.

4 A new attack on LCMQ

In this section, we describe our attack using the parameter choice $n = m - 1$, as this simplifies the description slightly. This is the parameter choice proposed in all instantiations for the different security levels in Table 1. Also, any other choice of n does not change much; such an instance is still susceptible to the proposed attack with roughly the same complexity as if $n = m - 1$.

4.1 The setting

Consider the LCMQ protocol with secret $\mathbf{k}_1, \mathbf{k}_2$, parameters m, n, η, τ, the tag $\mathcal{T}_{\mathbf{k}_1,\mathbf{k}_2,\eta,n}$ and the reader $\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}$.

We aim for a full key recovery attack in the DET model. In this scenario, the adversary $\mathcal{A}$ interacts q times with a legitimate tag in Phase 1. On the i-th invocation, the tag receives a challenge vector $\mathbf{a}_i$ from the adversary. The tag then generates a random $\mathbf{b}_i \xleftarrow{\$} \mathbb{S}_m$ and the noise vector $\mathbf{v}_i \leftarrow \mathsf{Ber}_\eta^n$. It computes

$$\mathbf{y}_i = \mathbf{b}_i \circ \mathbf{C}_{\mathbf{k}_1}^{m\times n} \oplus \mathbf{v}_i, \tag{3}$$

$$\mathbf{z}_i = \mathbf{y} \circ \mathbf{C}_{\mathbf{k}_2\oplus\mathbf{a}_i}^{n\times m}. \tag{4}$$

and responds with $(\mathbf{b}_i, \mathbf{z}_i)$.

In Phase 2, $\mathcal{A}$ now tries to authenticate to $\mathcal{R}_{\mathbf{k}_1,\mathbf{k}_2,n,\tau}$ and measures its advantage through Equation 1. In our case, a successful key-recovery attack clearly gives maximum advantage.

4.2 The attack description

Our attack relies on the fact that $\mathcal{A}$ can challenge the tag with $\mathbf{a}_i$ of its choosing. With this freedom, $\mathcal{A}$ uses the same challenge vector $\mathbf{a}_i = \mathbf{a}, 1 \leq i \leq q$ in each invocation where $\mathbf{a}$ is an arbitrary vector in $\mathbb{S}_m^e$. For notation simplicity, we assume $\mathbf{a} = \mathbf{0}$, although any other value could be used as the recovery of $\mathbf{k}_2 \oplus \mathbf{a}$ implies that of $\mathbf{k}_2$.

We first turn our attention to the equation

$$\mathbf{z}_i = \mathbf{y} \circ \mathbf{C}_{\mathbf{k}_2}^{n\times m}.$$

We observe that this equation can be written in the form of a polynomial multiplication as

$$z_i(x) = y_i(x) * k_2(x) \bmod x^m + 1. \tag{5}$$

Since $x^m+1 = (x+1)*(x^{m-1}+x^{m-2}+\ldots+x+1)$, and it is known that m is a prime such that 2 is a primitive element mod m. This means that $p(x)$, defined as $p(x) = (x^{m-1}+x^{m-2}+\ldots+x+1)$, is an irreducible polynomial over $\mathbb{F}_2[x]$. We can split the above equation using the Chinese Remainder Theorem (CRT) into moduli $(x+1)$ and $p(x)$. However, as $\mathbf{k}_2$ has even weight, it follows that $k_2(x) \equiv 0 \bmod (x+1)$, and the equation (5) reduces to $0 \equiv 0 \bmod (x+1)$.

With respect to polynomials modulo $p(x)$, we define

$$z_i'(x) = z_i(x) \bmod (x^{m-1} + x^{m-2} + \ldots + x + 1),$$

which can be computed from the known $z_i(x)$. Furthermore, let

$$k_2'(x) = k_2(x) \bmod (x^{m-1} + x^{m-2} + \ldots + x + 1),$$

which is unknown. However, the knowledge of $k_2'(x)$ gives $k_2(x) \bmod x^m + 1$, since $k_2(x) \equiv 0 \bmod (x+1)$.

Finally, we note that $y_i(x)$ corresponds to the $(m-1)$-dimentional vector $\mathbf{y}_i$, so $y_i(x)$ has degree at most $m-2$. As such, it is not reduced w.r.t modulo $p(x)$, hence

$$y_i(x) \bmod p(x) = y_i(x).$$

We now can write Equation 5 as

$$z_i'(x) = y_i(x) * k_2'(x) \bmod p(x). \tag{6}$$

Since $p(x) = (x^{m-1} + x^{m-2} + \ldots + x + 1)$ is irreducible, there exists the inverse of $k_2'(x)$ in $\mathbb{F}_2[x]/\langle p(x) \rangle$, denoted by $k_2^{-1}(x)$. Again, this is unknown, but it is in one-to-one correspondence with $k_2(x)$. Multiplying both sides of Equation 6 gives

$$z_i'(x) * k_2^{-1}(x) = y_i(x) \bmod p(x). \tag{7}$$

Denote the unknown key vector $k_2^{-1}(x)$ as $k_2^{-1}(x) = k_0 + k_1 x + \ldots + k_{m-1}x^{m-1}$ and denote $\mathbf{k}_1$ as $\mathbf{k}_1 = (k_m, k_{m+1}, k_{m+2}, \ldots, k_{2m-2})$. Looking at the coefficient for x^j, $0 \leq j \leq 2m-2$, we can see that the left-hand side contains a known linear combination of key bits in $k_0, k_1, \ldots, k_{m-1}$. The right-hand side is of the form $\mathbf{y}_i = \mathbf{b}_i \circ \mathbf{C}_{\mathbf{k}_1}^{m \times n} + \mathbf{v}_i$, so position j is a linear combination of key bits in $k_m, k_{m+1}, k_{m+2}, \ldots, k_{2m-2}$ added with a single noise bit v_j.

Altogether, this means that from each invocation, we can form a system of $(m-1)$ equations as

$$(k_0, k_1, \ldots, k_{2m-2}) \circ \mathbf{H}_i + (v_0, v_1, \ldots, v_{m-1}) = \mathbf{0},$$

where $\mathbf{H}_i$ is an $(2m-2) \times (m-1)$ matrix that is known and can be computed from the bits in the known $\mathbf{z}_i'$ and $\mathbf{b}_i$. As the unknown key vector stays the same for each invocation, this can be viewed as an instance of the LPN problem with dimension $2m-1$ and noise level η, where each invocation gives $m-1$ LPN samples.

The matrix $\mathbf{H}_i$ from each invocation with the tag is constructed from the two matrices in the equations

$$\mathbf{k}_2^{-1} \circ \mathbf{M}_{\mathbf{z}_i'}^{n \times n} = \mathbf{k}_1 \circ \mathbf{C}_{\mathbf{b}_i}^{m \times n} + \mathbf{v}_i,$$

$$(\mathbf{k}_2^{-1} \parallel \mathbf{k}_1) \circ \begin{pmatrix} \mathbf{M}_{\mathbf{z}_i'}^{n \times n} \\ \mathbf{C}_{\mathbf{b}_i}^{m \times n} \end{pmatrix} = \mathbf{v_i}, \tag{8}$$

where $\mathbf{M}_{\mathbf{z}'}^{n\times n}$ corresponds to the matrix representation of the multiplication with $z_i'(x)$ in the field generated by $p(x)$, etc. For instance, assume $m = 5$ and $\mathbf{z}_i' = (z_0', z_1', z_2', z_3')$. Then

$$\mathbf{M}_{z'}^{n\times n} = \begin{pmatrix} z_0' & z_1' & z_2' & z_3' \\ z_3' & z_0'+z_3' & z_1'+z_3' & z_2'+z_3' \\ z_2'+z_3' & z_2' & z_0'+z_2' & z_1'+z_2' \\ z_1'+z_2' & z_1'+z_3' & z_1' & z_0'+z_1' \end{pmatrix}.$$

After q invocations, each providing a matrix $\mathbf{H}_i$, we form the matrix $\mathbf{C} = [\mathbf{H}_1\mathbf{H}_2\cdots\mathbf{H}_q]$, and it is now clear that the nonzero codeword of lowest weight in the code spanned by $\mathbf{C}$ corresponds to the secret key. The code has dimension $2m-1$ and length $q(m-1)$. The weight of the codeword corresponding to the secret key is around $\eta q(m-1)$. We find this codeword by applying an efficient ISD algorithm, and the complexity for different parameters can be measured with the *Syndrome Decoding Estimator* in [EB22c]. If q is very large, we would rather use an LPN solver.

4.3 Applying the attack on proposed parameters

The suggested security parameters of the LCMQ protocol from [LGQ13b] are shown in Table 1. It is claimed that $(m = 163, \eta = 0.08)$ can provide 80-bit security in the LCMQ protocol. However, our attack leads to the problem of solving an LPN instance in dimension 325 with noise $\eta = 0.08$. The complexity for solving such an instance is 2^{51} by using the BKW variant with subspace hypothesis testing [GJL14; GJL20].

For the 128-bit secure parameters, one needs to solve an LPN instance with dimension 633 with noise $\eta = 0.10$. The complexity is estimated to be 2^{86}.

We are also interested in the case when we interact with the Tag as few times as possible. This effectively turns the problem into a coding problem where we are tasked to find a codeword of weight roughly $\omega = q(m-1)\eta$ of a random code generated by a matrix of size $(2m-1)\times q(m-1)$. Note that the minimum distance of such a code can be extrapolated from the GV-bound as the smallest integer d such that

$$2^{q(m-1)-(2m-1)} \leq \sum_{i=0}^{d-1} \binom{q(m-1)}{i}.$$

Therefore, as long as the target weight ω is much smaller than d, the solution should be unique and match the key. Results are presented in Table 2.

Table 2: Complexity w.r.t LCMQ parameters and different smaller q. The complexity is chosen from the best ISD solver.

Security Level	q	ω	d_{GV}	Complexity
$80(m = 163, \eta = 0.08)$	4	52	74	64.6
$80(m = 163, \eta = 0.08)$	8	105	282	58.7
$112(m = 269, \eta = 0.09)$	4	96	121	107.4
$112(m = 269, \eta = 0.09)$	8	193	464	94.9
$128(m = 317, \eta = 0.10)$	8	253	546	119.3
$128(m = 317, \eta = 0.10)$	10	316	772	116.7

4.4 Example 1

Let $m = 5, n = 4$, the keys are $\mathbf{k}_1 = (1, 0, 1, 1, 0), \mathbf{k}_2 = (0, 0, 1, 0, 1)$. Assume $\mathbf{a} = \mathbf{0}$ and during the i-th interaction, the tag generates $\mathbf{v} = (0, 1, 0, 0), \mathbf{b} = (0, 1, 0, 0, 1)$, which gives $\mathbf{y} = (0, 1, 1, 1)$. Then, the adversary $\mathcal{A}$, receives

$$\mathbf{b} = (0, 1, 0, 0, 1), \quad \mathbf{z} = (0, 1, 1, 1, 1).$$

In polynomial terms modulo $f_m(x) = x^5 + 1$, we have

$$\begin{aligned} k_1(x) &= x^3 + x^2 + 1, & k_2(x) &= x^4 + x^2, \\ b(x) &= x^4 + x, & v(x) &= x, \\ z(x) &= x^4 + x^3 + x^2 + x, & y(x) &= x^3 + x^2 + x. \end{aligned}$$

On the other hand, w.r.t modulo $p(x) = x^4 + x^3 + x^2 + x + 1$ we have that $y(x)$ stays the same as before, and then

$$k_2'(x) = x^3 + x + 1, \quad k_2^{-1}(x) = x^3 + x^2 + x,$$

$$z'(x) = 1.$$

We check:

$$\begin{aligned} z(x) * k_2^{-1} &\equiv x^3 + x^2 + x \bmod p(x), \\ b(x) * y(x) + v(x) &\equiv x^3 + x^2 + x \bmod p(x). \end{aligned}$$

When turning $z'(x) * k_2^{-1}(x) \equiv y(x) \bmod p(x)$ to the vector/matrix form, we get from (8)

$$(\mathbf{k}_2^{-1} || \mathbf{k}_1) \circ \begin{pmatrix} \mathbf{M}_{\mathbf{z}'}^{n \times n} \\ \mathbf{C}_{\mathbf{b}}^{m \times n} \end{pmatrix} = \mathbf{v},$$

where

$$\begin{pmatrix} \mathbf{M}_{\mathbf{z}'}^{n\times n} \\ \mathbf{C}_{\mathbf{b}}^{m\times n} \end{pmatrix} = \left(\begin{array}{cccc} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \\ \hline 0&1&0&0 \\ 1&0&1&0 \\ 0&1&0&1 \\ 0&0&1&0 \\ 1&0&0&1 \end{array}\right)$$

4.5 Example 2

Assume now the attack scenario and that $\mathcal{A}$ gets the response from 5 invocations with the tag. For example,

$$(\mathbf{b}_1 = (0,1,0,0,1), \mathbf{z}_1 = (0,1,1,1,1)),$$

as in Example 1 and then

$$(\mathbf{b}_2 = (0,1,1,0,1), \mathbf{z}_2 = (1,0,0,0,1)),$$

$$(\mathbf{b}_3 = (1,1,1,1,0), \mathbf{z}_3 = (1,0,0,0,1)),$$

$$(\mathbf{b}_4 = (1,0,0,0,0), \mathbf{z}_4 = (1,1,1,0,1)),$$

$$(\mathbf{b}_5 = (1,1,1,0,0), \mathbf{z}_5 = (1,0,0,1,0)).$$

The unknown noise values are $\mathbf{v}_2 = \mathbf{v}_3 = \mathbf{v}_5 = \mathbf{0}_n$ and $\mathbf{v}_4 = (1,0,0,0)$. From these responses, $\mathcal{A}$ computes

$$\mathbf{z}_2' = (0,1,1,1), \mathbf{z}_3' = (0,1,1,1), \mathbf{z}_4' = (0,0,0,1), \mathbf{z}_5' = (1,0,0,1).$$

Then a 9×20 matrix $\mathbf{C} = [\mathbf{H}_1\mathbf{H}_2 \cdots \mathbf{H}_q]$ is computed as

$$\mathbf{C} = \begin{pmatrix} 1&0&0&0&0&1&1&1&0&1&1&1&0&0&0&1&1&0&0&1 \\ 0&1&0&0&1&1&0&0&1&1&0&0&1&1&1&1&1&0&1&1 \\ 0&0&1&0&0&1&1&0&0&1&1&0&1&0&0&0&1&0&1&0 \\ 0&0&0&1&0&0&1&1&0&0&1&1&0&1&0&0&0&1&0&1 \\ 0&1&0&0&0&1&1&0&1&1&1&1&1&0&0&0&1&1&1&0 \\ 1&0&1&0&1&0&1&1&0&1&1&1&0&1&0&0&0&1&1&1 \\ 0&1&0&1&0&1&0&1&1&0&1&1&0&0&1&0&0&0&1&1 \\ 0&0&1&0&1&0&1&0&1&1&0&1&0&0&0&1&1&0&0&1 \\ 1&0&0&1&1&1&0&1&1&1&1&0&0&0&0&0&1&1&0&0 \end{pmatrix}.$$

We can directly solve for the key with an ISD algorithm looking for a low-weight codeword in the code generated by $\mathbf{C}$. We first try to look for the minimum-weight codeword. We get the solution

$$\mathbf{k}_{\text{sol}} = (0,1,1,1,1,0,1,1,0),$$

which is precisely $(\mathbf{k}_2^{-1}, \mathbf{k}_1)$.

5 Conclusion

We have presented a very powerful key-recovery attack on the LCMQ Authentication Protocol that renders the scheme completely insecure as it is described. Possible ways to repair the scheme can be considered. We first note that choosing $n < m - 1$ introduces more unknown randomness (the $\mathbf{r}$ value), but it does not change much for the attack. It is performed in the same way, only that each invocation provides n samples instead of the $m - 1$ as described before. Trying to find simple means to repair the scheme is left for future work.

References

[BC08] J. Bringer and H. Chabanne. "Trusted-HB: a low-cost version of HB+ secure against Man-in-The-Middle attacks". In: *CoRR* abs/0802.0603 (2008). arXiv: 0802.0603.

[Bec+12b] A. Becker et al. "Decoding Random Binary Linear Codes in 2n/20: How 1 + 1 = 0 Improves Information Set Decoding". In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by D. Pointcheval and T. Johansson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 520–536.

[BKW03b] A. Blum, A. Kalai, and H. Wasserman. "Noise-tolerant learning, the parity problem, and the statistical query model". In: *Journal of the ACM (JACM)* 50.4 (2003), pp. 506–519.

[BM18] L. Both and A. May. "Decoding Linear Codes with High Error Rate and Its Impact for LPN Security". In: *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*. Ed. by T. Lange and R. Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 25–46.

[BMT78a] E. Berlekamp, R. McEliece, and H. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)" In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 384–386.

[Bog+21a] S. Bogos et al. "Towards Efficient LPN-Based Symmetric Encryption". In: *Applied Cryptography and Network Security*. Ed. by K. Sako and N. O. Tippenhauer. Cham: Springer International Publishing, 2021, pp. 208–230.

[Bon+18] D. Boneh et al. “Exploring Crypto Dark Matter: - New Simple PRF Candidates and Their Applications”. In: *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*. Ed. by A. Beimel and S. Dziembowski. Vol. 11240. Lecture Notes in Computer Science. Springer, 2018, pp. 699–729.

[DK07] D. N. Duc and K. Kim. *Securing* HB+ *against GRS man-in-the-middle attack*. 2007.

[EB22c] A. Esser and E. Bellini. “Syndrome decoding estimator”. In: *IACR International Conference on Public-Key Cryptography*. Springer. 2022, pp. 112–141.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[GJL14] Q. Guo, T. Johansson, and C. Löndahl. “Solving LPN Using Covering Codes”. In: *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*. Ed. by P. Sarkar and T. Iwata. Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 1–20.

[GJL20] Q. Guo, T. Johansson, and C. Löndahl. “Solving LPN Using Covering Codes”. In: *J. Cryptol.* 33.1 (2020), pp. 1–33.

[GRS05] H. Gilbert, M. Robshaw, and H. Sibert. *An Active Attack Against* HB+ *- A Provably Secure Lightweight Authentication Protocol*. `https://eprint.iacr.org/2005/237`. 2005.

[GRS08a] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. “Good Variants of HB + Are Hard to Find”. In: *Financial Cryptography and Data Security*. Ed. by G. Tsudik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 156–170.

[GRS08c] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. “HB#: Increasing the Security and Efficiency of HB+”. In: *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[HB01a] N. J. Hopper and M. Blum. “Secure Human Identification Protocols”. In: *Advances in Cryptology — ASIACRYPT 2001*. Ed. by C. Boyd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 52–66.

[JMN22] T. Johansson, W. Meier, and V. Nguyen. “Attacks on the Firekite Cipher”. In: *IACR Transactions on Symmetric Cryptology* 2022.3 (Sept. 2022), pp. 191–216.

[Joh23] Johansson, Thomas and Meier, Willi and Nguyen, Vu. “Differential cryptanalysis of Mod-2/Mod-3 constructions of binary weak PRFs”. swe. In: *2023 IEEE International Symposium on Information Theory (ISIT)*. IEEE - Institute of Electrical and Electronics Engineers Inc., 2023, 477–482.

[JW05a] A. Juels and S. A. Weis. “Authenticating pervasive devices with human protocols”. In: *Advances in Cryptology–CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005. Proceedings 25*. Springer. 2005, pp. 293–308.

[Kir11b] P. Kirchner. “Improved generalized birthday attack”. In: *Cryptology ePrint Archive* (2011).

[KS06b] J. Katz and J. S. Shin. “Parallel and Concurrent Security of the HB and HB + Protocols”. In: *Advances in Cryptology - EUROCRYPT 2006*. Ed. by S. Vaudenay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 73–87.

[LF06b] É. Levieil and P.-A. Fouque. “An improved LPN algorithm”. In: *Security and Cryptography for Networks: 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006. Proceedings 5*. Springer. 2006, pp. 348–359.

[LGQ13b] Z. Li, G. Gong, and Z. Qin. “Secure and efficient LCMQ entity authentication protocol”. In: *IEEE transactions on information theory* 59.6 (2013), pp. 4042–4054.

[LMM08] X. Leng, K. Mayes, and K. Markantonakis. “HB-MP+ Protocol: An Improvement on the HB-MP Protocol”. In: *2008 IEEE International Conference on RFID*. 2008, pp. 118–124.

[MMT11b] A. May, A. Meurer, and E. Thomae. “Decoding random linear codes in Ó(20.054n)”. In: *Proceedings of the 17th International Conference on The Theory and Application of Cryptology and Information Security*. ASIACRYPT’11. Seoul, South Korea: Springer-Verlag, 2011, pp. 107–124.

[MP07] J. Munilla and A. Peinado. “HB-MP: A further step in the HB-family of lightweight authentication protocols”. In: *Computer Networks* 51.9 (2007). (1) Advances in Smart Cards and (2) Topics in Wireless Broadband Systems, pp. 2262–2267.

[OOV08] K. Ouafi, R. Overbeck, and S. Vaudenay. "On the Security of HB# against a Man-in-the-Middle Attack". In: *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*. Vol. 5350. Lecture Notes in Computer Science. Springer, 2008, pp. 108–124.

[Pra62] E. Prange. "The use of information sets in decoding cyclic codes". In: *IRE Trans. Inf. Theory* 8.5 (1962), pp. 5–9.

[Ste88] J. Stern. "A method for finding codewords of small weight". In: *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*. Ed. by G. D. Cohen and J. Wolfmann. Vol. 388. Lecture Notes in Computer Science. Springer, 1988, pp. 106–113.

[WS21] T. Wiggers and S. Samardjiska. "Practically solving LPN". In: *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2021, pp. 2399–2404.

A New Sieving-Style Information-set Decoding Algorithm

Abstract

The problem of decoding random codes is a fundamental problem for code-based cryptography, including recent code-based candidates in the NIST post-quantum standardization process. In this paper, we present a novel Sieving-style Information-set Decoding algorithm, addressing the task of solving the syndrome decoding problem. Our approach involves maintaining a list of weight-$2p$ solution vectors to a partial syndrome decoding problem and then creating new vectors by identifying pairs of vectors that collide in p positions. By gradually increasing the parity-check condition by one and repeating this process iteratively, we find the final solution(s). We show that our novel algorithm performs better than other ISDs in the memory-restricted scenario when applied to McEliece. Notably, in the case of problem instances with very low relative weight, the sieving approach uses significantly less memory compared to other ISD algorithms while being competitive in terms of performance.

1 Introduction

The recent advancements in the development of quantum computers have greatly impacted cryptography. There is a threat to current standard cryptographic algorithms based on factoring and discrete-log problems, leading to an interest in

Q. Guo, T. Johansson and V. Nguyen. "A New Sieving-style Information-set Decoding Algorithm". In IEEE Transactions on Information Theory, vol. 70, no. 11, pp. 8303-8319, IEEE.

cryptographic algorithms based on other hardness assumptions. Post-quantum cryptography revolves around primitives that are not known to be broken by a large quantum computer.

One leading and promising field in post-quantum cryptography is code-based cryptography. Being introduced in the 70s, it has a long history with many proposed primitives that withstand classical as well as quantum attacks. Code-based cryptography relies on the difficulty of the problem of decoding random codes, which is a very well-studied hardness assumption. The ongoing NIST standardization process for post-quantum cryptography [NIS] includes in round 4 several code-based proposals (Classic McEliece [Ber+20], BIKE [Ara+23], and HQC [Mel+23a]).

One major challenge in these schemes is the selection of secure parameter sets for the proposals, which match the required security levels as decided by NIST. To determine and evaluate parameter sets, the exact cost of the best cryptanalysis tools - Information-set Decoding (ISD) on the proposed schemes and their corresponding hardness assumption is needed. Improving current ISD algorithms, as well as proposing new ones, are therefore of interest, and their complexity parameters, such as time and space, are important.

The problem of decoding random codes, or equivalently, the *Syndrome Decoding Problem* (SDP) can be described as: given a random matrix $\mathbf{H} \in \mathbb{F}_2^{r \times n}$, a syndrome $\mathbf{s} \in \mathbb{F}_2^r$ and an integer ω asks to find an error vector $\mathbf{e} \in \mathbb{F}_2^n$ with weight ω such that $\mathbf{s} = \mathbf{H}\mathbf{e}^\intercal$.

The SDP has been a well-established problem in cryptography and coding theory for more than half a century. Throughout history, the NP-complete class of problems has been the building blocks of cryptography. Similarly, the SDP has proven to be useful in constructing many cryptographic primitives. One can find numerous code-based constructions such as public-key cryptography [NIS; Ber+20; Ara+23; Mel+23a], stream ciphers [FS96], hash functions [AFS05; Ber+11], signatures [CFS01], zero-knowledge protocols [Ste93; Vér96], etc., just to name a few. In particular, the current NIST standardization project for post-quantum public-key cryptosystems includes code-based constructions such as McEliece, BIKE, and HQC. With such importance, it is unsurprising that extensive cryptanalytic efforts have been made to gain trust in code-based primitives.

The best algorithms to solve this problem belong to a class of algorithms known as Information-set Decoding (ISD). The first idea of an ISD algorithm was proposed by Prange in 1962 [Pra62], and then a long line of papers has provided subsequent improvements, see [LB88; Dum91; Leo88; Ste88; BLP08; FS09; Sen11; MMT11a; Bec+12a; MO15; BM17a; BM18]. Recently, an approach called *Statistical Decoding* has been revisited by Carrier et al. [Car+22], and shown to be advantageous in specific code rate regimes.

Most works study the problem for $\omega = cn$, where c is a constant, and investigate the asymptotic runtime exponent. However, for all code-based NIST PQC submissions, as well as other explicit proposals, the asymptotic expressions do not

give the estimated complexity as numbers that can be translated to a security level. Some of the asymptotic advantages of improved ISD algorithms have been shown to be less significant for certain parameter regimes [TS16]. Therefore, it is not clear which algorithms actually yield practical improvements. Another important aspect is that memory requirements are very high in the improved versions of ISD algorithms, and it is likely to be the limiting factor in practice. Hence, any algorithm that requires less memory but has a similar computational complexity offers a valuable alternative. Estimators for concrete complexity of solving the syndrome decoding problem for various algorithms have previously appeared in [HS13; Bal+19] and most recently in [EB22b].

1.1 Related works

An important ISD algorithm is the Stern algorithm [Ste88] that significantly improved the previous work of Prange. Its slightly improved version using the parity-check matrix as suggested in [FS09] is used in our work. Other improvements making use of *representation technique*, as in [MMT11a; Bec+12a], are notable among *enumeration-dominated* ISD. The *Nearest-neighbor Search* was introduced in [MO15] and later used in various steps of the algorithms [BM17a; BM18]. These improved versions of the Stern algorithm share a drawback: they generally require even larger memory, a bottleneck in many situations. Lattice sieving, a method of finding short vectors in a lattice [AKS01b; NV08; BGJ13], is an inspiration for our work. In our case, we are working with the Hamming metric. Our sieving method is, therefore, different from the known efficient lattice sieving methods due to the different metrics. A similar idea was also initiated by Bernstein in a cryptanalysis forum.[1]

While the performance assessment for our algorithm mainly deals with concrete parameters of code-based candidates, recently, Esser, Etinski, Ducas, and Kirshanova [Duc+23] have conducted an asymptotic analysis for our approach. More notably, they have also viewed our subroutine Algorithm IV.5 in Section 3.3 as an *Nearest-neighbor* instant, upon which several asymptotically better solutions were proposed. Their result further supports that the Sieving-style approach can be a promising research direction in code-based cryptanalysis.

1.2 Contributions

We propose a new ISD-like algorithm for solving the SDP, which we call *Sieving-style* ISD. From the simple observation that if two weight-p vectors $\mathbf{x}, \mathbf{y}$ collide (i.e., both have a one) in $p/2$ positions (assuming p is even), then their sum is also a weight-p vector. Moreover, if we impose a '*syndrome condition*', being $\mathbf{Hx}^\intercal, \mathbf{Hy}^\intercal \in \{\mathbf{0}, \mathbf{s}\}$ for some syndrome $\mathbf{s}$, then again $\mathbf{H}(\mathbf{x}+\mathbf{y})^\intercal \in \{\mathbf{0}, \mathbf{s}\}$. Therefore, instead of using birthday-style arguments like in the Stern algorithm and

[1] cryptanalytic-algorithms@list.cr.yp.to

its many subsequent improvements, we can construct new weight-p error vectors by combining in pairs stored weight p vectors, where the two vectors collide in $p/2$ positions. This procedure, together with an iterative increase in the number of considered syndrome positions, gives weight-p vectors fulfilling the syndrome equation.

Given a set $\mathcal{L}$ of small weight p vectors, we derive efficient algorithms for computing the new set of all weight-p vectors of the form $\mathbf{x} + \mathbf{y}$, where $\mathbf{x}, \mathbf{y} \in \mathcal{L}$. This is used as a part of the proposed ISD algorithm. We then analyze the concrete complexity of the proposed algorithm and make comparisons with previous works when considering memory and computational complexity. We argue that our proposed algorithm has better time-memory trade-offs, especially when we restrict the memory. Hence, our algorithm can contribute significantly to understanding the concrete security of code-based cryptographic constructions and improve complexity numbers when the memory is limited.

When comparing the complexity to other ISD algorithms, our approach yields similar performances for instances with very low relative weight (in particular, for code-based candidates such as BIKE or MDPC parameters). In that case, the new algorithm outperforms all previous algorithms w.r.t memory used while achieving comparable computational complexity. Given that high memory requirement is often a bottleneck of many enumeration-based ISDs, our algorithm presents a valuable alternative. In summary, we hope that our research enriches as a novel contribution to post-quantum cryptanalysis.

1.3 Organization

We start by giving preliminaries on coding theory and ISD algorithms in Section 2. In Section 3, we explain the new ideas and describe all parts of the new algorithm. Section 4 presents the complete complexity analysis for the new algorithm. In addition, we show that the *Decoding-one-out-of-many* technique can be applied to our new algorithm. Section 5 then illustrates the performance by making comparisons with some of the best-known ISD algorithms for parameter choices selected from proposed schemes. Section 6 gives some details and results from an actual algorithm implementation for small parameters, supporting theoretical estimates. Section 7 concludes the paper.

2 Preliminaries

Throughout the paper, we use the following notations. We denote by

- bold letters, e.g., $\mathbf{v}$ and $\mathbf{H}$, row vectors and matrices. In particular, $\mathbf{I}_n$ denotes the identity matrix of size $n \times n$.
- $(\mathbf{x})$ the Hamming weight of a vector $\mathbf{x}$.

- $\mathbf{v} = (v_1, v_2, \ldots, v_n)$ a binary vector of length n.
- $\mathbf{x} + \mathbf{y}$ the bit-by-bit XOR between binary vectors $\mathbf{x}$ and $\mathbf{y}$.
- $\mathbb{F}_2$ the binary finite field and $\mathbb{F}_2^{m \times n}$ the vector space over $\mathbb{F}_2$ of dimension $m \times n$.
- log the logarithm base 2.
- $[i] := \{1, \ldots, i\}$ for an integer $i \in \mathbb{N}$.
- $\mathbf{H}_{[i]}$ the matrix restricted to the first i rows of $\mathbf{H}$.
- $\mathbf{v}_{[i]}$ the projection $\mathbf{v}$ onto the coordinates indexed by $[i]$.
- $\mathcal{O}(.)$ the usual Landau notation for the asymptotic behavior of algorithms, and $\tilde{\mathcal{O}}(.)$ means we suppress arbitrary polynomial factors.

We should also point out that all complexity expressions consider the actual complexity in the number of bit operations and not the corresponding asymptotic form of complexity expressions.

2.1 Linear codes and related hard problems

Let $\mathbb{F}_2^n$ be the vector space of all n-tuples over the finite field $\mathbb{F}_2$. A linear code, denoted by $\mathcal{C}$, is a vector subspace of $\mathbb{F}_2^n$. An element of the code $\mathbf{c} = (c_1, \ldots, c_n) \in \mathcal{C}$ where $c_i \in \mathbb{F}_2, i = 1, \ldots, n$ is called a *codeword*. If $\mathcal{C}$ is of dimension k, then we say it to be a $[n, k]$-linear code over $\mathbb{F}_2$. The *minimum distance* d of the code is defined as the minimum Hamming weight of nonzero codewords of $\mathcal{C}$.

A code $\mathcal{C}$ is often represented by a *generator* matrix, which is a $k \times n$ binary matrix $\mathbf{G}$, where the rows constitute a basis of $\mathcal{C}$. Any set of k independent columns of $\mathbf{G}$ forms an *information set* of $\mathcal{C}$. It is also a common practice to denote the remaining coordinates, called *redundancy* of $\mathcal{C}$, by $r = n - k$. Another representation of a code is with a *parity-check matrix*. In particular, there exists an $r \times n$ matrix $\mathbf{H}$ such that $\mathbf{H}\mathbf{c}^\intercal = \mathbf{0}, \forall \mathbf{c} \in \mathcal{C}$. In general, there are many generator and parity-check matrices for a code $\mathcal{C}$. When $\mathbf{G} = \begin{pmatrix} \mathbf{I}_k & \mathbf{A} \end{pmatrix}$ or $\mathbf{H} = \begin{pmatrix} \mathbf{A}^\intercal & \mathbf{I}_{n-k} \end{pmatrix}$, we say that they are in *systematic form*.

Let $\mathbf{y} \in \mathbb{F}_2^n$ be an arbitrary vector, we call $\mathbf{s} = \mathbf{H}\mathbf{y}^\intercal \in \mathbb{F}_2^r$ the syndrome of $\mathbf{y}$ through $\mathbf{H}$. To ease the notation, we omit the transposition and write $\mathbf{y}$ instead of $\mathbf{y}^\intercal$, and it should be clear from the context unless otherwise mentioned. We observe that if $\mathbf{y}$ is not a codeword of $\mathcal{C}$, i.e., $\mathbf{y} = \mathbf{c} + \mathbf{e}$, for some $\mathbf{c} \in \mathcal{C}$ and an "error vector" $\mathbf{e}$, then the syndrome of $\mathbf{y}$ is nonzero and $\mathbf{s} = \mathbf{H}\mathbf{y} = \mathbf{H}\mathbf{e}$.

Definition 6. *Let $\mathcal{C}$ be a $[n, k]$-linear code with a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k) \times n}$. Given a noisy codeword $\mathbf{y} \in \mathbb{F}_2^n$, its syndrome $\mathbf{s} = \mathbf{H}\mathbf{y}$, and an integer $\omega > 0$, the syndrome decoding problem is to find an error vector $\mathbf{e} \in \mathbb{F}_2^n$ such that $(\mathbf{e}) = \omega$,*

$\mathbf{y} + \mathbf{e} \in \mathcal{C}$, *or equivalently* $\mathbf{He} = \mathbf{s}$. *We say that* $\mathbf{e}$ *solves the* $(\mathbf{H}, \mathbf{s}, \omega)$ *instance of the syndrome decoding problem.*

The SDP is closely related to the *coset weights problem*, also known as the decisional SDP (DSDP), which has been shown to belong to the NP-complete complexity class by Berlekamp et al. [BMT78b].

Definition 7. *Let* $\mathbf{H}$ *be a random* $r \times n$ *matrix,* $\mathbf{s}$ *be a vector in* $\mathbb{F}_2^r$*, and* ω *be a positive integer. The coset weights problem is to determine if there exists a vector* $\mathbf{e} \in \mathbb{F}_2^n$ *such that* $(\mathbf{e}) \leq \omega$ *and* $\mathbf{He} = \mathbf{s}$*.*

Arora et al., in [AB09], showed that the search and decision SDP are polynomial-time equivalent, i.e., there exists a polynomial search-to-decision reduction. Therefore, it is common in the literature to say the SDP is NP-complete despite the fact that the definition of NP applies to decisional problems.

2.2 Information-set Decoding Algorithms

The most prominent and well-studied approach to solving the SDP is the class of so-called Information-set Decoding algorithms. In a naive attempt, one can search exhaustively through the space of error vectors with weight ω, which is $\binom{n}{\omega}$ and the complexity is $\tilde{\mathcal{O}}\left(\binom{n}{\omega}\right)$. A long line of studies has gone back to Prange in 1962, who realized we could significantly improve this approach using simple linear algebra. Since then, ISD algorithms have remained an active field of research [Pra62; LB88; Leo88; Ste88; Dum91; BLP08; FS09; Sen11; MMT11a; Bec+12a; MO15; BM17a; BM18]. Following, we describe the general ISD framework and explain some of the technical details of relevant ISD algorithm variants. The essential idea of ISD algorithms is to reduce the search space's dimension with Gaussian elimination. In short, one applies a random permutation $\mathbf{P}$ as

$$\mathbf{He} = \mathbf{HPP}^{-1}\mathbf{e} = \overline{\mathbf{H}}\overline{\mathbf{e}} = \mathbf{s}. \tag{1}$$

A Gaussian elimination process yields an invertible matrix $\mathbf{Q} \in \mathbb{F}_2^{(n-k)\times(n-k)}$ such that

$$\mathbf{Q}\overline{\mathbf{H}}\overline{\mathbf{e}} = \begin{pmatrix}\widehat{\mathbf{H}} & \mathbf{I}_{n-k}\end{pmatrix}\overline{\mathbf{e}} = \mathbf{Qs} = \overline{\mathbf{s}}. \tag{2}$$

Therefore, we can reconstruct a solution of $(\mathbf{H}, \mathbf{s}, \omega)$ by solving the new instance $(\mathbf{Q}\overline{\mathbf{H}}, \overline{\mathbf{s}}, \omega)$. The random permutation $\mathbf{P}$ imposes a particular weight distribution to $\overline{\mathbf{e}} = (\overline{\mathbf{e}}', \overline{\mathbf{e}}'') \in \mathbb{F}_2^k \times \mathbb{F}_2^{n-k}, \omega_H(\overline{\mathbf{e}}') = p < \omega$. Therefore, equation (2) becomes

$$\widehat{\mathbf{H}}\overline{\mathbf{e}}' + \overline{\mathbf{e}}'' = \overline{\mathbf{s}}. \tag{3}$$

In the original Prange's ISD algorithm, one looks for $\mathbf{P}$ that sends all the erroneous bits to the second part, i.e., corresponding to the case of $p = 0$ and $\overline{\mathbf{e}}'' = \overline{\mathbf{s}}$ (equivalently guessing an error-free information-set of $\mathbf{H}$). Therefore, the running time of this algorithm is determined by finding a *correct* permutation, which

happens with probability

$$\Pr_{\text{success}} = \frac{\binom{n-k}{\omega}}{\binom{n}{\omega}}. \tag{4}$$

Intuitively, Prange's ISD is suitable for the low-weight error regime as it is more likely that a random permutation will yield the desired weight distribution. Hence, the original ISD is still one of many main cryptanalysis tools to estimate the security of many code-based cryptosystems, most notably NIST post-quantum candidates such as McEliece, BIKE, or HQC public-key cryptosystems.

In contrast, modern variants of ISD allow some error weight $p > 0$ in $\bar{\mathbf{e}}'$. Therefore, one looks for a weight-p vector $\bar{\mathbf{e}}'$ such that

$$\omega_H(\widehat{\mathbf{H}}\bar{\mathbf{e}}' + \bar{\mathbf{s}}) = \omega - p. \tag{5}$$

Lee and Brickell [LB88] solved the above equation by simply *enumerating* $(\widehat{\mathbf{H}}\bar{\mathbf{e}}' + \bar{\mathbf{s}})$ until a low weight $\bar{\mathbf{e}}''$ is found via (5). Leon in [Leo88] improved this approach by imposing a ℓ-window of zeroes in $\bar{\mathbf{e}}''$; hence, the contribution from the first ℓ bits of $\bar{\mathbf{s}}$ comes only from $\mathbf{e}'$. In particular, we can write again as $\bar{\mathbf{e}} = \left(\bar{\mathbf{e}}', \mathbf{0}^{\ell}, \bar{\mathbf{e}}''\right) \in \mathbb{F}_2^k \times \mathbb{F}_2^{\ell} \times \mathbb{F}_2^{n-k-\ell}$. Although such a constraint reduces the probability of a good permutation, it offers a check via the equation

$$\widehat{\mathbf{H}}_{[\ell]}\bar{\mathbf{e}}' = \bar{\mathbf{s}}_{[\ell]}. \tag{6}$$

It has been shown that such versions of ISD can not gain more than a polynomial factor compared to Prange's ISD.

The first asymptotic improvement came from the Stern ISD algorithm [Ste88] by employing a *Meet-in-the-Middle* strategy to construct the candidates for equation (6). The strategy is to further split up $\bar{\mathbf{e}}' = \mathbf{e}_1 + \mathbf{e}_2$, where $\omega_H(\mathbf{e}_1) = \omega_H(\mathbf{e}_2) = p/2$. Moreover, this approach also mandates that $\mathbf{e}_1$ (and $\mathbf{e}_2$) contributes $p/2$ ones only among the left (right, respectively) $k/2$ coordinates. This is done by storing all $\binom{k/2}{p/2}$ possible values of $(\widehat{\mathbf{H}}_{[\ell]}\mathbf{e}_1 + \bar{\mathbf{s}}_{[\ell]})$ in a look-up table and enumerating all possible values for $\widehat{\mathbf{H}}_{[\ell]}\mathbf{e}_2$. We also notice that the Stern ISD algorithm was also the first variant to introduce a non-polynomial memory requirement, namely, a look-up table of size $\binom{k/2}{p/2}$.

Later, Finiasz et al. [FS09], and Dumer [Dum91] argued that one can increase the success probability of each permutation by removing the window of ℓ-zeroes condition and allowing some error bits to that region. More specifically, instead of a full Gaussian elimination, one can apply a *partial* Gaussian elimination to (1) (with an additional parameter ℓ) and obtain the following form

$$\begin{pmatrix} \mathbf{H}' & \mathbf{0} \\ \mathbf{H}'' & \mathbf{I}_{n-k-\ell} \end{pmatrix} \bar{\mathbf{e}} = \begin{pmatrix} \mathbf{H}' & \mathbf{0} \\ \mathbf{H}'' & \mathbf{I}_{n-k-\ell} \end{pmatrix} \begin{pmatrix} \bar{\mathbf{e}}' & \bar{\mathbf{e}}'' \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{s}}' & \bar{\mathbf{s}}'' \end{pmatrix}, \tag{7}$$

where $\mathbf{H}' \in \mathbb{F}_2^{\ell \times (k+\ell)}$, $\mathbf{H}'' \in \mathbb{F}_2^{(n-k-\ell)\times(k+\ell)}$, $(\bar{\mathbf{e}}', \bar{\mathbf{e}}'') \in \mathbb{F}_2^{k+\ell} \times \mathbb{F}_2^{n-k-\ell}$. Then

we proceed to find (almost) all solutions for the 'small' syndrome decoding instance $(\mathbf{H}', \bar{\mathbf{s}}', p)$ in the form $\bar{\mathbf{e}}' = \mathbf{e_1} + \mathbf{e_2}$, where $\omega_H(\mathbf{e_1}) = \omega_H(\mathbf{e_2}) = p/2$ (in a similar manner as the Stern algorithm), i.e.,

$$\mathbf{H}'\mathbf{e}_1 + \mathbf{H}'\mathbf{e}_2 = \bar{\mathbf{s}}' \tag{8}$$

and then check for

$$\omega_H(\mathbf{H}''(\mathbf{e_1} + \mathbf{e_2}) + \bar{\mathbf{s}}'') = \omega - p. \tag{9}$$

The equations (8) and (9) are sometimes called the *exact matching* and *approximate matching*, respectively, in literature. The state-of-the-art ISD algorithms such as MMT/BJMM [MMT11a; Bec+12a] further speed up the process of constructing $\bar{\mathbf{e}}'$ via the *representation technique*. This practice allows more flexibility on how p error bits are presented in the vector $\bar{\mathbf{e}}'$. We refer the readers to the original works for more details of the representation technique. Subsequently, *Nearest-neighbor Search* [MO15] was introduced to amortize the cost of the approximate matching problem, e.g., as in [BM18].

In comparison with Prange original ISD, whose running time depends on the number of permutations one has to perform (with a polynomial factor for every iteration), enumeration-dominated ISD variants raise the success probability in (4) to

$$\mathrm{Pr}_{\text{success}} = \frac{\binom{n-k-\ell}{\omega-p}\binom{k+\ell}{p}}{\binom{n}{\omega}}. \tag{10}$$

Therefore, modern ISD variants are beneficial in the weight regime where a random permutation is not likely to send all the error weight to $\bar{\mathbf{e}}''$. For concrete security of code-based cryptosystems, enumeration-based ISD remains an essential cryptanalysis tool. However, enumeration introduces significant memory overheads (and the cost of accessing memory). Estimates based solely on the algorithmic steps can, therefore, lead to security underestimation of code-based cryptosystems. Hence, cryptographers have expressed skepticism about how much modern ISD algorithms can improve code-based cryptanalysis, especially for cryptosystems of interest.

To this end, there have been comprehensive surveys of ISD algorithms such as Baldi et al. [Bal+19], Esser-Bellini [EB22b], where concrete bit security estimates for code-based schemes are provided. Importantly, in their works, the memory access cost was taken into consideration to understand better the security of McEliece, HQC, and BIKE. Recently, Esser et al. [EMZ22] provided an efficient implementation of the MMT/BJMM algorithm (by deploying multiple techniques and speed-ups such as the *Parity bit trick, Method of the four Russians for Inversions*, and *Decoding-one-out-of-Many* (DOOM) [Sen11]) with optimized parameters for McEliece and a quasi-cyclic setting. More notably, they also did cryptanalysis with medium-sized instances (60 bits). They showed that the data

from their record computations could be used to extrapolate the bit-security of McEliece/HQC parameters in the NIST standardization process.

3 A new heuristic ISD algorithm

In this section, we describe, in brevity, the main steps of our new ISD algorithm.

3.1 Our ISD framework

We follow the same ISD framework presented in the previous section that derives Equation (7). As in (8), we are now assuming that the first part of the (permuted) error vector, $\bar{\mathbf{e}}'$, is of weight p. So we are looking for all weight-p vectors $\bar{\mathbf{e}}' \in \mathbb{F}_2^{k+\ell}$ that satisfy

$$\mathbf{H}'\bar{\mathbf{e}}' = \bar{\mathbf{s}}'. \tag{11}$$

Once such a vector is found, we can directly compute the corresponding $\bar{\mathbf{e}}''$ giving the desired syndrome and finally check whether the overall weight is ω. When no vector of weight ω is found, we apply a new random permutation, a new partial Gaussian elimination, and the procedure is repeated until success.

Continuing, we assume that the parity-check matrix is already in the form of (7), and from now on, we assume that p is even. Hence, the weight $2p$ is used instead. Moreover, to ease the notation, we refer to matrix and vectors in (11) as $\mathbf{H}$, $\mathbf{e}$, and $\mathbf{s}$. To summarize, we are searching all weight-$2p$ vectors $\mathbf{e} \in \mathbb{F}_2^{k+\ell}$ fulfilling

$$\mathbf{He} = \mathbf{s}, \tag{12}$$

where ℓ is a parameter giving the number of parity-check equations used for the first part $\bar{\mathbf{e}}'$.

3.2 New ideas

The new idea behind our approach is to build an algorithm that keeps a list of weight-$2p$ vectors for which a part of the parity-check equations are fulfilled. From this list, we create a new list of weight-$2p$ vectors for which a larger number of the parity-check equations are met. Iterating this procedure several times, we end up with a final list of weight-$2p$ vectors for which all considered parity checks are fulfilled.

We suggest the following algorithm for computing new weight-$2p$ vectors from old weight-$2p$ vectors based on a modified form of the sieving idea from computing short vectors in lattices [AKS01a; BGJ14]:

Assume that $\mathbf{e}, \mathbf{f}$ are two weight-$2p$ vectors. If they collide in p positions, meaning that $e_i = f_i = 1$ for $i = \{i_1, i_2, \ldots, i_p\}$, then their sum is a new weight-$2p$ vector. In addition, we have one more restriction, namely that the new vector should fulfill a parity-check equation. Recall that $\mathbf{s}_{[i]}$ is the syndrome $\mathbf{s}$

restricted to its first i positions. The parity-check condition used in the first run is

$$\mathbf{H}_{[1]}\mathbf{e} \in \{0, \mathbf{s}_{[1]}\},$$

i.e., only weight-$2p$ vectors fulfilling this condition are kept. In the next run, the parity-check will be considered up to the second coordinate, i.e., $\mathbf{H}_{[2]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[2]}\}$ and so on.

The underlying observation is that if two vectors $\mathbf{e}_1, \mathbf{e}_2$ satisfy $\mathbf{H}_{[i]}\mathbf{e}_j \in \{\mathbf{0}, \mathbf{s}_{[i]}\}$ for $j = 1, 2$, then their sum will also have $\mathbf{H}_{[i]}(\mathbf{e}_1 + \mathbf{e}_2) \in \{\mathbf{0}, \mathbf{s}_{[i]}\}$. Therefore, $\mathbf{H}_{[i+1]}(\mathbf{e}_1 + \mathbf{e}_2) \in \{\mathbf{0}, \mathbf{s}_{[i+1]}\}$ is then fulfilled (when the newly added parity-check is applied) with probability roughly one half.

Let us now explain and discuss the algorithmic description that is to be found in Algorithm IV.1 and Algorithm IV.2. This describes the inner parts of the full ISD algorithm.

Algorithm IV.1: Sieve_Syndrome_Dec()

Input: A parity-check matrix $\mathbf{H}$ with $k + \ell$ columns and ℓ rows, a length ℓ syndrome vector $\mathbf{s}$, the fixed weight $2p$ of the error vectors and list size M.

Output: A set of weight $2p$ vectors $\mathbf{e}$ such that $\mathbf{He} = \mathbf{s}$.

```
Initiate a set L_0 with M vectors of weight 2p;
for i = 1 to ℓ do
    Create the new set L_i ← {e ∈ L_{i−1} : H_[i]e ∈ {0, s_[i]}};
    M_i ← Merge_Set(L_{i−1}, i);
    L_i ← L_i ∪ M_i;
```

Return: $\mathcal{L}_\ell \setminus \{\mathbf{e} : \mathbf{He} = \mathbf{0}\}$.

The algorithm is centered around keeping a set $\mathcal{L}$ of M vectors of weight $2p$. In each iteration i, we aim to generate a new set of weight-$2p$ vectors with the same cardinality, where now one additional parity-check equation from $\mathbf{He} = \mathbf{s}$ is fulfilled. On the one hand, this new set keeps the existing vectors in the set $\mathcal{L}_{i-1}$ (from the previous iteration), for which one more parity check is still valid (that keeps roughly half of them). On the other hand, we create new weight-$2p$ vectors by considering sums of any two vectors in $\mathcal{L}_{i-1}$, which hold the collision condition and fulfill the aforementioned parity-check. This central part of the approach, called the Merge_Set() subroutine, is extracted as Algorithm IV.2 and will be discussed in detail later.

The Merge_Set() subroutine is called ℓ times, corresponding to the number of parity-check equations that need to be satisfied. Note that the parity-check condition in the previous iteration will also be valid in the next. Therefore, we eventually have a 'candidate' list of weight-$2p$ error vectors that match the ℓ bits of syndrome $\mathbf{s}$. Such candidates are subsequently tested for the approximate match-

Algorithm IV.2: Merge_Set()

Input: A parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{\ell \times (k+\ell)}$, a syndrome $\mathbf{s} \in \mathbb{F}_2^{\ell}$, an integer i, and a set $\mathcal{L} = \{\mathbf{e} \in \mathbb{F}_2^{k+\ell} : \mathbf{H}_{[i-1]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i-1]}\}, w_H(\mathbf{e}) = 2p\}$ of size M.

Output: A set $\mathcal{M}$ of vectors of weight $2p$ such that for $\mathbf{e} \in \mathcal{M}$ we have $\mathbf{H}_{[i]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i]}\}$.

Initiate a set $\mathcal{M} \leftarrow \emptyset$;
for $\mathbf{e}, \mathbf{e}' \in \mathcal{L}$ **do**
  If $w_H(\mathbf{e} + \mathbf{e}') = 2p$ then $\mathcal{M} \leftarrow \mathcal{M} \cup (\mathbf{e} + \mathbf{e}')$;
Return: $\mathcal{L}_\ell \setminus \{\mathbf{e} : \mathbf{He} = \mathbf{0}\}$.

ing condition as in (9). Putting everything together, we have a high-level description of our Sieving-style ISD algorithm as in Algorithm IV.3.

Algorithm IV.3: Full_ISD

Input: A matrix $\mathbf{H}$ with $n-k$ rows and n columns, received length n vector $\mathbf{y}$, required weight ω and algorithm parameter ℓ.

Output: A weight-ω vector $\mathbf{e}$ such that $\mathbf{Hy} = \mathbf{He}$.

Compute the syndrome $\mathbf{s} = \mathbf{Hy}$;
repeat
  Pick a random row permutation $\mathbf{P}$; ;
  $\overline{\mathbf{H}} \leftarrow \overline{\mathbf{H}}\mathbf{P}, \bar{\mathbf{e}} \leftarrow \mathbf{P}^{-1}\bar{\mathbf{e}}$; ;
  Perform (partial) Gaussian elimination on $\overline{\mathbf{H}}$ resulting in $\widehat{\mathbf{H}}\bar{\mathbf{e}} = \begin{pmatrix} \mathbf{H}' & 0 \\ \mathbf{H}'' & \mathbf{I}_{n-k-\ell} \end{pmatrix} \begin{pmatrix} \bar{\mathbf{e}}' & \bar{\mathbf{e}}'' \end{pmatrix} = \bar{\mathbf{s}} = \begin{pmatrix} \bar{\mathbf{s}}' & \bar{\mathbf{s}}'' \end{pmatrix}$; ;
  Let $\mathbf{H}' = \widehat{\mathbf{H}}_{[\ell]}$ and $\bar{\mathbf{s}}' = \bar{\mathbf{s}}_{[\ell]}$; ;
  $\mathcal{L} \leftarrow$ Sieve_Syndrome_Dec$(\mathbf{H}', \bar{\mathbf{s}}', 2p)$;;
  for $\tilde{\mathbf{e}} \in \mathcal{L}$ **do**
    if $\omega_H(\mathbf{H}''\tilde{\mathbf{e}} + \mathbf{s}) = \omega_H(\tilde{\mathbf{e}}'') = \omega - 2p$ then return $\mathbf{P}^{-1}(\tilde{\mathbf{e}}, \tilde{\mathbf{e}}'')$;
until *solution is found*;

3.3 The Merge_Set() algorithm

As introduced above, the Merge_Set() algorithm operates on a set of weight-$2p$ vectors and should return any weight-$2p$ sum of two such vectors. There is an additional parity-check requirement, but since this is valid for half of the vectors, it does not pose a problem. We simply check for each sum vector of weight $2p$.

In short, the problem is to find an efficient way of generating pairs of vectors that sum to a new weight-$2p$ vector.[2] A naive implementation of Algorithm IV.2 would require checking all pairs of vectors, hence requiring quadratic (in list size) complexity.

Let a (low-weight) vector $\mathbf{e}$ be represented by the indices of its ones, i.e., $(i_1, i_2, \ldots, i_{2p})$ in rising order, written $\mathbf{e} \sim (i_1, i_2, \ldots, i_{2p})$. We want to find two vectors that share p indices. In a first attempt to find an efficient solution, we could generate $\binom{2p}{p}$ 'labels' for each vector. A label would be a selection of p out of the $2p$ indices for the vector. With M vectors in total, we would have $\binom{2p}{p} \cdot M$ such labels. They would then be stored in a sorted way so that collisions among them are detected. Labels of the form $(i_1, i_2, \ldots, i_p)$ can be mapped to integers and, with a hash table, one can then get close to complexity $\binom{2p}{p} \cdot M$ and the same memory.

Algorithm IV.4: Merge_Set_Implementation0

Input: A parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{\ell \times (k+\ell)}$, a syndrome $\mathbf{s} \in \mathbb{F}_2^{\ell}$, an integer i, and a set $\mathcal{L} = \{\mathbf{e} \in \mathbb{F}_2^{k+\ell} : \mathbf{H}_{[i-1]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i-1]}\}, w_H(\mathbf{e}) = 2p\}$ of size M. Parameters p', p''.

Output: A set $\mathcal{M}$ of vectors of weight $2p$ such that for $\mathbf{e} \in \mathcal{M}$ we have $\mathbf{H}_{[i]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i]}\}$.

1 Declare and initiate parameter (set of vectors) $\mathcal{M} \leftarrow \emptyset$;
2 Find_Collision($\mathcal{L}$, p, p',0);
3 **return** $\mathcal{M} = \{\mathbf{e} \in \mathcal{M} : \mathbf{H}_{[i]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i]}\}\}$;

However, we propose a more balanced implementation, where we, in particular, reduce the amount of memory since $\binom{2p}{p} \cdot M$ can be prohibitively large compared to M. This approach is described in Algorithm IV.4 together with Algorithm IV.5. For the latter, we use recursive calls to ease the description of the procedure. We first describe the basic ideas of the procedure, and later, we revisit the exact steps of Algorithm IV.4 and Algorithm IV.5.

We split p (and vectors, correspondingly) into two parts as $p = p' + p''$. Each vector given by $(i_1, i_2, \ldots, i_{2p})$ parses into $(i_1, i_2, \ldots, i_{p'})$ as a first part and $(i_{p'+1}, \ldots, i_{2p})$ as a second part. We consider the set $\mathcal{L}$ of length-$(k+\ell)$ and weight-$2p$ vectors to be arranged in a number of 'buckets', where each bucket initially contains the vectors, of which the first part is $(i_1, i_2, \ldots, i_{p'})$.

[2] It can also be viewed as a special variant of the Nearest-neighbor (NN) problem. Although NN-solvers such as *Local-sensitive Hashing* or *Nearest-neighbor Search* could apply, due to our setting (e.g., very low weight $2p$ and fixed weight samples) and interest in the concrete analysis, it is unclear to what extend a direct application of such algorithms can be useful.

It means that the number of p'-buckets is $\binom{k+\ell-p''}{p'}$. Note that each vector is only in one bucket. Furthermore, p' should be chosen in such a way that there will likely be some collisions inside each bucket. Now we consider the first bucket, indexed by $(1, 2, \ldots, p')$. The vectors in this bucket already collide in p' positions, and we seek pairs of vectors that collide in an additional p'' positions out of the $2p - p'$ remaining ones. This is done in the following way. We assume we can access an array $\mathbf{A}$ of size $\binom{k+\ell-p'}{p''}$, indexed by p'' positions. For each vector in the bucket, we create the $\binom{2p-p'}{p''}$ different possible combinations of the remaining p'' positions and write a one in the corresponding position in $\mathbf{A}$. Also, if there was already a one in that position, we have found a collision, which is recorded. Finally, after all collisions in a bucket are found, the vectors are placed in their 'next bucket'-indexed by the next value for the p' positions.

We now elaborate this idea by describing the procedure in a recursive way as in Algorithm IV.5. To give a brief explanation, it starts with a call to Find_Collision(), looking for collisions in p positions. It has a bucket (list) of vectors as input. These vectors are now placed in new buckets, depending on the vector's first index value. In bucket $\mathcal{B}_1$, all vectors have a one in position 1, so within $\mathcal{B}_1$, we only need to look for collisions in $p - 1$ additional positions. Therefore, the call to Find_Collision$(\mathcal{B}_1, p - 1, p' - 1, i + 1)$. Once this call has returned possible collisions, the vectors in $\mathcal{B}_1$ may still collide in other ways, excluding position 1. This is why we then move the vectors to the next bucket corresponding to the second lowest index in the vector. Since the position 1 was removed from further combinations, the vector now has only $2p - 1$ indices. Let us assign an index x where the vectors are considered to 'start'.

If the remaining depth is not zero (checked in Line 1), we are simply going to put the vectors in different buckets $\mathcal{B}_{x+1}, \ldots, \mathcal{B}_{k+\ell}$ depending on their next index that is greater than x. For instance, if the next index in order is y, the vector is put in bucket $\mathcal{B}_y$ (Line 2). Then we go through all these buckets in order and find all collisions in bucket $\mathcal{B}_i$ by the call Find_Collision$(\mathcal{B}_i, p-1, p'-1, i+1)$ (Line 4). Note that since all vectors in bucket $\mathcal{B}_i$ already collide in position i, we decrease the depth and required collision while increasing the index by one.

Once all collisions in $\mathcal{B}_i$ have been found, these vectors may provide further collisions in indices i. Thus, we must move the vectors in $\mathcal{B}_i$ to the next bucket corresponding to the next index that is greater than i. This is done according to Line 5.

When the remaining depth is 0, there are not enough vectors in the input bucket to further motivate a split in smaller buckets. Instead, we now directly find the collisions. For this purpose, we use an array $\mathbf{A}$, indexed by p''-tuples. For each vector, we create all possible p''-tuples of its remaining indices $(i_j, i_{j+1}, \ldots, i_{2p})$ and we write up $\mathbf{A}$ by one in each such position (Line 11). We also keep the address to the vector $\mathbf{v}$ in an array $\mathbf{D}$ where we assume that in each entry, we can store a few elements (Line 14). While updating the array, one may hit an index where $\mathbf{A}$

Algorithm IV.5: Find_Collision()

Input: A set $\mathcal{L}$ of vectors of length $k+\ell$; collision weight p; depth sizes p'; first index x.

Output: All vectors of the form $\mathbf{x}+\mathbf{y}$, where $\mathbf{x},\mathbf{y} \in \mathcal{L}$ and they collide in p positions, written to global parameter $\mathcal{M}$.

```
1  if p' > 0 then
2      Put the vectors in L in new buckets B_{x+1}, ..., B_{k+ℓ} depending on
         its first index greater than x;
3      for i = x + 1 ... k + ℓ do
4          Find_Collision(B_i, p − 1, p' − 1, i + 1);
5          Move the vectors in B_i to its next buckets among
             B_{i+1}, ..., B_{k+ℓ} depending on its first index greater than i;
6  else
7      Initiate two arrays A ← 0, D ← 0 ;
8      for each vector v ~ (i_j, i_{j+1}, ..., i_{2p}), j > x, in B_i do
9          create a set Y of all its p''-tuples. ;
10         for each p''-tuple y = (y_1, y_2, ..., y_{p''}) ∈ Y do
11             A[y] ← A[y] + 1;
12             if A[y] ≥ 2 then
13                 store v + D[y] = {v + u, u ∈ D[y]} as collisions in M
14             D[y] ← D[y] ∪ {v};
```

is already non-zero (meaning collisions). One directly writes them to the global output parameter $\mathcal{M}$.

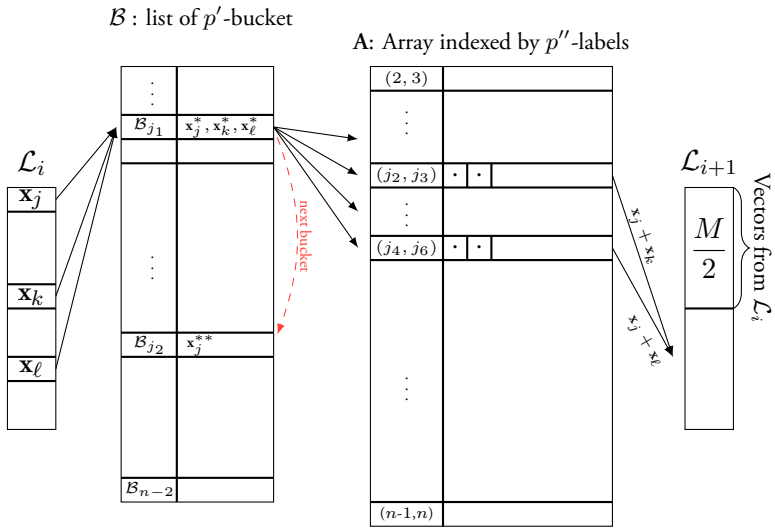


Figure 1: Checking vector in $\mathcal{L}_i$ and Merge_Set().

Example 2. *We can visualize the checking step and* Merge_Set() *(Lines* **3** *and* **4** *in Algorithm IV.1) by Figure 1. For simplicity, let* $p = 3$, $p' = 1$, *and* $p'' = 2$. *In the* i*-th iteration, we have a list* $\mathcal{L}_i$ *of vectors. First, we put vectors in* $\mathcal{L}_i$ *in buckets corresponding to their first coordinate. Assume we have* $\mathbf{x}_j, \mathbf{x}_k, \mathbf{x}_\ell \in \mathcal{L}_i$ *where* $\mathbf{x}_j \sim (j_1, \ldots, j_{2p})$ *(and so forth), and they have the same first coordinate, i.e., they are put in* $\mathcal{B}_{j_1}$. *Then we only need to proceed with their shortened versions, written as* $\mathbf{x}^*_j \sim (j_2, \ldots, j_{2p})$, *etc., since we have excluded the first coordinate. We then detect collisions in this 'bucket' by producing* p'' *labels for each vector and marking them on* **A** *correspondingly. For example, if both* $\mathbf{x}^*_j, \mathbf{x}^*_k$ *include* (j_2, j_3), *then we potentially have* $\mathbf{x}_j + \mathbf{x_k}$ *as a 'good' combination to be added in* $\mathcal{L}_{i+1}$.

After processing $\mathcal{B}_{i_1}$, *we move (dashed red line) vectors in this bucket to their next buckets. For instance,* $\mathbf{x}_i$ *to* $\mathcal{B}_{i_2}$, $\mathbf{x}_j$ *to* $\mathcal{B}_{j_2}$ *and so forth. We now exclude the first two coordinates of* $\mathbf{x}_j$ *(hence, we use* $\mathbf{x}^{**}_j \sim (j_3, \ldots, j_{2p})$*). Note that in the list* $\mathcal{L}_{i+1}$, *we also have half the vectors from* $\mathcal{L}_i$ *that survive the syndrome condition.*

4 Analysis of the new ISD algorithm

This section provides estimations on the time complexity, denoted C, and the space complexity. The space is essentially the number of stored vectors M (so it is not given in bits). Some smaller additional memory is required for other parts of the algorithm.

4.1 The list size M and parameters selection

We first determine the list size M required for the new algorithm to work. Let us recall that the inner iteration of our ISD algorithm, i.e., the Sieve Syndrome Decoding (Algorithm IV.1), consists of two steps: the Merge_Set subroutine, and verifying the next parity-check for vectors in the list that we are processing. Assume that we initiate Algorithm IV.1 with a list $\mathcal{L}_0$ where $|\mathcal{L}_0| = M$ and we aim to keep the list size constant after every (or the majority of) iteration of the parity-check condition. At the i-th iteration, one has for each $\mathbf{e} \in \mathcal{L}_i$ that

$$\omega_H(\mathbf{e}) = 2p, \text{ and } \mathbf{H}_{[i]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i]}\}.$$

We observe that, on average, half of them shall satisfy the next parity-check condition, i.e., $\mathbf{H}_{[i+1]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{[i+1]}\}$. Therefore, we choose M that yields another $\frac{M}{2}$ 'good' combinations. We denote the probability of two random weight-$2p$ vectors of length $k+\ell$ colliding in precisely p positions (of the ones) by q, then

$$q = \frac{\binom{2p}{p}\binom{k+\ell-2p}{p}}{\binom{k+\ell}{2p}}.$$

Given a list of M vectors, we can form $\frac{M(M-1)}{2} \approx \frac{M^2}{2}$ combinations. However, as will be explained later, a significant part of our vectors are actually dependent. Two phenomena arise: 1) We have *more* combinations than the uniformly random case, and 2) Combinations can be already existing vectors (called duplicates). For this purpose, we introduce δ as the fraction of all combinations that give rise to *new* vectors. Continuing, on average, new weight-$2p$ vectors survive the parity-check with probability $\frac{1}{2}$. In conclusion, we require

$$\frac{\delta \cdot M^2 \cdot q}{2 \cdot 2} \approx \frac{M}{2}$$

or

$$M \approx \frac{2}{\delta \cdot q}. \tag{13}$$

Let us define

$$\mathcal{N} = \{\mathbf{e} \in \mathbb{F}_2^{k+\ell} | \omega_H(\mathbf{e}) = 2p \text{ and } \mathbf{He} = \mathbf{s}\},$$

which is the number of solutions for the exact matching equation (8) (not to be confused with the original syndrome decoding problem). One can expect that the cardinality of N is around

$$\frac{\binom{k+\ell}{2p}}{2^\ell}.$$

The final list of Sieve_Syndrome_Dec contains around $M/2$ solutions of the exact matching equation (the other half yields null syndrome). If ℓ is not too

large, there will be many possible solutions, and they all need to be stored in the final list. Therefore, to guarantee that our ISD algorithm is able to retrieve all (or most) solutions of the exact matching problem, we need that

$$M \geq \frac{\binom{k+\ell}{2p}}{2^{\ell-1}}. \tag{14}$$

In conclusion, the list size is first set by (13). Then, to find the optimal parameters for our algorithm, we search for $p \in [0, \omega]$, and ℓ in a 'reasonable' range[3], so that (14) holds, and we select the parameters that yield the lowest complexity.

4.2 Heuristic arguments for duplicated vectors

In this subsection, we investigate the fact that due to dependency in our list of vectors, there will be some combinations that do not contribute to the new list. We stress in advance that the duplicates originate not only by chance but also from the nature of our proposed algorithm (in particular, due to keeping half of the list from previous iterations). In other words, we will see that the duplicates are entirely different from the situation of "many representations" for weight-$2p$ vectors in the representation-based ISD algorithms. In short, we try to heuristically determine δ in Equation (13) as a countermeasure to the latter, as well as the number of combinations we have to compute in each iteration.

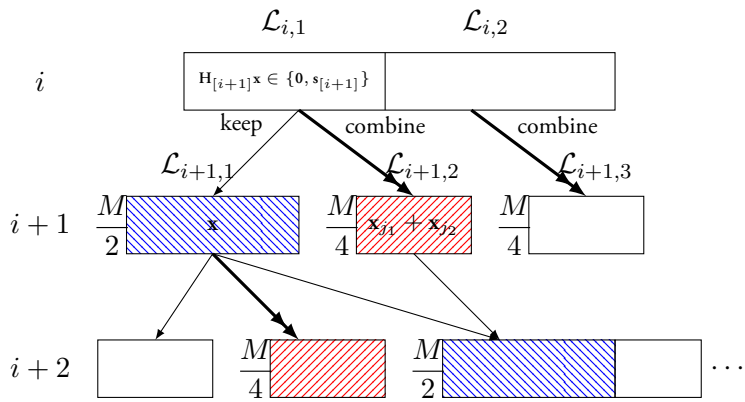


Figure 2: An example of duplicates

Let us look at three consecutive iterations in Sieve_Syndrome_Dec() as in Figure 2. Vectors in $\mathcal{L}_i$ can be split into two sets: $\mathcal{L}_{i,1}$ as the set of vectors from the i-th iteration that also fulfill the syndrome condition up to iteration $i+1$, and $\mathcal{L}_{i,2}$ as the rest. Next, the vectors in iteration $i+1$ are made of three sets

[3] Similar to Baldi et al. in [Bal+19]. We extend the range of ℓ until the optimal value of ℓ is no longer on the edge of the range.

$\mathcal{L}_{i+1,j}, j = 1, 2, 3$. Vectors from $\mathcal{L}_{i+1,1} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ which is a set of size $\frac{M}{2}$ (directly kept from $\mathcal{L}_{i,1}$); $\mathcal{L}_{i+1,2}$ (resp. $\mathcal{L}_{i+1,3}$) as sums of two vectors both from $\mathcal{L}_{i,1}$ (resp. $\mathcal{L}_{i,2}$) which is a set of size $\frac{M}{4}$. [4]

Note that there can be no sum of one vector from $\mathcal{L}_{i,1}$ and one from $\mathcal{L}_{i,2}$, as then the syndrome condition is not fulfilled.

We now look at all the different combinations that give already existing vectors. First, any combination of two vectors from $\mathcal{L}_{i+1,1}$ already exists in $\mathcal{L}_{i+1,2}$. They will account for $(\frac{M}{2})^2 \cdot \frac{q}{2} = \frac{M^2 \cdot q}{8}$ combinations that do not contribute. Then there are also duplicates when combining $\mathcal{L}_{i+1,1}$ and $\mathcal{L}_{i+1,2}$. When a vector $\mathbf{x}_{j_1} + \mathbf{x}_{j_2} \in \mathcal{L}_{i+1,2}$ is added to either $\mathbf{x}_{j_1} \in \mathcal{L}_{i+1,1}$ or $\mathbf{x}_{j_2} \in \mathcal{L}_{i+1,1}$, there will be a duplicate. Since $|\mathcal{L}_{i+1,2}| = \frac{M}{4}$, and for each $\mathbf{x}_{j_1} + \mathbf{x}_{j_2}$, we can have two duplicates which are $\mathbf{x}_{j_1}$ and $\mathbf{x}_{j_2}$. Therefore, the number of generated duplicates is of order $\frac{M}{2}$. Then, we may also have additional duplicates from other combinations.

Again, besides the 'useless' combinations from $\mathcal{L}_{i+1,1}$ and $\mathcal{L}_{i+1,2}$, we also have other additions that can give new vectors. In particular, the type $\mathbf{x}_{j_1} + \mathbf{x}_{j_2} + \mathbf{x}_k$ where $\mathbf{x}_{j_1} + \mathbf{x}_{j_2} \in \mathcal{L}_{i+1,1}$ and $\mathbf{x}_k \in \mathcal{L}_{i+1,2}, \mathbf{x}_k \neq \mathbf{x}_{j_1}, \mathbf{x}_{j_2}$. Additions of this kind yield approximately $\frac{M}{4} \cdot (\frac{M}{2} - 2) \cdot q \approx \frac{M^2 \cdot q}{8}$ new vectors. To summarize, it is necessary that the total number of combinations is around M, excluding the duplicates. In other words, one has

$$\begin{aligned} & M^2 \cdot q \cdot \left(\underbrace{\frac{1}{8}}_{(1,1)} + \underbrace{\frac{1}{8}}_{(1,3)} + \underbrace{\frac{1}{32}}_{(2,2)} + \underbrace{\frac{1}{16}}_{(2,3)} + \underbrace{\frac{1}{32}}_{(3,3)} \right) \\ & + \underbrace{\left(\frac{M}{2} + \frac{M^2 \cdot q}{8} \right)}_{(1,2)} \underbrace{\left(\frac{M}{2} + \frac{M^2 \cdot q}{8} \right)}_{\text{duplicates}} \\ & \approx M, \end{aligned}$$

where (j, k) represents the combinations between $\mathcal{L}_{i+1,j}$ and $\mathcal{L}_{i+1,k}$. Simplifying, one obtains

$$M \approx \frac{8}{3 \cdot q} = \frac{4}{3} \cdot \frac{2}{q}. \tag{15}$$

This corresponds to selecting $\delta = \frac{3}{4}$ in Equation (13). However, this is not sufficiently small due to the other (rarer) duplicates. To be conservative, we choose $\delta = \frac{2}{3}$, and it is more than sufficient according to simulations.

The number of duplicates is then *at least* (excluding other rarer cases)

$$\frac{M}{2} + \frac{M^2 \cdot q}{8} = \frac{7 \cdot M}{8}.$$

[4] Both sets are made of $\frac{M^2 \cdot q}{4}$ combinations. Assume $\mathbf{s}_{i+1} = 0$, so $\mathcal{L}_{i,2} = \{\mathbf{x} : \mathbf{H}_{i+1}\mathbf{x} \in \{(\mathbf{s}_{[i]}, 1), (\mathbf{0}_{[i]}, 1)\}\}$. We can check that one can combine roughly the same way (compared to vectors in $\mathcal{L}_{i,1}$) to be in $\{\mathbf{s}_{[i+1]}, \mathbf{0}_{[i+1]}\}$. Therefore, $|\mathcal{L}_{i+1,2}| = |\mathcal{L}_{i+1,3}| = \frac{M}{2 \cdot 2}$.

We are motivated by this heuristic estimate and expect (conservatively) to have to create around $2 \cdot M$ combinations for each iteration. Note that: 1) this estimate is not necessarily true once the number of possible weight-$2p$ vectors decreases in later iterations (in a sense, this estimate allows us to be on the safe side in (many) early iterations by maintaining the list size) and 2) the constant δ covers the duplicates as described in Figure 2. Other duplicates (rarer, arose by chance) depend on the parameters k, ℓ, and p.

Example 3. *We supported our heuristic arguments of δ with simulations. We test various sets of parameters, and simulations confirm the heuristic arguments. We stop* Merge_Set *once we observe that the list size is maintained, and we look at the number of total combinations we have done. For example, two (out of many tested) parameter sets are $(k, \ell, p) = (500, 30, 2)$ and $(k, \ell, p) = (1000, 30, 2)$. We record the total amount of collisions and duplicates for each iteration.*

- *For $(k, \ell, p) = (1000, 30, 2)$, we have $M \approx 2^{15.35}$. For the majority of iterations, we obtain M unique vectors (hence, $\frac{M}{2}$ survive after the check). The ratio between duplicates and M varies around $\frac{7}{8}$ and peaks at 0.93 (i.e. we create at most $0.93 \cdot M$ duplicates).*
- *For $(k, \ell, p) = (500, 30, 2)$, we have $M \approx 2^{13.43}$. We observe similar behavior and the ratio between duplicates and M peaks at 1.*

4.3 Memory requirement

The memory required for the algorithms is broken down into: 1) the list size M, the list $\mathcal{B}$ of p' "buckets", and the size of the array $\mathbf{A}$ used for detecting collisions. In our optimization, we selected parameters p' and p'' so that $|\mathbf{A}| = \binom{k+\ell-p'}{p''}$ and $|\mathcal{B}| = \binom{k+\ell-p''}{p'}$ are not greater than the list size M. Moreover, we will see from our result tables (e.g., Table 1 and Table 7) that the memory is predominantly determined by M. Therefore, the memory requirement in bits, denoted by $\hat{M}$ for our algorithm is:

$$\log(\hat{M}) = \log(M) + \log(n). \tag{16}$$

Although the memory can be further reduced by only storing the indices of the weight-$2p$ vectors, and the algorithm benefits from such optimization when p is very small[5], we refrain from such optimization to be fair in comparison with other algorithms (that also use $\log(n)$ in calculating the memory in bits).

4.4 The probability of finding a desired vector

We next provide some heuristic arguments concerning the probability of finding one or several desired vectors, i.e., if the code contains a weight-$2p$ code-

[5] For example, we can store only the position of the indices of the ones instead of the full-length vector.

word, what is the probability that it is included in the list given as output from Sieve_Syndrome_Dec()?

Recall the assumption that, throughout the Sieve_Syndrome_Dec(), we have M unique vectors moved from one iteration to the next. However, when i is large enough, this will no longer be true. We now introduce M_i' as the expected number of weight-$2p$ vectors that fulfill up to i parity-check conditions. Then

$$M_0' = \binom{k+\ell}{2p}$$

and

$$M_i' = \frac{\binom{k+\ell}{2p}}{2^i}.$$

Note that we also have the same amount of weight $2p$ vectors that fulfill the null syndrome $\mathbf{0}_{[i]}$.

Now, the heuristic argument is that the set of generated vectors in iteration i is a random selection among all M_i' vectors. So for each created vector in the iteration, we view it as a random pick. Let $M_i = |\mathcal{L}_i|$ denote the list size in iteration i, where $M_i \leq M$. Then the M_i vectors come from the primary check and Merge_Set() (Lines 3 and 4 in Algorithm IV.1). We denote the cardinality of these two sets by $M_i^{(1)}$ and $M_i^{(2)}$, respectively. Then,

$$M_i = \min\left(M, M_i^{(1)} + M_i^{(2)}\right).$$

The primary check contributes, on average, $M_i^{(1)} = \frac{M_{i-1}}{2}$ distinct vectors from the previous iteration.

We now estimate $M_i^{(2)}$ as the expected number of **unique new vectors** from Merge_Set(). Intuitively, when $M_i' \gg M$, it is unlikely that we will generate the same vector twice or more (besides systematic duplications described in 4.2), and we have a high chance to reach $M_i = M$ new vectors for the next iteration. However, when M_i' gets closer to M as i grows, we are forced to have more duplicates, and Merge_Set() we will not generate $\frac{M}{2}$ new vectors.

As previously discussed, after excluding the obvious obsolete combinations, Merge_Set creates $\frac{3M_{i-1}^2 \cdot q}{8}$ more combinations in iteration i. With the choice of $\delta = \frac{2}{3}$ to ensure that we expect to generate more new vectors than needed. We have an expected number of $\frac{9 \cdot M_{i-1}^2}{16 \cdot M}$ new vectors.

A vector is unique if it is not among the $M_i^{(1)}$ vectors in the first part and not the same as any previously kept one. Hence, the first vector has a probability $1 - \frac{M_i^{(1)}}{2 \cdot M_i'}$ of being unique, and the second vector has a probability larger than $1 -$

$\frac{M_i^{(1)}+1}{2\cdot M_i'}$, and so on. In total, the expected number of unique vectors is estimated at around

$$\frac{9\cdot M_{i-1}^2}{16\cdot M} - \frac{M_i^{(1)} + (M_i^{(1)}+1) + \ldots}{2\cdot M_i'} \approx \frac{9\cdot M_{i-1}^2}{16\cdot M}\left(1 - \frac{M_i^{(1)} + \frac{9\cdot M_{i-1}^2}{32\cdot M}}{2\cdot M_i'}\right).$$

We expect $M_i^{(2)}$ to be the minimum of $M/2$ and the above expression.

Now assume $\mathbf{e}$ is a desired weight-$2p$ vector that fulfills ℓ parity-checks. Then, we know that if $\mathbf{e}$ has appeared in an iteration i, it continues to be present in all subsequent iterations $j \geq i$. Recall that we initialize Sieve_Syndrome_Dec() with a list of size M. The probability that $\mathbf{e}$ is not randomly selected is

$$\left(1 - \frac{1}{M_0'}\right)^M.$$

For $i = 1, \ldots, \ell$, as the primary check does not produce new vectors, then $\mathbf{e}$ is not present after each iteration if it is not produced from Merge_Set(). This routine produces $M_i{}^{(2)}$ more vectors; hence the probability is

$$\left(1 - \frac{1}{2\cdot M_2'}\right)^{M_i^{(2)}},$$

where the factor 2 can be explained as the newly created vectors can be those whose syndromes are either $\mathbf{s}_{[i]}$ or $\mathbf{0}_{[i]}$. Therefore, the probability that $\mathbf{e}$ is not found after Sieve_Syndrome_Dec is

$$\left(1 - \frac{1}{M_0'}\right)^M \cdot \prod_{i=1}^{\ell}\left(1 - \frac{1}{2\cdot M_i'}\right)^{M_i^{(2)}}.$$

In other words, our algorithm finds $\mathbf{e}$ with probability

$$1 - \left(1 - \frac{1}{M_0'}\right)^M \cdot \prod_{i=1}^{\ell}\left(1 - \frac{1}{2\cdot M_i'}\right)^{M_i^{(2)}}.$$

We stress that the quantities above are, for a large part, heuristic estimates for the expected number of vectors in Sieve_Syndrome_Dec(); hence, the provided formulae are not rigorous. In fact, from simulation, we can see that we slightly overestimate $M_i^{(2)}$ and underestimate the actual success probability. However, we can use the expressions to roughly estimate the desired probability for cases where we cannot simulate. If we do that, we observe that the probability typically lies in the range $50-100\%$. In Section 6, we give some examples from implementations

to justify the above heuristic approach.

This probability can be adjusted in two ways. On the one hand, the probability of finding a valid vector is mostly larger than 0.5 if ℓ is large enough. On the other hand, the parity trick in the Gaussian elimination part, explained in [EMZ22], can force the weight of all codewords to be even and then $\mathrm{Pr}_{\text{success}}$ slight increases. In addition, later in Section 4.6, we introduce another trick called the "hybrid approach" that can potentially speed up our algorithm by a factor of 2. Therefore, we adopt the approximation that these two factors cancel each other out.

4.5 Complexity Estimation

We study the complexity in the RAM model, i.e., the cost of reading and writing to one memory address is $\mathcal{O}(1)$ operations, with the memory access cost set to 1. This method is the most traditional way of estimating the complexity, used in many previous papers and also in the complexity estimator given in [EB22b].

Outer iterations

Let us recall that the probability that a permutation yields the correct weight distribution, that is, $2p$ in the first $k+\ell$ bits and $\omega - 2p$ in the remaining $n-k-\ell$ bits, is

$$\mathrm{Pr}_{\text{success}} = \frac{\binom{k+\ell}{2p}\binom{n-k-\ell}{w-2p}}{\binom{n}{w}}.$$

Therefore, we have to perform, on average, $\frac{1}{\mathrm{Pr}_{\text{success}}}$ iterations. We subsequently examine the cost for each iteration, denoted by C_{iter}.

Gaussian elimination

Similarly to recent ISD analysis works [EB22b; EMZ22], we employ the *Method of the four Russians* for Gaussian elimination, which was proposed in [BLP11; BLP08]. There also exists a theoretical analysis [Bar07], along with an open-source version of this method, which was later adopted by Esser et al. [EB22b] for performing the partial Gaussian elimination that is necessary for our framework.

Sieve_Syndrome_Dec()

This routine consists of performing Merge_Set() ℓ times, corresponding to ℓ parity-checks. Let us recap the Merge_Set() subroutine of our ISD algorithm. Assume that a list of size M is sufficient, as stated in Section 4.2. In Algorithm IV.4, we go through the list to check if vectors fulfill the parity-check conditions. For every sample of weight $2p$, the checking corresponds to summing $2p$ bits in the parity-check matrix and the syndrome bit. Therefore, the cost for this step is about

$$C_{\text{check}} = 2p \cdot M.$$

The next step is Algorithm IV.5, which combines samples so that we create another $M/2$ vectors for the next iteration. By parsing $p = p' + p''$, we put our vectors in an ordered table of size $\binom{k+\ell-p''}{p'}$ and distribute our vectors according to their first p' coordinates (in the representation form). This way, when we move our vectors, we only need to read the value of the 'next' p' coordinate and move correspondingly; thus, the cost of moving is constant for each vector. Assume we are at the first 'bucket', i.e., examining all the vectors with 1 in their first p' coordinates. We produce all p'' labels for each vector, and we make use of an array **A** to keep track of how many times the labels have been produced (recall that we index **A** using the p'' labels). We then run through the list of labels that occurred more than once to find the vectors that need to be combined. This routine then ends by moving its content to the next 'bucket'. Note that, for every sample, we do not have to produce labels that include previous coordinates (as those labels are already processed in past buckets). The cost of this step can be broken down into the following parts:

- For each vector, we create precisely $\binom{2p}{p}$ markings on the array **A**. This is the total number of times Lines 11-14 executed for each vector. It consists of two assignments and one comparison. On rare occasions (with probability q), we additionally get collisions to handle. We also include the cost of creating a p'' label (Line 9-10). Assume that the cost of reading and marking each label in **A** is c_{label} operations. Then we need

$$C_{\text{label}} = \binom{2p}{p} \cdot c_{\text{label}} \cdot M,$$

  operations for this part.

- The cost of moving vectors (Line 5). Since the remaining number of coordinates of a vector has to be at least p'', we only have to move a vector $\binom{2p-p''}{p'}$ times. Therefore, moving vectors cost

$$C_{\text{move}} = \binom{2p-p''}{p'} \cdot M.$$

- The cost of combining vectors. In the worst case, we have $2 \cdot M$ collisions, but only $\frac{M}{2}$ new unique vectors are kept (as explained in Section 4.2). For each collision, the cost of producing the new vector is the cost of creating the new $2p$ positions. Colliding positions are known, so it reduces to copying the other positions in an ordered form. We also need to compute the other parts of the vector representation and check for duplicates. This last part corresponds to bit-wise adding two values of bit size slightly larger than $\log M$ and then checking in a hash table if it is a duplicate. It may cost

$2 \log M$ operations.[6] Therefore, this step is estimated to cost

$$C_{\text{combine}} = (2p + 2 \log M) \cdot 2 \cdot M.$$

The Merge_Set() routine is then repeated ℓ times. Thus, if we introduce $C_{\text{Syn\_Dec}}$ as the bit complexity of performing all these steps then

$$C_{\text{Syn\_Dec}} = \left(2p + \binom{2p}{p} \cdot c_{\text{label}} + \binom{2p - p''}{p'} + 4(p + \log M) \right) \cdot \ell \cdot M.$$

Testing candidates

Finally, we have to go through the last list and check for weight-$(\omega - 2p)$ solutions, i.e., via the identity $\omega_H(\mathbf{H}''\mathbf{e} - \mathbf{s}) = \omega - 2p$. This corresponds to adding $2p$ length-$(n - k - \ell)$ columns in $\mathbf{H}''$; moreover, the number of solutions for the exact matching Equation (8) is $\frac{\binom{k+\ell}{2p}}{2^\ell}$. Hence

$$C_{\text{solution\_check}} = 2p \cdot (n - k - \ell) \cdot \frac{\binom{k+\ell}{2p}}{2^\ell}.$$

Theorem 3. *The bit complexity C of the Sieving-Style ISD algorithm is*

$$C = \frac{1}{\text{Pr}_{\text{success}}} \cdot \left(C_{\text{Gauss}} + C_{\text{Synd\_Dec}} + C_{\text{solution\_check}} \right), \tag{17}$$

where C_{Gauss} is the cost of the Gaussian elimination step.

Some observations that decrease the complexity slightly are: For the case of one or a small number of valid solutions, the list size will decrease, and hence, the complexity drops in later iterations. Therefore, these later iterations can be less expensive than the previous ones.

We also note that for 'larger' error weight regimes, for instance, McEliece parameter sets (p is not very small, e.g., $p = 14$ for Category I), then the dominating part of the complexity expression is $(\binom{2p}{p} \cdot c_{\text{label}} \cdot \ell \cdot M)/\text{Pr}_{\text{success}}$.

4.6 A small improvement through a hybrid approach

We propose the idea of a hybrid algorithm first using enumeration to construct a list of weight-$2p$ vectors for Sieving-style ISD. Typically, this will lower the complexity by 1-2 bits.

[6] Here, we assume that the vector representation includes a "key" of bit-length larger than $\log M$. We check if the key is already present, which, in such a case, means that we created a duplicate. When we add two vectors, we also add their keys. The keys can be constructed as a syndrome vector for a random code.

Instead of repeating the Merge_Set() for all ℓ iterations, we split in two parts by $\ell = \ell_1 + \ell_2$. We now directly construct weight-$2p$ vectors that satisfy all first ℓ_1 parity checks by employing a birthday-style algorithm. After that, we continue with our Sieving-style ISD for the ℓ_2 remaining iterations.

First, we create M' random weight-p vectors $\mathbf{e}$, compute the partial syndromes $\mathbf{H}_{[\ell_1]}\mathbf{e}$, $\mathbf{H}_{[\ell_1]}\mathbf{e} + \mathbf{s}_{[\ell_1]}$, and store pairs $(\mathbf{H}_{[\ell_1]}\mathbf{e}, \mathbf{e})$ and $(\mathbf{H}_{[\ell_1]}\mathbf{e} + \mathbf{s}_{[\ell_1]}, \mathbf{e})$ in a table of size 2^{ℓ_1}. If there exist two pairs in an entry of the table, we combine $\mathbf{e} + \mathbf{e}'$ that will satisfy the null-syndrome or $\mathbf{s}_{[\ell_1]}$. The goal of this step is to create M weight-$2p$ vectors for the sieving step, Assume we start with two lists of weight-p vectors of size M', then M' satisfies

$$\frac{(2 \cdot M')^2}{2 \cdot 2^{\ell_1}} \approx M \text{ or } M' \approx \sqrt{M \cdot 2^{\ell_1 - 1}}$$

and to not introduce significantly more memory, we can choose $\ell_1 - 1 \leq \log(M)$ so that $M' \leq M$. The complexity of this step is broken down as

- Computing $2 \cdot M'$ syndromes for the list of size M'

$$2 \cdot M' \cdot p \cdot \ell_1.$$

- Combining M weight-$2p$ vectors

$$M \cdot 2p.$$

Example 4. *Suppose we attack the SDP instance* $(3488, 2720, 64)$*, then the optimized parameters for our algorithm are* $\ell = 83, p = 7$*, and* $\log M = 46$*. The complexity of the inner iteration for this instance is* $C = 2^{65.17}$*. Let* $\ell_1 = 41$*, then* $M' \approx 2^{43}$*. The cost of the new approach is:*

$$2 \cdot M' \cdot p \cdot \ell_1 + M \cdot 2p + \frac{42}{83} \cdot C \approx 2^{64.18}$$

or we improve by 1 bit from this approach.

4.7 Decoding-out-of-many in our sieving-style ISD

In code-based cryptography, the technique of *Decoding-one-out-of-many* (DOOM) [Sen11; JJ02] can significantly enhance the efficiency of previous ISD algorithms for attackers solving a single SDP instance from multiple instances. This approach is especially critical for the key-recovery attack on the HQC cryptosystem and the message-recovery attack on the BIKE system. In the following paragraphs, we sketch how we can, in principle, adapt our new algorithm to benefit from the DOOM technique. However, more work is needed to verify the effect of the modified algorithm in terms of performance and success probability.

The DOOM problems from the key-recovery attack on HQC and the message-recovery attack on BIKE. We aim to search for a specific $2k$-dimensional vector, namely $\mathbf{e} = (\mathbf{e}_0, \mathbf{e}_1)$, which adheres to the condition $\mathbf{He} = \mathbf{s}$. The parity-check matrix of the code employed in HQC and BIKE is given by $\mathbf{H} = (\mathbf{H}_0, \mathbf{H}_1)$, where the matrices $\mathbf{H}_0$ and $\mathbf{H}_1$ are cyclic. In this context, $\mathbf{s}_i$ represents the i-th left cyclic shift of the syndrome $\mathbf{s}$. For each instance of $\mathbf{s}_i$, an $\mathbf{e}$ that satisfies $\mathbf{He} = \mathbf{s}_i$ can be found. The crux of the problem lies in outputting a single vector $\mathbf{e}$ amongst the k potential solutions. The problem is easier than the syndrome decoding problem due to the k potential solutions.

Incorporating DOOM into the Sieving-style ISD algorithm We make minor modifications to the new ISD algorithm for its application in the DOOM context. The DOOM problem appears to aggregate k sub-problems, i.e., creating k types of vectors and progressing to the subsequent iteration. Yet, the new Sieving-style ISD presents a unique characteristic that enables the merging of vectors that have an all-zero syndrome $\mathbf{0}$ across all k categories. By leveraging this feature, we demonstrate that simply doubling the list size is sufficient to maintain a stable list size.

To be specific, in the i-th iteration, we include an index for each vector to distinguish which parity-check equation is satisfied. In the i-th iteration, one has for each $\mathbf{e} \in \mathcal{L}_i$ that

$$\omega_H(\mathbf{e}) = 2p, \text{ and } \mathbf{H}_{[i]}\mathbf{e} \in \{\mathbf{0}, \mathbf{s}_{j,[i]}\} \text{ for } 0 \leq j \leq k-1.$$

Here $\mathbf{s}_{j,[i]}$ represents the j-th left cyclic shift of the syndrome $\mathbf{s}$ restricted to its first i-th positions. We assign an index j to $\mathbf{e}$ to classify the vector $\mathbf{e} \in \mathcal{L}_i$ to $k+1$ categories, where $j = t$ for $0 \leq t \leq k-1$ represents $\mathbf{H}_{[i]}\mathbf{e} = \mathbf{s}_{t,[i]}$ and $j = k$ represents $\mathbf{H}_{[i]}\mathbf{e} = \mathbf{0}$.

Under the continued assumption that the list size is M per iteration—with $M/2$ samples carried forward from the preceding iteration—it becomes crucial to generate $M/2$ fresh samples via collision detection. In each iteration, we assume that $M/2$ samples have the index k and $M/(2k)$ samples have the index t for $0 \leq t \leq k-1$. Let δ as defined in Section 4.1 be the fraction of all combinations that give rise to new vectors.

We create new samples, specifically $\frac{M}{4k}$ of them, carrying indices t where $0 \leq t \leq k-1$. This results in the equation

$$\frac{M}{2} \cdot \frac{M}{2k} \cdot \frac{\delta q}{2} = \frac{M}{4k},$$

yielding the minimum value for M as $\frac{2}{\delta q}$, aligning with the estimate in Equation (13). For the category with index k, we produce new samples, $\frac{M}{4}$ in number, which can be created by merging two vectors with identical indices. Hence, se-

lecting $M = \frac{4}{\delta q}$, we obtain

$$\frac{\delta q \cdot \frac{M^2}{4 \cdot 2}}{2} = \frac{M}{4},$$

where the left side is the new vectors created from merging two vectors with the index k. In this analysis, combinations paired with an index differing from k are ignored due to their less frequent occurrence.

5 Numerical results

In this section, we provide the concrete complexity of our described algorithm when considering some proposed code-based schemes and comparisons with other ISD algorithms.

Our analysis focuses first on the Classic McEliece parameter sets. Subsequently, we examine our algorithm's performance for a set of SDP parameters, referred to as MDPC parameters, akin to those used in QC-MDPC cryptosystems such as BIKE, where noise weights are on the order of $\mathcal{O}(\sqrt{n})$. In this phase, we disregard the quasi-cyclic structure, focusing solely on the complexity of the corresponding SDP problem. Finally, we incorporate the quasi-cyclic structure of BIKE, assessing the bit complexity estimates for BIKE parameters in the context of key-recovery attacks. As for other ISD algorithms, we mainly focus on the MMT and BJMM variants as the bit-complexity can be computed in a similar fashion as in Section 4.5. In particular, we opt not to include Both-May or May-Ozerov variants as it is unclear how the polynomial overhead by using *Nearest-neighbor Search* can affect the concrete complexity.

A comprehensive estimator for the aforementioned ISD algorithms was provided in [EB22b], where the bit complexity is computed as the number of vector operations multiplied by the cost per operation. However, the authors set the cost of each $\mathbb{F}_2$-vector operation as n. This is not the case in our analysis. Therefore, to achieve a fair comparison, our estimator[7] adjusted the cost for each operation for the MMT and BJMM algorithms in a similar manner as in Section 4.5. As a result, Figure 3 and Table 6 yield different optimizations and bit-complexity from [EB22b].

In this section, we examine the security estimates with two values of c_{label} (reading and marking a label into $\mathbf{A}$), namely $c_{\text{label}} = 2$ and $c_{\text{label}} = 5$. The first case, corresponding to a value of 2, represents the optimal scenario and is intended to allow for comparisons with previous works, as the constant in the big $\mathcal{O}(\cdot)$ notation corresponding to using a hash table or similar, is typically set to 1.

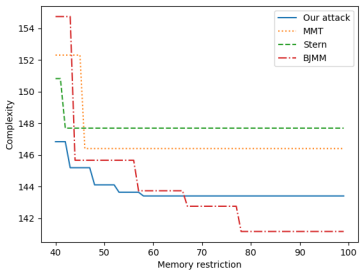
[7] The estimator and optimal parameters can be found in the implementation repository https://github.com/vunguyen95/Review-ISD-Sieving.

The second case, corresponding to a value of 5, reflects our attempt to gauge the impact of increasing this constant value on the complexity.

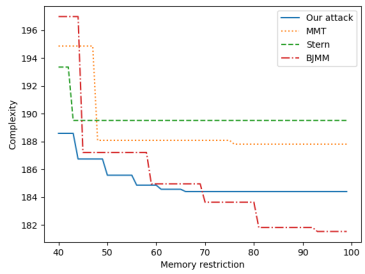
5.1 Numerical results for Classical McEliece

Table 5 provides the security parameter sets of the Classic McEliece cryptosystem. Five parameter sets are published, including one for Category 1, one for Category 3, and three sets for Category 5.

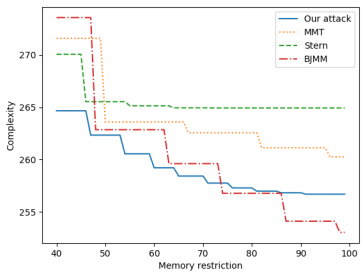
In Table 7, we present the bit security estimates of our new sieving-style ISD algorithm on these Classic McEliece parameter sets. The reference values are from the estimator in [EB22b].



(a) Category 1 ($n = 3488$)



(b) Category 3 ($n = 4608$)



(c) Category 5

Figure 3: Time-memory trade-offs of different ISD (including ours, $c_{\text{label}} = 2$) algorithms.

Figure 3 demonstrates that, under memory constraints, our algorithm outperforms competing ISD algorithms. As detailed in Table 7, our algorithm has improvements to other ISD algorithms for the Classical McEliece security parameters when memory limitation is set to 2^{60} bits. Although it is not competitive with no memory constraint, our algorithm enriches the cryptanalytic arsenal with its different behavior regarding the time-memory trade-off.

With a closer look at the optimal parameters, we notice an interesting fact about our algorithm. With the same memory restriction, the sieving approach can work with larger values of p. For example, for $M < 2^{60}$, the optimal p for

sieving-ISD is $p = 14$ while $p = 8$ for MMT and $p = 12$ for BJMM algorithms. The explanation is that our algorithm only works with a small subset of weight-$2p$ vectors (instead of enumerating all). The iterative 'sieving' steps manipulate this set towards containing the solution(s). This hints that Sieving-style ISD can have more significant improvements in low-memory configurations, which constrains the values of p.

5.2 Applications to MDPC and BIKE (key-recovery) parameter sets

Next, we extend the application of our new algorithm to address MDPC parameters, demonstrating that, in terms of memory efficiency, the Sieving-style ISD algorithm surpasses other ISD algorithms across a set of parameters. Our focus is on SDP parameters where $n = 2k$ and the error weight scales with $\mathcal{O}(\sqrt{n})$. Specifically, as detailed in Table 1, we explore SDP parameters derived from the parameter sets of BIKE and HQC, as listed in Table 6, disregarding their quasi-cyclic structures. It is noteworthy that BIKE and HQC represent two leading code-based KEM proposals in the fourth round of the NIST PQC project, with NIST aiming to potentially standardize one of these code-based KEMs at the conclusion stage of this round. Therefore, analyzing these specific sparse MDPC parameters becomes crucial for cryptographic research.

In the concrete setting, HQC has an even sparser, low-weight error than BIKE. It is a commonly held belief that there have been limited advancements in the enhancement of modern ISD algorithms for sparse parameters as proposed in BIKE and HQC, as evidenced in [EB22b]. It has been shown that the recent ISD methods of BJMM/MMT, Both-May and May-Ozerov have not made a significant improvement to Stern regarding these sparse parameters.

The bit security estimates on the new sieving-like ISD algorithm are shown in Table 1. It is worth pointing out that our approach, while being very competitive with modern ISDs such as BJMM and MMT, uses significantly less memory. In particular, for $(81946, 40973, 264)$ parameter set, sieving ISD and BJMM only differ by $0.5 - 1$ bit in complexity, despite the prior employing 2^{10} less memory.

Memory-efficient behavior is prevalent; different memory-cost models have been examined [EB22b] recently to take into consideration the cost of accessing huge memory, especially in the context of enumeration-based ISD algorithms. Therefore, we believe that, in a practical scenario where memory is a limiting factor, our approach offers a valuable alternative.

As described in [Ara+23], in key-recovery attacks for the BIKE scheme, the quasi-cyclic structure gives us k cyclic shifts of the searched secret key. The bit complexity numbers of a key-recovery attack on BIKE could be reduced by $\log(k)$ bits (from the middle part of Table 1) since BIKE is homogeneous.

Moreover, the value of c_{label} has a minimal effect on the time complexity in contrast to the Classic McEliece scenario. This is primarily due to the fact that the parameter p is set to 3 when solving these highly sparse instances, and thus the

Table 1: Bit security estimates of the MDPC parameters. Here T is the log of the bit complexity and $\hat{M}$ is the log of the memory in bits. The parenthesis notation $(6, 48)$ specifies that $2p = 6$ and $\ell = 48$. We have $p' = 1$. Furthermore, the tuple $(24646, 12323, 134)$ characterizes an SDP instance with parameters $n = 24646$, $k = 12323$, and $\omega = 134$.

	$(24646, 12323, 134)$		$(49318, 24659, 199)$		$(81946, 40973, 264)$	
	T	$\hat{M}$	T	$\hat{M}$	T	$\hat{M}$
Stern	147.9	51	212.4	55	277.2	58
MMT	149.6	57	214.0	62	278.7	66
BJMM	146.6	54	210.9	59	275.7	63
Our ISD, $c_{\text{label}} = 2$	146.7	46	211.4	50	276.2	53
	(6,48)		(6,52)		(6, 55)	
Our ISD, $c_{\text{label}} = 5$	147.1	46	211.8	50	276.6	53
	(6,48)		(6,52)		(6, 55)	
	$(24646, 12323, 142)$		$(49318, 24659, 206)$		$(81946, 40973, 274)$	
	T	$\hat{M}$	T	$\hat{M}$	T	$\hat{M}$
Stern	155.5	51	219.2	55	286.9	58
MMT	157.1	57	220.7	62	288.3	66
BJMM	154.0	54	217.5	59	285.3	63
Our ISD, $c_{\text{label}} = 2$	154.3	46	218.2	50	286.0	53
	(6,48)		(6,52)		(6, 55)	
Our ISD, $c_{\text{label}} = 5$	154.7	46	218.5	50	286.3	53
	(6,48)		(6,52)		(6, 55)	
	$(35338, 17669, 132)$		$(71702, 35851, 200)$		$(115274, 57637, 262)$	
	T	$\hat{M}$	T	$\hat{M}$	T	$\hat{M}$
Stern	147.3	53	214.8	57	276.5	60
MMT	149.6	60	216.8	65	278.4	68
BJMM	146.4	57	213.6	62	275.4	65
Our ISD, $c_{\text{label}} = 2$	146.2	48	213.7	52	275.5	55
	(6,51)		(6,55)		(6, 57)	
Our ISD, $c_{\text{label}} = 5$	146.5	48	214.1	52	275.9	55
	(6,50)		(6,54)		(6, 57)	

cost associated with c_{label} is not the primary contributing factor to the complexity. In addition, we have computed the probability of finding the desired vector by numerical means for this particular example, using the method presented in Section 4.4; our calculation results in an estimated value of more than 50% for the reported attack parameters. We have verified this observation on other parameter sets as well, thereby affirming the soundness of our complexity analysis in conjunction with the hybrid approach in Section 4.6 and the parity-bit trick, which can make up to a factor of 2 needed for the success probability.

6 Simple implementations for Sieve_Syndrome_Dec() with smaller parameters

In this section, we discuss our simple implementation with smaller parameters to support arguments and assumptions that we have made throughout the papers.[8] It is crucial to show in simulation that Sive_Syndrome_Dec() is capable of producing solutions for the exact matching equation as theory predicts, and the parameters such as list size can be sustained. In the implementation, we generate a random target error vector $\mathbf{e}$ and observe whether this vector can be found by Sieve_Syndrome_Dec after ℓ iteration of Merge_Set. Moreover, we set $p' = 1$ in our simulations.

Example 5. *One of many implemented parameter sets is* $k = 300$, $2p = 6$, *i.e.,* $p = 3$. *We set* $p' = 1$, $p'' = 2$ *for the* Merge_Set *algorithm. For the parameter* ℓ, *we choose* $\ell \approx 28$ *(recall Equation (14), we also need* $\frac{M}{2} \geq \frac{\binom{k+\ell}{2p}}{2^\ell}$*) which corresponds to the exponentially many solutions case.*

The probability that the XOR of two weight-6 *vectors results in another weight-*6 *vector is*

$$q = \frac{\binom{2p}{p}\binom{k+\ell-2p}{p}}{\binom{k+\ell}{2p}} \approx 2^{-13.86}.$$

$$M \approx \frac{2}{\delta q} \approx \delta^{-1} \cdot 2^{14.86}.$$

As explained in Section 4.4, we increase M *with a factor* $\delta^{-1} \approx 3/2$ *so that we can keep the list size relatively constant for the majority of iterations (until* Merge_Set *can not produce* $\frac{M}{2}$ *new vectors). Hence, we select* $M = 2^{15.44}$.

We run the implementation 10^2 *times and find* $\mathbf{e}$ *in* 60 *runs, i.e., a* 60% *success rate.*

Table 2: Comparison between the heuristic arguments and the actual implementation for $k = 300$, $\ell = 28$, $p = 3$, in terms of each iteration memory M_i (in log) and success probability (S.P).

Iteration	1	...	14	15	16	...	25	26	27	28
M_i (pred.)	15.46	...	15.46	15.46	15.46	...	15.44	14.80	13.96	13.09
M_i (impl.)	15.46	...	15.46	15.46	15.46	...	15.22	14.71	13.91	12.96
S.P (pred.)	0	...	$22 \cdot 10^{-5}$	0.00045	0.0009	...	0.363	0.441	0.477	0.507

We note that any sufficiently large enough ℓ *can be chosen. As an example, in the case where* $\ell = 50$ *(i.e., on average, only one solution), our algorithm still finds the target* $\mathbf{e}$ *with promising probability (> 50%). It can also be inferred from Table 2 that*

[8] Implementation repository https://github.com/vunguyen95/Review-ISD-Sieving.

one can choose ℓ in order to raise the success probability to a desired range as claimed in Section 4.4.

Example 6. *It is also of interest to see how our implementation fares with larger instance of k (e.g., close to the medium-sized instance of McEliece). In particular, we proceed with $k = 1000$ and $2p = 4$. We choose a smaller values of p to have a manageable memory requirement for a commercial computer. The following numerical values are derived in the same manner as in Example 1.*

For $\ell = 27$, it gives $M \approx 2^{15.42}$. A target vector $\mathbf{e}$ is found in 56 out of 10^2 tests, i.e., a 56% success probability.

Table 3: Comparison between the heuristic arguments and the actual implementation for $k = 1000$, $\ell = 27$, $p = 2$, in terms of each iteration memory M_i (in log) and success probability (S.P).

Iteration	1	...	20	21	22	23	24	25	26	27
M_i (pred.)	15.42	...	15.35	14.64	13.82	12.94	12.01	11.05	10.08	9.09
M_i (impl.)	15.42	...	14.97	14.32	13.46	12.50	11.51	10.49	9.50	8.41
S.P (pred.)	0	...	0.380	0.442	0.482	0.511	0.529	0.539	0.545	0.548

Discussions We have observed that the actual implementation results are comparable to or even surpass the estimation results obtained using the method described in Section 4.4. Moreover, in Table 4, we present the evolution of the estimated list size and estimated success probability over the course of various iterations, utilizing an attack instance on Classic McEliece as reported in Table 7. In both our theoretical calculations and empirical investigations, we have identified a critical juncture, referred to as a '*breaking point*', which corresponds to the iteration at which the list size of M_i begins to decrease. While the initial decline is gradual, it gains momentum as subsequent iterations progress.

One favorable aspect in this iterative process is that upon reaching the 'breaking point', the success probability becomes non-negligible and quickly rises above 50%. Subsequent iterations will result in a further reduction of the list size, leading to a slower increase in the success probability in finding the targeted vector.

We have observed that the attack instances reported in the previous section all select the parameter ℓ several iterations after the occurrence of the 'breaking point', thereby guaranteeing a success probability exceeding 50%. Additionally, our experiments demonstrate that, for a choice of ℓ close to the 'breaking point', the actual list size is consistent with the theoretical estimation and the observed success probability meets (or even surpasses) the estimated value.

7 Concluding remarks

We have presented a novel Sieving-style Information-set Decoding algorithm for solving the syndrome decoding problem and made a heuristic analysis. The al-

Table 4: The predicted memory (in log) and success probability (S.P) for attacking a Classic McEliece instance with $k = 3360, \ell = 96, p = 8$.

Iteration	1	...	89	90	91	92	93	94	95	96
M_i (pred.)	53.03	...	53.03	52.93	52.60	51.99	51.18	50.28	49.34	48.37
S.P (pred.)	0	...	0.138	0.242	0.358	0.441	0.485	0.509	0.523	0.529

gorithm makes advancements to state-of-the-art algorithms when complexity is considered in the RAM model and is characterized by its time-memory trade-off profile. For instance, in the McEliece cryptographic scheme, an attack using the algorithm achieves lower complexity when the memory is limited (for example, 2^{60} bits). Interestingly, it was also shown that the low-weight regime (as in MDPC parameters or in constructions such as BIKE) benefits our algorithm compared to the state-of-the-art. In particular, we achieve very comparable performance with significantly less memory. This finding is of great interest as the advantage of enumeration-based ISD variants is often compromised by the huge memory requirements.

Besides the described algorithm, many other versions of the algorithms can be considered. For instance, we can amend the problem of duplicates by only combining vectors in the first few iterations and including the checking routine later. The motivation is that the correct error vector will not likely be created in early iterations, and combining vectors does not result in noticeable dependencies between vectors. Moreover, in specific settings, such as BIKE and HQC, where the optimal value of p is small and the memory requirement is not high, more efficient implementation could be achieved.

Lastly, we note that accelerating the new ISD algorithm using sophisticated instruction sets such as AVX-256 in practical software design seems non-trivial. Further exploration of this intriguing topic and actual, full-scaled implementations of concrete parameters of code-based schemes are left for future endeavors.

References

[AB09] S. Arora and B. Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[AFS05] D. Augot, M. Finiasz, and N. Sendrier. "A Family of Fast Syndrome Based Cryptographic Hash Functions". In: *Progress in Cryptology - Mycrypt 2005, First International Conference on Cryptology in Malaysia, Kuala Lumpur, Malaysia, September 28-30, 2005, Proceedings*. Ed. by E. Dawson and S. Vaudenay. Vol. 3715. Lecture Notes in Computer Science. Springer, 2005, pp. 64–83.

[AKS01a] M. Ajtai, R. Kumar, and D. Sivakumar. "A sieve algorithm for the shortest lattice vector problem". In: *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*. STOC '01. Hersonissos, Greece: Association for Computing Machinery, 2001, pp. 601–610.

[AKS01b] M. Ajtai, R. Kumar, and D. Sivakumar. "A sieve algorithm for the shortest lattice vector problem". In: *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. 2001, pp. 601–610.

[Ara+23] N. Aragon et al. "BIKE: Bit-flipping key encapsulation". In: *NIST PQC* (2023).

[Bal+19] M. Baldi et al. "A Finite Regime Analysis of Information Set Decoding Algorithms". In: *Algorithms* 12.10 (2019), p. 209.

[Bar07] G. Bard. "Algorithms for solving linear and polynomial systems of equations over finite fields with application to cryptanalysis." PhD thesis. Faculty of the Graduate School of the University of Maryland, College Park, 2007.

[Bec+12a] A. Becker et al. "Decoding Random Binary Linear Codes in 2 n/20: How 1 + 1 = 0 Improves Information Set Decoding". In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 520–536.

[Ber+11] D. J. Bernstein et al. "Really Fast Syndrome-Based Hashing". In: *Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings*. Ed. by A. Nitaj and D. Pointcheval. Vol. 6737. Lecture Notes in Computer Science. Springer, 2011, pp. 134–152.

[Ber+20] D. J. Bernstein et al. "Classic McEliece: conservative code-based cryptography". In: *NIST submissions* (2020).

[BGJ13] A. Becker, N. Gama, and A. Joux. "Solving shortest and closest vector problems: The decomposition approach". In: *IACR Cryptol. ePrint Arch.* (2013), p. 685.

[BGJ14] A. Becker, N. Gama, and A. Joux. "A sieve algorithm based on overlattices". In: *LMS Journal of Computation and Mathematics* 17.A (2014), pp. 49–70.

[BLP08] D. J. Bernstein, T. Lange, and C. Peters. "Attacking and Defending the McEliece Cryptosystem". In: *Post-Quantum Cryptography, Second International Workshop, PQCrypto 2008, Cincinnati, OH, USA, October 17-19, 2008, Proceedings*. Ed. by J. Buchmann and J. Ding. Vol. 5299. Lecture Notes in Computer Science. Springer, 2008, pp. 31–46.

[BLP11] D. J. Bernstein, T. Lange, and C. Peters. "Smaller Decoding Exponents: Ball-Collision Decoding". In: *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*. Ed. by P. Rogaway. Vol. 6841. Lecture Notes in Computer Science. Springer, 2011, pp. 743–760.

[BM17a] L. Both and A. May. "Optimizing BJMM with nearest neighbors: full decoding in 22/21n and McEliece security". In: *WCC workshop on coding and cryptography*. 2017, p. 214.

[BM18] L. Both and A. May. "Decoding Linear Codes with High Error Rate and Its Impact for LPN Security". In: *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*. Ed. by T. Lange and R. Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 25–46.

[BMT78b] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)" In: *IEEE Trans. Inf. Theory* 24.3 (1978), pp. 384–386.

[Car+22] K. Carrier et al. "Statistical Decoding 2.0: Reducing Decoding to LPN". In: *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV*. Ed. by S. Agrawal and D. Lin. Vol. 13794. Lecture Notes in Computer Science. Springer, 2022, pp. 477–507.

[CFS01] N. T. Courtois, M. Finiasz, and N. Sendrier. "How to Achieve a McEliece-Based Digital Signature Scheme". In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. Ed. by C. Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 157–174.

[Duc+23] L. Ducas et al. *Asymptotics and Improvements of Sieving for Codes*. Cryptology ePrint Archive, Paper 2023/1577. `https://eprint.iacr.org/2023/1577`. 2023.

[Dum91] I. Dumer. "On minimum distance decoding of linear codes". In: *Proc. 5th Joint Soviet-Swedish International Workshop Information Theory*. 1991, pp. 50–52.

[EB22b] A. Esser and E. Bellini. "Syndrome Decoding Estimator". In: *Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part I*. Ed. by G. Hanaoka, J. Shikata, and Y. Watanabe. Vol. 13177. Lecture Notes in Computer Science. Springer, 2022, pp. 112–141.

[EMZ22] A. Esser, A. May, and F. Zweydinger. "McEliece Needs a Break - Solving McEliece-1284 and Quasi-Cyclic-2918 with Modern ISD". In: *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III*. Ed. by O. Dunkelman and S. Dziembowski. Vol. 13277. Lecture Notes in Computer Science. Springer, 2022, pp. 433–457.

[FS09] M. Finiasz and N. Sendrier. "Security Bounds for the Design of Code-Based Cryptosystems". In: *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*. Ed. by M. Matsui. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 88–105.

[FS96] J. Fischer and J. Stern. "An Efficient Pseudo-Random Generator Provably as Secure as Syndrome Decoding". In: *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*. Ed. by U. M. Maurer. Vol. 1070. Lecture Notes in Computer Science. Springer, 1996, pp. 245–255.

[HS13] Y. Hamdaoui and N. Sendrier. "A non asymptotic analysis of information set decoding". In: *Cryptology ePrint Archive* (2013).

[JJ02] T. Johansson and F. Jönsson. "On the complexity of some cryptographic problems based on the general decoding problem". In: *IEEE Trans. Inf. Theory* 48.10 (2002), pp. 2669–2678.

[LB88] P. J. Lee and E. F. Brickell. "An Observation on the Security of McEliece's Public-Key Cryptosystem". In: *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*. Ed. by C. G. Günther. Vol. 330. Lecture Notes in Computer Science. Springer, 1988, pp. 275–280.

[Leo88] J. S. Leon. “A probabilistic algorithm for computing minimum weights of large error-correcting codes”. In: *IEEE Trans. Inf. Theory* 34.5 (1988), pp. 1354–1359.

[Mel+23a] C. A. Melchor et al. “Hamming quasi-cyclic (HQC)”. In: *NIST PQC* (2023).

[MMT11a] A. May, A. Meurer, and E. Thomae. “Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$”. In: *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*. Ed. by D. H. Lee and X. Wang. Vol. 7073. Lecture Notes in Computer Science. Springer, 2011, pp. 107–124.

[MO15] A. May and I. Ozerov. “On Computing Nearest Neighbors with Applications to Decoding of Binary Linear Codes”. In: *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*. Ed. by E. Oswald and M. Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 203–228.

[NIS] NIST. *NIST Post-Quantum Cryptography Standardization*. `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization`. Accessed: 2022-11-30.

[NV08] P. Q. Nguyen and T. Vidick. “Sieve algorithms for the shortest vector problem are practical”. In: *Journal of Mathematical Cryptology* 2.2 (2008), pp. 181–207.

[Pra62] E. Prange. “The use of information sets in decoding cyclic codes”. In: *IRE Trans. Inf. Theory* 8.5 (1962), pp. 5–9.

[Sen11] N. Sendrier. “Decoding One Out of Many”. In: *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*. Ed. by B. Yang. Vol. 7071. Lecture Notes in Computer Science. Springer, 2011, pp. 51–67.

[Ste88] J. Stern. “A method for finding codewords of small weight”. In: *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*. Ed. by G. D. Cohen and J. Wolfmann. Vol. 388. Lecture Notes in Computer Science. Springer, 1988, pp. 106–113.

[Ste93] J. Stern. "A New Identification Scheme Based on Syndrome Decoding". In: *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*. Ed. by D. R. Stinson. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 13–21.

[TS16] R. C. Torres and N. Sendrier. "Analysis of Information Set Decoding for a Sub-linear Error Weight". In: *Post-Quantum Cryptography*. 2016.

[Vér96] P. Véron. "Improved identification schemes based on error-correcting codes". In: *Appl. Algebra Eng. Commun. Comput.* 8.1 (1996), pp. 57–69.

Supporting Tables.

Table 5: Security parameters of the Classical McEliece scheme.

Category	n	k	ω
1	3488	2720	64
3	4608	3360	96
5	6688	5024	128
5	6960	5413	119
5	8192	6528	128

Table 6: BIKE and HQC security parameters.

	Category	n	k	w
BIKE (message)	1	24646	12323	134
	3	49318	24659	199
	5	81946	40973	264
BIKE (key)	1	24646	12323	142
	3	49318	24659	206
	5	81946	40973	274
HQC	1	35338	17669	132
	3	71702	35851	200
	5	115274	57637	262

Table 7: Bit security estimates of the Classic McEliece scheme. Here T is the log of the bit complexity, and $\hat{M}$ is the log of memory in bits. The parenthesis notation $(14, 83, 3)$ specifies that $2p = 14$ and $\ell = 83, p' = 3$.

	Category 1 ($n = 3488$)		Category 3 ($n = 4608$)		Category 5 ($n = 6688$)		Category 5 ($n = 6960$)		Category 5 ($n = 8192$)	
	T	$\hat{M}$	T	$\hat{M}$	T	$\hat{M}$	T	$\hat{M}$	T	$\hat{M}$
Stern	147.7	41	189.5	43	264.9	72	265.5	73	300.4	92
MMT	146.6	45	187.9	76	260.0	109	260.0	123	292.9	114
BJMM	141.2	77	180.6	101	250.4	109	250.2	110	283.4	141
$\hat{M} \leq 60$	143.9	56	185.1	59	263.1	48	263.8	48	299.4	49
Our ISD, $\hat{M} \leq 60$										
$c_{\text{label}} = 2$	143.4	58	184.8	55	259.2	59	259.8	60	296.6	55
$c_{\text{label}} = 5$	144.6	58	186.0	55	260.4	59	261.0	60	297.6	55
	(14,83,3)		(12,75,2)		(12,78,2)		(12,79,2)		(10,68,2)	
Our ISD, any $\hat{M}$										
$c_{\text{label}} = 2$	143.4	58	184.4	65	256.7	91	256.8	92	290.6	95
$c_{\text{label}} = 5$	144.6	58	185.7	65	258.0	91	258.1	92	291.9	95
	(14,83,3)		(16,96,3)		(24,144,5)		(24,146,5)		(24,149,5)	

Efficient Authentication Protocols from the Restricted Syndrome Decoding Problem

Abstract

In this paper, we introduce an oracle version of the Restricted Syndrome Decoding Problem (RSDP) and propose novel authentication protocols based on the hardness of this problem. They follow the basic structure of the HB-family of authentication protocols and later improvements but demonstrate several advantages.

An appropriate choice of multiplicative subgroup and ring structure gives rise to a very efficient hardware implementation compared to other *Learning Parity with Noise* based approaches. In addition, the new protocols also have lower key size, lower communication costs, and potentially better completeness/soundness compared to learning-based alternatives. This is appealing in the context of low-cost, low-powered authenticating devices such as radio frequency identification (RFID) systems. Lastly, we show that with additional assumptions, RSDP can be used to instantiate a Man-in-the-Middle secured authentication protocol.

1 Introduction

Authentication serves as one of the foundational pillars of cryptography, playing a crucial role in securing communication and data integrity. As technology

Thomas Johansson, Mustafa Khairallah, Vu Nguyen. "Efficient Authentication Protocols from the Restricted Syndrome Decoding Problem". In *IEEE European Symposium on Security and Privacy 2025 (EuroS&P), Venice, Italy*.

advances, particularly with the widespread adoption of low-cost RFID tags, the demand for lightweight yet robust authentication protocols has intensified. Due to the efficient nature of using only simple operations, *Learning parity with Noise* (LPN) emerged as a prime candidate for constructing lightweight identification protocol. The LPN problem does not only stand out under the "practical" but also from the theoretical viewpoint. It is closely related to the well-studied *decoding a random linear code* problem or the equivalent *syndrome decoding* problem, and they are both strongly believed to resist known quantum attacks.

However, traditional approaches often struggled to balance efficiency and security, prompting an ongoing search for solutions catering to resource-constrained environments without compromising security. For example, early attempts at employing LPN, e.g., [HB01a; JW05a; DK07; MP07; BC08; GRS08c], were met with security issues when examined under the scopes of different, more advanced adversarial models, such as in [GRS05; GRS08a; OOV08]. Numerous attempts to improve and explore other (yet still related to the LPN-based) directions have been proposed in recent years [LGQ13a; Kil+11; LM13a]. They appear to have achieved some level of security while still promising efficiency.

Hopper and Blum [HB01a] laid the foundation for LPN-based authentication with their strikingly simple 2-round design, called HB. It provides provable security under *passive attack*, in which an adversary can only eavesdrop on the communications. It was noted by Juels and Weis [JW05a] that an active attacker (i.e., with query power) could easily break HB, and they proposed an augmented 3-round version called HB+. HB+ was shown to be susceptible to attacks in the Man-in-the-Middle attack model (MitM) [GRS05]. Gilbert et al. [GRS08c] proposed HB# to resist the attack from [GRS05]. To make the proposal suitable for low-cost hardware, the authors also proposed using $\mathbf{X}$ as a *Toepliz* matrix, which implies a slightly different LPN hardness assumption. Unfortunately, in a more general model of MitM adversary, HB# was shown to be not sufficient [OOV08].

LPN-based MitM-secured authentication was finally achieved with a series of breakthrough works: Kiltz et al. [Kil+11] proposed a variant of LPN called *Subset-LPN* and built efficient MACs based on this problem. Notably, they also achieve ϵ security (compared to $\sqrt{\epsilon}$ for other LPN-based protocols, excluding HB) by avoiding *rewinding* in their security proof. One of the drawbacks is a large key size that can be prohibitive in some cases. While HB# can circumvent this problem by using "structural" LPN (e.g., employing a Toepliz matrix) [GRS08c], it is unfortunately not the case for [Kil+11]. Li et al. in [LGQ13a] showed an interesting design called LCMQ without constructing MACs by applying a chosen ciphertext secure encryption on top of the LPN samples. To reduce communication and the computation burden on the low-cost tag, the encryption scheme uses the so-called P2-circulant matrix. By masking the LPN samples, they proposed aggressive parameters for their claim security, thus demonstrating the efficiency of the design. However, Nguyen et al. [NJG24] have recently devised a key-recovery attack on LCMQ, which could significantly compromise the performance by forcing bigger

key sizes.

Further advancement came when Lyubashevsky and Masny showed how to generically build a MitM-secured authentication from any (randomized) weak pseudo-random functions (wPRFs), which only has to fulfill a few (reasonable) properties [LM13a]. Therefore, LPN can be seen as an instantiation of this design. Other well-studied LPN-type assumptions were employed to achieve manageable key sizes and efficiency required for low-cost environments, such as *Ring-LPN* [Hey+12] and *Toepliz-LPN* [GRS08b]. Despite the tightness of the security proof being "only" $\sqrt{\epsilon}$, their proposal remains impressive as it is more efficient than MACs construction like [Kil+11]. Moreover, it is unclear how the gap in the security proof affects instances for practical uses.

Similarly to LPN, the *Restricted Syndrome Decoding Problem* (RSDP) is another NP-hard variant of SDP. It was first proposed by Baldi et al. in [Bal+20a] as a new research direction toward efficient code-based zero-knowledge identification. It has recently been featured in one of the post-quantum signature candidates, CROSS [Bal+24c], in the ongoing NIST additional call for signatures.

Motivation

LPN-based authentication protocols are, in general, attractive in lightweight applications but suffer from drawbacks, such as large key size, a large number of rounds, etc., since parameters have to be quite large to attain sufficient security. Our work is motivated by the belief that changing the underlying problem from LPN to RSDP may render several advantages, both in security and performance. From a more theoretical perspective, the modified Oracle RSDP is analogous to the situation between the LPN and the Syndrome Decoding Problem. Thus, we aim to diversify the applications of the novel RSDP problem towards lightweight cryptography.

Contributions

In this work, we introduce an Oracle version of the RSDP and then we propose novel authentication protocols based on the hardness of this problem. We first explore the idea of building a highly efficient RSDP-based authentication protocol that is secure in the *active attack* model. By selecting the secret keys from a suitable restricted set, multiplication in the finite field can be as simple as a cyclic shift, which can be performed very efficiently even with low-cost RFID tags. Our designs also have smaller key sizes, lower communication costs, and potentially better completeness/soundness compared to HB-like protocols based on LPN or similar problems. On the theoretical side, restricting the secret keys in such a fashion does not compromise the hardness of RSDP. Moreover, relying on the best cryptanalytic tools for RSDP [Bal+24c], we show, for example, that the key size for 128-bit security can be as low as 396-bit compared to 896 for LPN-based constructions. In Table 1, we provide a head to head comparison between these two protocols,

demonstrating that not only our protocol has small key size, but also a few orders of magnitude lower communication bandwidths and significantly smaller area. Besides, our protocol offers a speed-area trade off that can further lower the area at the expense of speed, a trade-off that is not feasible with LPN protocols; reducing their speed does not significantly reduce the area. We further propose another stronger design that is proven to be secure in the MitM model. The cost is larger keys and more costly implementation, but still, the design compares favorably with other protocols secure in the MitM model.

Table 1: Head to head comparison between HB+ protocols based on LPN vs RSDP for a 128-bit security level. BW: Bandwidth. GE: Gate Equivalents. The implementation estimates are based on the ASIC synthesis results presented in Table 6.

Assumption	**Security**	**Total BW**	**#Cycles/Round**	**Total #Cycles**	**Key Size**	**Area** (GEs)
LPN	128	1,023,447	3	3492	896	19428.97
RSDP	128	30,135	3∼105	105∼ 4305	396	13249.47∼7304.22

Organization

In Section 2, we give the basics on authentication protocols, coding theory, LPN, and RSDP. We introduce some new definitions in relation to RSDP. In Section 3, we give the basic RSDP authentication protocol with security against active attacks. The proof of security is given in the same section. Section 4 is devoted to parameter instantiation and performance evaluation, including some comparisons based on FPGA and ASIC implementations. Section 5 then gives an extended protocol that is secure in the MitM and includes proof of this fact.

2 Preliminaries

Throughout the paper, we use the notation

- $a, \mathbf{a}, \mathbf{A}$ for single elements, vectors, and matrices, respectively.
- $\langle \cdot, \cdot \rangle$ the inner products of two vectors.
- $\mathbf{I}_n$ the identity matrix of size $n \times n$.
- $\mathbb{F}_p$ the finite field of order p, where p is a prime and $\mathbb{F}_p^n$ is the corresponding vector space of dimension n.
- $a \xleftarrow{\$} A$ an element drawn uniformly at random from some set A.
- $\omega_H(\mathbf{x})$ the Hamming weight of a vector $\mathbf{x} \in \mathbb{F}_p^n$, defined as the number of its non-zero coordinates.

2.1 Authentication Protocol

We are interested in an interactive authentication protocol where we define two entities: a *Tag* (a.k.a Prover, denoted by $\mathcal{T}$) and a *Reader* (a.k.a Verifier, denoted by $\mathcal{R}$). Together, they share secret keys, which have been communicated via a secure channel before the authentication begins. Moreover, they are also parameterized by other (public) values, such as the length of the secret key, the domain, and so on. They interact on an insecure communication channel. After the interaction, $\mathcal{R}$ either outputs accept or reject. An authentication protocol is said to have a completeness error P_c if $\mathcal{R}$ rejects a legitimate $\mathcal{T}$ with probability at most P_c. Conversely, the protocol is said to have a soundness error P_s if $\mathcal{R}$ accepts a random response from $\mathcal{T}$ with probability at most P_s.

For security notions, we consider the most common adversary models: *passive attack*, *active attack*, and *Man-in-the-middle* attack [JW05a; GRS05; OOV08; LGQ13b].

2.2 The security game of an authentication protocol

For simplicity, we use the 3-round model (*commit, challenge, response*) for the interaction between $\mathcal{T}$ and $\mathcal{R}$. They share s as secret(s). The protocol is described as: the $\mathcal{T}$ initiates by sending the commit b to $\mathcal{R}$, then $\mathcal{R}$ replies with the challenge a. A response z is produced by $\mathcal{T}$ and $\mathcal{R}$ outputs accept or reject. We characterize the interaction with the transcript (a, b, z). An adversary $\mathcal{A}$, who has access to $\mathcal{T}$ and $\mathcal{R}$, operates in 2-phases.

- Phase 1 (Interaction): The adversary $\mathcal{A}$ invokes $\mathcal{T}$ and $\mathcal{R}$. It stands between the communication of $\mathcal{T}$ and $\mathcal{R}$.
- Phase 2 (Impersonation): $\mathcal{A}$ interact with $\mathcal{R}$. If $\mathcal{A}$ sends a commit $\hat{a}$, received $\hat{b}$ from $\mathcal{R}$, then produce a response $\hat{z}$.

The adversary $\mathcal{A}$ "wins" the above security games if $(\hat{a}, \hat{b}, \hat{z}) \neq (0, 0, 0)$ and $\mathcal{R}$ outputs accept in Phase 2. Generally, we assume that $\mathcal{A}$ only has one chance to convince $\mathcal{R}$ as a legitimate $\mathcal{T}$. The behavior in Phase 1 of $\mathcal{A}$ in the passive, active, MitM security game is described in the following figures (they are identical in Phase 2).

Passive attack

A passive attacker $\mathcal{A}$ can only observe interactions between $\mathcal{T}$ and $\mathcal{R}$ (Figure 1).

Active attack

Here, $\mathcal{A}$ is given *query power* in Phase 1. Specifically, $\mathcal{A}$ can choose a in (a, b, z) to interrogate $\mathcal{T}$ (Figure 2).

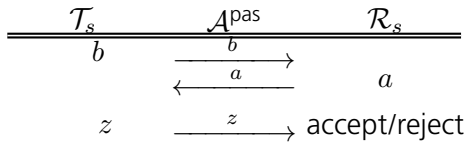


Figure 1: Phase 1 of a passive attack.

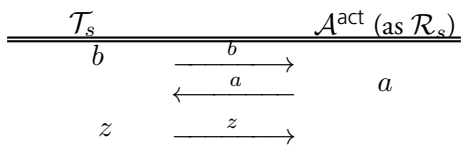


Figure 2: Phase 1 of an ative attack.

Man-in-the-Middle attack

In this scenario, $\mathcal{A}$ can modify (a, b, z) before forwarding to either $\mathcal{T}$ or $\mathcal{R}$. We represent the interference as (a', b', z') (Figure 3).

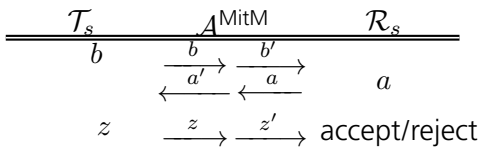


Figure 3: Phase 1 of an MitM attack.

Definition 13 (Security of an authentication protocol)**.** *An authentication protocol is said to be (t, Q, ϵ)-secure in the X-model (X is either passive, active, or MitM) if for all X-adversaries $\mathcal{A}$, running in time t, making Q queries with $\mathcal{T}$, the probability of $\mathcal{R}$ outputs* accept *in Phase 2 is at most ϵ. In particular, we have*

$$\Pr[(\mathcal{A}^X, \mathcal{T}, \mathcal{R}) = \text{accept}] - P_s \leq \epsilon.$$

2.3 Coding theory

Let p be a prime number and $\mathbb{F}_p$ be a finite field. A $[n, k]$-linear code $\mathcal{C}$ over $\mathbb{F}_p$ ($k \leq n$) is a vector subspace of dimension k in $\mathbb{F}_p^n$. A full-rank matrix $\mathbf{G} \in \mathbb{F}_p^{k \times n}$ is said to be the generator of $\mathcal{C}$ if $\mathcal{C} = \{\mathbf{uG} : \mathbf{u} \in \mathbb{F}_p^k\}$, i.e., $\mathbf{G}$ is a basis of $\mathcal{C}$. Let $\mathbf{H} \in \mathbb{F}_p^{(n-k) \times n}$ be a matrix such that $\mathbf{GH}^\intercal = \mathbf{0}$. Then $\mathbf{H}$ is called the *parity-check* matrix of $\mathcal{C}$ and for $\mathbf{y} \in \mathbb{F}_p^n$, $\mathbf{s} = \mathbf{yH}^\intercal$ is called its *syndrome* (w.r.t $\mathbf{H}$). As $\mathbf{G}$ is full-rank, it is sometimes convenient to assume $\mathbf{G}$ to be in its *systematic form*,

that is $\mathbf{G} = \begin{pmatrix}\mathbf{I}_k & \mathbf{A}\end{pmatrix}$ for some $\mathbf{A} \in \mathbb{F}_p^{k\times(n-k)}$. We can then readily compute the parity-check matrix, also in systematic form, as $\mathbf{H} = \begin{pmatrix}-\mathbf{A}^\intercal & \mathbf{I}_{n-k}\end{pmatrix}$.

Problem 5 (Syndrome Decoding Problem (SDP)). *Given* $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$*, a syndrome* $\mathbf{s} \in \mathbb{F}_p^{n-k}$*, and a positive integer* $t \leq n$*. Find* $\mathbf{e}$ *(if any) where* $\omega_H(\mathbf{e}) \leq t$ *such that* $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$*.*

It is a well-studied NP-complete [BMT78b], highlighted by the fact that it has been a crucial building block in many cryptosystems, such as zero-knowledge proof [Ste93; Vér96], signatures [CFS01], hash functions [AFS05; Ber+11], stream ciphers [FS96], and so on. In particular, for post-quantum cryptography, it has been popular and appeared in many public key cryptosystems (such as McEliece [Ber+20], BIKE [Ara+21], HQC [Mel+23a]), as well as the recent NIST call for signatures (CROSS [Bal+24c], SDitH [Mel+23b], Wave [Ban+23], and so on).

One also finds many variants of SDP, such as the *Restricted Decoding Problem*, *Regular Decoding Problem*, *Permuted Kernel Problem*, …, or the SDP defined with different metrics (e.g., rank metric, Lee metric), that all prove useful in constructing cryptographic primitives. This work involves employing the *Restricted Decoding Problem* (RSDP) in lightweight authentication protocols.

Let $\mathbb{F}_p$ be a finite field where p is a prime, and let $\mathbb{E} = \{g^i, i = 0, \ldots, z\}$ be a multiplicative subgroup of order z, generated by some element $g \in \mathbb{F}_p$.

Problem 6 (Restricted Syndrome Decoding Problem). *Given* $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_p^{n-k}$*. Find* $\mathbf{e} \in \mathbb{E}^n$ *such that* $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$*.*

The RSDP was first introduced in [Bal+20a] for $z = 2$. Moreover, the original RSDP asks for non-full weight $\mathbf{e}$, that is, entries of $\mathbf{e}$ are sampled from $\mathbb{E}_0 := \mathbb{E} \cup \{0\}$ and $\omega_H(\mathbf{e}) \leq t$ for some $t \in \mathbb{N}$. Notably, via a reduction to the classical SDP, they showed that this new problem is also NP-complete. As an application, the authors proposed a zero-knowledge identification scheme (adaptation by replacing SDP with RSDP), which is promising in terms of reduced public key size and communication cost. Recently, the versatility of RSDP has been extended by Baldi et al. [Bal+24c], where the order z is no longer restricted to $z = 2$. In addition, CROSS - a digital signature scheme based on maximum-weight RSDP (as defined in Problem 6) was proposed.[1] Such a modification does not compromise the problem hardness [Bal+24a], and contrarily to Hamming-weight RSDP, the uniqueness of the solution is still guaranteed.

For our application (in later sections), we rely on the security of RSDP in the "oracle form". We thus define the so-called RSDP Oracle, which will later be used in our construction.

[1] A specialized version of RSDP, called RSDP(G) was also investigated in the same work.

Definition 14 (RSDP-Oracle). *Let p be a prime number and $\mathbb{E} = \{g^i, i = 1, \ldots, z\}$ be a multiplicative subgroup of order z in $\mathbb{F}_p$. Fix a secret $\mathbf{s} \in \mathbb{F}_p^k$. The RSDP Oracle, denoted by $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$, gives pairs of samples*

$$\{\mathbf{a} \in \mathbb{F}_p^k, b = \langle \mathbf{a}, \mathbf{s} \rangle + e \bmod p\}$$

where $\mathbf{a} \xleftarrow{\$} \mathbb{F}_p^k$ and $e \xleftarrow{\$} \mathbb{E}$.

We write $\Lambda_k(\mathbf{s})$ for the distribution over $\mathbb{F}_p^k \times \mathbb{F}_p$, where the samples are obtained by querying $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$. The *decisional* RSDP, written as ORSDP_k, is to distinguish the $\Lambda_k(\mathbf{s})$ samples from the uniform distribution U_{k+1}. On the other hand, the *search* version asks for the recovery of the secret $\mathbf{s}$.

Definition 15. *The ORSDP_k is said to be (t, Q, ϵ)-hard if for every algorithm D running in time t, making Q oracle queries*

$$\left| \Pr[\mathbf{s} \xleftarrow{\$} \mathbb{F}_p^k : D^{\Lambda_k(\mathbf{s})} = 1] - \Pr[D^{U_{k+1}} = 1] \right| \leq \epsilon,$$

where we denote D^X by the algorithm D taking oracle input from a distribution X.

Next, we informally argue that the decisional RSDP problem is **as hard as** the search version via simple reduction. Indeed, suppose there exists some distinguisher D that can solve ORSDP_k with non-negligible probability ϵ. It can then be used to recover $\mathbf{s}$. Let $\{\mathbf{a}, b\}$ be a $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$ sample. The distinguisher D then picks $\bar{s}_1 \in \mathbb{F}_p$ as its guess for the first value of $\mathbf{s}$, and computes

$$\{\mathbf{a}' = (a_2, a_3, \ldots, a_k), b' = b - a_1 \bar{s}_1\}.$$

If the guess $\bar{s}_1$ is correct, then $\{\mathbf{a}', b'\}$ is precisely a sample from $\Lambda_{k-1}(\mathbf{s})$. By definition, D can successfully distinguish such samples after Q queries with probability at least ϵ. On the other hand, if the guess is incorrect, b' will be independent of $\mathbf{a}'$ (since a_1 is chosen uniformly at random). In such a case, D enjoys no advantage. Repeating the same procedure in the order of $1/\epsilon$ times, D eventually recovers $\mathbf{s}$.

One can see that trying to recover the secret $\mathbf{s}$ after some n queries amounts to finding a noisy codeword $\mathbf{b}$ in the code $\mathcal{C}$ generated by $\mathbf{a}_i, i = 1, \ldots n$, where the noise consists of elements in $\mathbb{E}$. Equivalently, it suffices to find $\mathbf{e} \in \mathbb{E}^n$ such that $\mathbf{e}\mathbf{H}^\intercal = \mathbf{b}\mathbf{H}^\intercal = \mathbf{s}$. In conclusion, the hardness of RSDP implies the hardness of ORSDP_k, which again implies the "pseudo-randomness" of $\Lambda_k(\mathbf{s})$. In the next section, we show that it also extends to the case where the secret is not drawn uniformly at random.

When the number of Oracle calls is unlimited and the field size in the problem is fixed, then the RSDP problem from Definition 14 can be solved in polynomial time. This follows from algebraic attacks in the style of Aurora-Ge [AG11] and

can be launched whenever there are many values for the noise with probability 0. To this end, we introduce a slightly modified version of the RSDP-Oracle, called the δ-RSDP-Oracle.

Definition 16 (δ-RSDP-Oracle). *Fix $0 < \delta < 1$. The δ-RSDP-Oracle, denoted by $\mathcal{O}_{\mathbf{s}}^{\delta-\mathsf{RSDP}}$, responds as an RSDP-Oracle with probability $1-\delta$ and with a random pair of samples $\{\mathbf{a} \in \mathbb{F}_p^k, b \in \mathbb{F}_p\}$ with probability δ.*

To sum up, we consider two presumably difficult problems, the first one is the problem of distinguishing the RSDP-Oracle from random assuming a limited fixed number Q of queries, which corresponds to the RSDP problem. The second one is the same for the δ-RSDP-Oracle without limit on queries. In the sequel, we use the RSDP-Oracle, but the modifications needed to fit the use of δ-RSDP-Oracle are rather straightforward.

RSDP with a non-uniformly random secret

We briefly also discuss the hardness of RSDP when the secret $\mathbf{s}$ is not chosen uniformly at random. In particular, the entries of the secret $\mathbf{s}$ itself can be drawn from the same distribution as the error, i.e., randomly from $\mathbb{E}$. The reason for wanting this specific choice is that it lowers the secret key size and can give rise to very efficient implementations. For instance, with a proper choice of p and $\mathbb{E}$, e.g., $p = 127$ and $\mathbb{E} = \{2^i, i = 0, \ldots, 6\}$, multiplication of $\alpha \in \mathbb{F}_p$ with elements in $\mathbb{E}$ amounts to performing cyclic shifting on (the binary representation) of α, which can be implemented very efficiently on hardware as well as in software. This is crucial to our goal of constructing a lightweight cryptosystem. In our proposed constructions, we will however use $p = 127$ and $\mathbb{E} = \{(-2)^i, i = 0, \ldots, 13\}$, which share the same nice implementation properties.

However, one needs to be careful whether such a particular form of the secret key compromises the security of RSDP. Our case resembles the situation in the very well-studied learning problem *Learning with Errors* (LWE) [Reg09]. In their seminal work, Applebaum et al. [App+09a] shed light on a crucial observation: for an arbitrary noise distribution, the Learning with Error problem, where the secret's values are sampled from the same distribution as the noise, is **as hard as** the case where the secret is taken uniformly at random. Let us denote ORSDP_k^* by the problem of distinguishing the samples of a $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$ where values of $\mathbf{s}$ are drawn according to the noise distribution. By applying the same standard reduction as in [App+09a], we directly achieve that ORSDP_k^* is as hard as ORSDP_k

Lemma 8. *Assume there exists an algorithm D that can distinguish samples from $\mathcal{O}_{\mathbf{s}'}^{\mathsf{RSDP}}$ and uniform distribution U, where $\mathbf{s}'$ are drawn from the noise distribution. Then, one can leverage D to distinguish (with non-negligible probability) samples from $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$ and U, where $\mathbf{s}$ is chosen uniformly at random.*

Proof. (Sketch) Suppose we have access to an oracle $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$ where the secret is chosen uniformly at random. Our objective is to 'manipulate' samples from the

given oracle to obtain new ones as if they are given from another oracle, but the secret has the same distribution as the noise. First, let $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$ give out $\{\mathbf{a}_i^*, b_i^* = \langle \mathbf{a}_i^*, \mathbf{s}\rangle + e_i^*\}, i = 1, \ldots, k$ such that the matrix $\mathbf{A} = \{\mathbf{a}_i^*\}_{i=1}^k$ is invertible (with certain probability), and let $\mathbf{b}^* = \{b_i^*\}$. Then, ORSDP_k continues to sample fresh $\{\mathbf{a}_i, b_i\}$, and D computes

$$\{\mathbf{a}_i' = -\mathbf{A}^{-1}\mathbf{a}_i, b_i' = b + \langle \mathbf{a}_i', \mathbf{b}^*\rangle\}.$$

Then, one can check

$$\begin{aligned} b_i' &= \langle \mathbf{a}_i, \mathbf{s}\rangle + e_i + \langle -\mathbf{A}^{-1}\mathbf{a}_i, \mathbf{A}\mathbf{s} + \mathbf{e}_i^*\rangle \\ &= \langle \mathbf{a}_i', \mathbf{e}^*\rangle + e_i. \end{aligned}$$

In other words, $\{\mathbf{a}_i', b_i'\}$ are $\mathcal{O}_{\mathbf{e}^*}^{\mathsf{RSDP}}$ samples, where $\mathbf{e}^*$ is now the secret taken from the noise distribution. Indeed, with the view so far, since $\mathbf{a}_i$ is random and $\mathbf{A}$ is singular, $\mathbf{a}_i'$ is also random. By definition, D can distinguish these samples and, by extension, we can use the output from D to distinguish samples from $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$ and uniform distribution U. □

2.4 RSDP-solving Algorithms

With the introduction of RSDP, adaptations of SDP-solving algorithms have been proposed [Bal+20a; Bal+24c; Bit+23; BM24] in the new setting. In particular, we briefly investigate the Stern/Dumer variant, the BJMM variant, and the algebraic approach, which we will use to derive security parameters for our construction.

Stern/Dumer algorithm

The following is a straightforward adaptation of the Stern/Dumer algorithm [Ste88], originally proposed for the classical SDP. We recall the SDP instance given by $(\mathbf{H}, \mathbf{s}, \mathbb{E})$, where $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$ and $\mathbb{E}$ is a multiplicative subgroup of order z in $\mathbb{F}_p$. Our task is to find $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$, where $\mathbf{e} \in \mathbb{E}^n$. To ease the notation, we omit the transposition notation from now on. First, one applies to the parity-check matrix a partial Gaussian Elimination $\mathbf{H} \leftarrow \mathbf{U}\mathbf{H}\mathbf{P}$, with an invertible matrix $\mathbf{U}$ and some permutation $\mathbf{P}$, resulting in

$$(\mathbf{e}_1, \mathbf{e}_2)\begin{pmatrix} \mathbf{I}_{n-k-\ell} & \mathbf{H}_1 \\ \mathbf{0} & \mathbf{H}_2 \end{pmatrix} = (\mathbf{s}_1, \mathbf{s}_2),$$

where $\mathbf{e}\mathbf{P}^{-1} = (\mathbf{e}_1, \mathbf{e}_2) \in \mathbb{F}_p^{k+\ell} \times \mathbb{F}_p^{k+\ell}$ (with an additional later-to-be-optimized parameter ℓ). The equation gives rise to the identities

$$\mathbf{e}_1 + \mathbf{e}_2\mathbf{H}_1 = \mathbf{s}_1 \tag{1}$$

$$\mathbf{e}_2\mathbf{H}_2 = \mathbf{s}_2 \tag{2}$$

The next step of the Stern/Dumer approach is to enumerate $\mathbf{e}_2 \in \mathbb{E}^{k+\ell}$ to have candidates for (2). Equation (1) is then used to check if the candidates are the solution(s), that is, $\mathbf{s}_1 - \mathbf{e}\mathbf{H}_1$ is also a $\mathbb{E}^{k+\ell}$ vector. The enumeration step can be done via a Meet-in-the-Middle approach. In particular, we further parse $\mathbf{e}_2 = (\mathbf{x}_1, \mathbf{x}_2)$ where $\mathbf{x}_i \in \mathbb{E}^{(k+\ell)/2}$ and $\mathbf{H}_2 = \begin{pmatrix} \mathbf{H}_{21} \\ \mathbf{H}_{22} \end{pmatrix}$. We then construct two lists

$$\mathcal{L}_1 = \{(\mathbf{x}_1, \mathbf{x}_1\mathbf{H}_{21}), \mathbf{x}_1 \in \mathbb{E}^{(k+\ell)/2}\},$$

$$\mathcal{L}_2 = \{(\mathbf{x}_2, \mathbf{s}_2 - \mathbf{x}_2\mathbf{H}_{22}), \mathbf{x}_2 \in \mathbb{E}^{(k+\ell)/2}\},$$

and find collisions between them. For simplicity, we can assume $|\mathcal{L}_1| = |\mathcal{L}_2| = z^{(k+\ell)/2}$, and on average, there are $|\mathcal{L}_1|^2 \cdot p^{-\ell}$ collisions between them. The complexity of this step consists of building the lists and checking for solutions. In particular,

$$C_{\text{Stern/Dumer}} = \frac{C_1 + C_2 + C_{\text{coll}}}{(\text{number.of.solutions})} \times (\text{memory.access.cost}), \quad (3)$$

where

$$C_1 = C_2 = |\mathcal{L}_1| \cdot \left(\frac{k+\ell}{2} \cdot \log(z) + \ell \cdot \log(p) \right),$$

$$C_{\text{coll}} = |\mathcal{L}_1|^2 \cdot p^{-\ell} \cdot (k+\ell) \cdot \log(p).$$

The number of solutions is given by

$$1 + |\{\mathbf{e} \in \mathbb{E}^n : \mathbf{e}\mathbf{H} = \mathbf{s}\}| = 1 + z^n p^{k-n}.$$

BJMM algorithm

Similar to the BJMM algorithm [Bec+12b] in classical SDP, we can also adapt the *representation technique* in the enumeration step. This has been investigated in [Bit+23] and [Bal+24c]. Interestingly, the performance of such adaptations depends on the additive structure of the restricted set $\mathbb{E}$. We will briefly explain the idea and skip the details of the approach. For a more thorough understanding, we refer the readers to [Bal+24c].

Simplistically speaking, the representation technique aims to construct $\mathbf{e}_2 = \mathbf{e}_1^{(1)} + \mathbf{e}_2^{(1)}$ that solves equation (2). Since $\mathbb{E}$ is only a multiplicative subgroup, it is usually not closed under addition. Therefore, we have to sample $\mathbf{e}_i^{(1)}$ from some larger search domain defined as $\mathbb{E} \cup \mathbb{D} \cup \{0\}$ for some carefully chosen $\mathbb{D}$. The cost of this approach depends on the so-called "linearity" of $\mathbb{E}$ and $\mathbb{D}$. In particular, for

a random element $a \in \mathbb{E}$, we define the two quantities

$$\ell_{\mathbb{E}}(a) = |[b \in \mathbb{E} : \exists c \in \mathbb{E}, b + c = a]|,$$

$$\ell_{\mathbb{D}}(a) = |[b \in \mathbb{E} : \exists c \in \mathbb{D}, b + c = a]|.$$

These values will determine the number of representations r for an element during the enumerating process; hence, it implies how much we can save (by only enumerating a $1/r$-fraction). For the choice of p and $\mathbb{E}$ in CROSS, the above quantities are independent of a. Similar to the SDP case, one can repeat this step for $\mathbf{e}^{(1)} = \mathbf{e}_1^{(2)} + \mathbf{e}_2^{(2)}$ and so on, increasing the depth of the tree. However, for a fixed code rate of k/n and in the full-weight RSDP setting (as in CROSS parameters [Bal+24c]), this approach does not seem to yield improvements over the Stern/Dumer algorithm.

A clever alternative was proposed by the same authors: one can solve for a *shifted* instance of RSDP as $(\mathbf{e} + \mathbf{x})\mathbf{H} = \mathbf{s} + \mathbf{s}_{\mathbf{x}}$, where $\mathbf{s}_{\mathbf{x}}$ is the syndrome of a well-chosen $\mathbf{x}$ (e.g., $\mathbf{x} \in \mathbb{E}^n$). Such a transformation introduces 0 to the shifted error, from which BJMM might benefit. In particular, it was found that for the CROSS parameter, shifted-BJMM yields smaller spaces *shifted* $\mathbb{E}_x$ and $\mathbb{D}_x$ (while having the same linearity), which slightly outperforms Stern/Dumer.

However, as we will see in our setting and parameters selection, applying shifted-BJMM is very tricky. For example, if $\mathbb{E} := \{\pm 2^i\}$, then the linearity of shifted $\mathbb{E}$ and $\mathbb{D}$ are not constant. Nevertheless, we can conservatively lower bound the cost of shifted-BJMM by testing with different possible $\ell_{\mathbb{E}}$ and $\ell_{\mathbb{D}}$.

Both Stern/Dumer and BJMM enumerating approach requires very high memory usage (e.g., 2^{141} and 2^{116}, respectively, for NIST Category 1 parameters [Bal+24c]). Therefore, it is reasonable to take into account certain memory cost models. Similar to the CROSS authors, we use log model to estimate the cost of memory access. For example, in Equation (3), the complexity is multiplied with $\log(\max(|\mathcal{L}_1|, |\mathcal{L}_2|))$.

Algebraic approach

The basic Aurora-Ge approach [AG11] involves forming nonlinear equations for each sample of the form

$$\prod_{e \in \mathbb{E}} (\langle \mathbf{a}, \mathbf{s} \rangle + e - b) = 0,$$

which will be of degree z. Then, the system of equations is transformed into a linear one with around $\sum_{i=1}^{z} \binom{k}{i}$ unknowns (each is a monomial up to degree z of variables s_i in $\mathbf{s}$). Assuming in the order of $\binom{k}{z}$ oracle calls and $z < k/2$, one can then solve for the unknowns by Gaussian elimination in time around $\binom{k}{z}^3$. Better algorithms can lower the exponent.

More advanced algebraic attacks have been recently studied as an alternative to the traditional ISD algorithms, particularly for specific variants of SDP such as the Regular SDP [BØ23]. In [BM24], the authors investigated the prospect of applying algebraic attacks to various other variants, one of which is RSDP. Within this framework, SDP is modeled as solving a polynomial system. A general approach is to compute the Gröbner basis for the system, whose complexity depends on estimating the *degree of regularity* of said system. Algebraic approaches based on Gröbner basis techniques improve performance, but the improvement is often difficult to estimate.

2.5 The HB-family of authentication protocols

The HB-family authentication protocols were pioneered by Hopper and Blum [HB01a] to achieve a simple yet secure protocol based on the hardness of the learning problem. Since our design takes inspiration from HB-family protocols, especially the HB and the HB+ protocols, we briefly revisit the LPN problem, which gives rise to the aforementioned constructions.

Problem 7 (Learning Parity with Noise (LPN) Problem)**.** *Let* Ber_τ *be the Bernoulli distribution over* $\mathbb{Z}_2$ *with bias* $\tau \in \left(0, \frac{1}{2}\right)$*. That is, a random variable* $x \leftarrow \mathsf{Ber}_\tau$ *if* $\Pr[x = 1] = \tau$*. Given a secret* $\mathbf{s} \in \mathbb{Z}_2^k$*, the LPN Oracle* $\mathcal{O}_{\mathbf{s},\tau}^{LPN}$ *returns pairs in* $\mathbb{Z}_2^k \times \mathbb{Z}_2$ *of the form*

$$\{\mathbf{a} \xleftarrow{\$} \mathbb{Z}_2^k, z = \langle \mathbf{a}, \mathbf{s} \rangle \oplus e\},$$

where $e \leftarrow \mathsf{Ber}_\tau$*. The (decisional)* $\mathsf{LPN}_{\tau,k}$ *problem is defined as distinguishing the samples obtained from the above* $\mathcal{O}_{\mathbf{s},\tau}^{LPN}$ *from the uniform distribution.*

The HB protocol

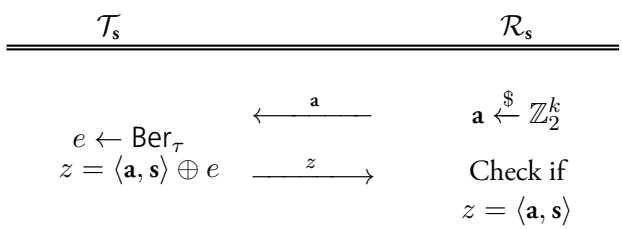


Figure 4: One round of the HB authentication protocol.

We recall the HB protocol proposed by Hopper and Blum [HB01a]. The Tag and the Reader share a binary length-k secret $\mathbf{s}$. One round of the protocol is illustrated in Figure 4. It is an simple 2-round interaction between $\mathcal{T}$ and $\mathcal{R}$. First, a "challenge" $\mathbf{a}$ is send by $\mathcal{R}$ and $\mathcal{T}$ responds with $z = \langle \mathbf{a}, \mathbf{s} \rangle \oplus e$ where $e \leftarrow \mathsf{Ber}_\tau$. Finally, $\mathcal{R}$ verifies where $z = \langle \mathbf{a}, \mathbf{s} \rangle$. On average, the check passes with

probability $(1-\tau)$, and a random response from an illegitimate tag is accepted with probability $\frac{1}{2}$. Therefore, to raise the confidence that $\mathcal{T}$ has the secret key, one repeats this protocol for n times. Therefore, $\mathcal{R}$ outputs accept if there are at most $\tau \cdot n$ failed checks[2].

The HB+ protocol

It was later shown by Juels and Weis [JW05a] that HB is only secure against a passive attacker. However, against a (more realistic) active adversary, the protocol is easily compromised. In particular, such an attacker $\mathcal{A}$ in Phase 1 can repeatedly challenge $\mathcal{T}_{\mathbf{s}}$ with the same $\mathbf{a}$. Since e is sampled according to Ber_τ, $\mathcal{A}$ can perform majority voting (after enough queries) to reveal the noise-free value $\langle \mathbf{a}, \mathbf{s} \rangle$. Repeating this process with linearly independent challenges $\mathbf{a}_i$, $\mathcal{A}$ can recover $\mathbf{s}$ by Gaussian Elimination. Hence, HB+ was proposed by Juels and Weis as an augmented version of HB. They turned HB into a 3-round interaction by requiring $\mathcal{T}$ to sends a "blinding" factor $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_2^k$ before $\mathcal{R}$'s challenge. Moreover, there are now two secret keys $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_2^k$. One round of the HB+ protocol is shown in Figure 5. Similarly, $\mathcal{R}$ outputs accept after n rounds if the number of failed checks is, at most, up to a certain threshold.

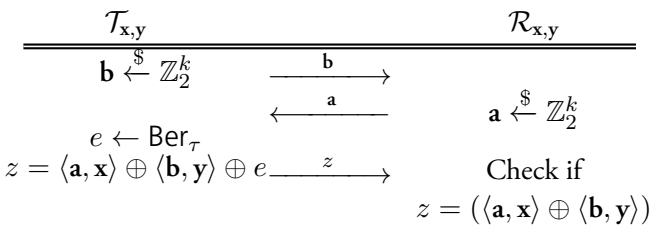


Figure 5: The HB+ authentication protocol.

However, against a simple MitM attack called GRS [GRS05], HB+ fell short. Roughly speaking, the active adversary $\mathcal{A}$ can add some perturbation $\mathbf{a}'$ to a challenge and send $\mathbf{a} + \mathbf{a}'$ to $\mathcal{T}$. If the authentication is successful, it follows that $\mathbf{x}\mathbf{a}' = 1$ with overwhelming probability. After retrieving $\mathbf{x}$, $\mathcal{A}$ can impersonate $\mathcal{T}$ in Phase 2 by setting $\mathbf{b} = 0$.

The HB# protocol

Gilbert et al. [GRS08b] propose the HB# protocol, employing two matrices as secret keys $\mathbf{X}, \mathbf{Y} \in \mathbb{Z}_2^{k \times n}$, effectively condensing all n rounds of HB+ into one. That is, the response is $\mathbf{z} = \mathbf{a}\mathbf{X} + \mathbf{b}\mathbf{Y} + \mathbf{e}$ that is secured to the GRS attack. However, it is vulnerable to a more general MitM adversary who can modify the

[2] In [JW05a], a higher threshold τ' can be chosen to minimize the number of rounds needed to obtain $P_c \leq 2^{-40}$ and $P_s \leq 2^{-80}$.

entire transcript. In very broad strokes, given an authorized transcript $(\bar{\mathbf{a}}, \bar{\mathbf{b}}, \bar{\mathbf{z}})$ (hidden noise $\bar{\mathbf{e}}$), $\mathcal{A}$ can invoke the interaction by replacing $(\mathbf{a}, \mathbf{b}, \mathbf{z}) \rightarrow (\bar{\mathbf{a}} + \mathbf{a}, \bar{\mathbf{b}} + \mathbf{b}, \bar{\mathbf{z}} + \mathbf{z})$. The crucial observation is that $\mathcal{R}$ will output accept if and only if $\omega_H(\bar{\mathbf{e}} + \mathbf{e}) \leq n\tau$, where $\mathbf{e}$ is the (hidden) noise for $(\mathbf{a}, \mathbf{b}, \mathbf{z})$. Repeating many enough times, $\mathcal{A}$ can confidently estimate $\omega_H(\bar{\mathbf{e}})$. By changing $\mathbf{z}$ one bit to $\bar{\mathbf{z}}'$, repeat the interception above with $(\bar{\mathbf{a}}, \bar{\mathbf{b}}, \bar{\mathbf{z}})$ and estimate the new noise $\bar{\mathbf{e}}'$, $\mathcal{A}$ can recover one bit of $\bar{\mathbf{e}}$. Eventually, the non erroneous value $\bar{\mathbf{a}}\mathbf{X} + \bar{\mathbf{b}}\mathbf{Y}$ is recovered. The mathematics is rather convoluted; hence, we refer the readers to [OOV08].

3 An authentication protocol based on RSDP

In this section, we propose a simple novel authentication protocol that is secure in the active model. In essence, it is the HB+ protocol, but the LPN problem has been replaced by the RSDP problem. In particular, we deploy the full-weight version of RSDP that has also been used recently in CROSS [Bal+24c].

Public parameters

The following are public parameters where n, k, p and z depends on the security parameter λ.

- $\mathbb{F}_p$: integers modulo prime p.
- $\mathbb{E} = \{g^i, i = 0, \ldots, z-1\}$: the multiplicative subgroup of order z in $\mathbb{Z}_p$.
- $k \in \mathbb{N}$: length of the secret keys.
- $n \in \mathbb{N}$: number of rounds in the authentication protocol.

Secret keys

The Tag and the Reader share two secret keys $\mathbf{x}, \mathbf{y} \in \mathbb{E}^k$. Figure 6 describes one round in the authentication protocol with $\mathcal{T}_{\mathbf{x},\mathbf{y}}$ and $\mathcal{R}_{\mathbf{x},\mathbf{y}}$. Similar to the HB+ protocol, it consists of three interactions between the Tag and the Reader. First, $\mathcal{T}_{\mathbf{x},\mathbf{y}}$ sends $\mathbf{b}$ and $\mathcal{R}_{\mathbf{x},\mathbf{y}}$ responds with a challenge $\mathbf{a}$. Then, $\mathcal{T}_{\mathbf{x},\mathbf{y}}$ samples an element $e \in \mathbb{E}$ at uniform random and compute $u = \langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle + e \bmod p$, which is then relayed to $\mathcal{R}_{\mathbf{x},\mathbf{y}}$. The Reader performs a check if $u - (\langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle) \in \mathbb{E}$. The round is then repeated n times to raise the confidence in the Tag. The Reader accepts if and only if all n checks were fulfilled.

Remark 1. *An active Adversary can repeatedly query with a challenge* $\mathbf{a}$*, e.g.,* $\mathbf{a} = \mathbf{0}$*, trying to solve recover* $\mathbf{y}$ *as the other key* $\mathbf{x}$ *no longer contributes to the response* z*. Therefore, the protocol security relies on the length of* $\mathbf{y}$*, and we only need to set the length of* $\mathbf{x}$ *so that guessing is infeasible. For example, 80-bit security requires the length of* $\mathbf{x}$ *to be at least* 22 *as* $z^{22} \approx 2^{83}$*. The protocol can be instantiated with*

$\mathcal{T}_{\mathbf{x},\mathbf{y}}$		$\mathcal{R}_{\mathbf{x},\mathbf{y}}$
$\mathbf{b} \xleftarrow{\$} \mathbb{F}_p^k$	$\xrightarrow{\mathbf{b}}$	
	$\xleftarrow{\mathbf{a}}$	$\mathbf{a} \xleftarrow{\$} \mathbb{F}_p^k$
$e \xleftarrow{\$} \mathbb{E}$		
$u = \langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle + e \bmod p$	$\xrightarrow{u}$	Check if
		$u - (\langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle) \bmod p \in \mathbb{E}$

Figure 6: One round of the RSDP-based HB+ authentication protocol. For simplicity, we let $k_{\mathbf{x}} = k_{\mathbf{y}} = k$.

different key lengths, denoted by $k_{\mathbf{x}}, k_{\mathbf{y}}$. For simplicity, in Figure 6 and the proof, we use the same key length.

Completeness of the authentication protocol

Contrary to HB-family authentication protocol, a legitimate tag in our proposal will produce $\mathbf{u}$ that passes the check of $\mathcal{R}_{\mathbf{x},\mathbf{y}}$ with probability 1.

Soundness of the authentication protocol

Let u_i be a random response in round i. To be authenticated, all $u_i, i = 1, \ldots, n$ need to pass the Reader's checks. Without knowing the secret, such a single event happens with a probability

$$P_s = (\Pr[u_i - (\langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle) \bmod p \in \mathbb{E}])^n = \left(\frac{z}{p}\right)^n.$$

3.1 Security against an active adversary

In this section, we show a reduction of the RSDP problem to the authentication protocol in the active model. To formalize an active attacker, we introduce the following notation. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be the active adversary in Phase 1 (querying the tag) and Phase 2 (authenticate to the reader), respectively. In phase 1, $\mathcal{A}$ has access to $\mathcal{T}$ in at most Q authenticating executions, and the adversary's actions are characterized by the program $\mathcal{A}_1$. During each execution, it receives $\mathbf{b}_i, i = 1, \ldots, n$ from $\mathcal{T}$, and sends back $\mathbf{a}_i$, for $i = 1, \ldots, n$ respectively, as challenges. Note that $\mathcal{A}_1$ can choose $\mathbf{a}_i$ as an active attacker. Then, $\mathcal{T}$ computes some z_i for $i = 1, \ldots, n$ as responses. In the end, $\mathcal{A}_1$ outputs some state σ that contains all information used for the next phase.

In the second phase, the attacking adversary impersonates the tag $\mathcal{T}$. Its complete action is described by the program $\mathcal{A}_2$, which takes the state σ as input. $\mathcal{A}_2$

sends some $\widehat{\mathbf{b}}_i$ (which is derived from σ), $\mathcal{R}$ then challenges with $\widehat{\mathbf{a}}_i$ and $\mathcal{A}_2$ provides the final $\widehat{z}_i$, attempting to pass the authentication protocol. After n rounds, $i = 1, \ldots, n$, the reader decides to accept or reject. The running time of $\mathcal{A}$, denoted by t, is determined by the maximum run time between $\mathcal{A}_1$ and $\mathcal{A}_2$.

We need to introduce further notation related to the error set $\mathbb{E}$. Let $\mathbb{D}$ denote the set

$$\mathbb{D} := \{e - e' | e, e' \in \mathbb{E}\}.$$

Furthermore, let $\alpha_{\mathbb{E}} := \frac{|\mathbb{D}|}{p}$.

Theorem 4. *Assume that the RSDP-based HB+ protocol in Figure 6 (with parameters p, k, n, and $\mathbb{E} = \{g^i, i = 0, \ldots, z-1\}$) is not (t, Q, ϵ)-secure, that is, there exists an active adversary $\mathcal{A}$, running in time t, interacting with the tag in at most Q executions, and achieving success probability at least ϵ in Phase 2. Then there exists a distinguisher D running in time $\mathcal{O}(t/\epsilon)$, making $Q \cdot n$ queries to an RSDP oracle $\mathcal{O}_{\mathbf{s}}^{\mathsf{RSDP}}$, for which*

$$\left| \Pr\left[\mathbf{s} \leftarrow \mathbb{E}^k : D^{\Lambda_k(\mathbf{s})} = 1\right] - \Pr\left[D^{U_{k+1}} = 1\right] \right| \geq 1 - \epsilon',$$

with

$$\epsilon' \approx \alpha_{\mathbb{E}}^n \cdot c^2/\epsilon^2 + (c+1) \cdot \exp(-c),$$

for some small constant c.

Proof. Let $(\mathbf{b}_i, u_i') \in \mathbb{Z}_p^k \times \mathbb{Z}_p$ be ordered pairs that are from an unknown distribution, which is either $\Lambda_k(\mathbf{s})$ or U_{k+1}. Assume that D has access to the program description $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ of an adversary $\mathcal{A}$ that can successfully attack the protocol described in Figure 6, in time t and after Q queries, with advantage ϵ. We now show that D can use $\mathcal{A}$ to distinguish whether its samples are from an RSDP oracle or drawn uniformly at random.

The following steps take inspiration from the reduction for the HB+ protocol in [KS06b]. The algorithm D first picks a random $\mathbf{y} \xleftarrow{\$} \mathbb{E}^k$. The idea is that D simulates a transcript as if, in the query phase, $\mathcal{A}$ is interacting with an honest tag $\mathcal{T}_{\mathbf{y},\mathbf{s}}$.

Phase 1

D runs $\mathcal{A}_1$. Recall that, in this phase $\mathcal{A}_1$ challenges the Tag with $\mathbf{a}_i$ after receiving some $\mathbf{b}_i$. First, D relays $\mathbf{b}_i$ to $\mathcal{A}_1$. For every challenge $\mathbf{a}_i$ made by $\mathcal{A}_1$, D computes and responds with

$$u_i = u_i' + \langle \mathbf{a}_i, \mathbf{y} \rangle \bmod p,$$

for $i = 1, \ldots, n$. After Q such executions, $\mathcal{A}_1$ outputs the state σ.

Phase 2

Next, D will use the $\mathcal{A}_2$ program. $\mathcal{A}_2(\sigma)$ starts by sending $\widehat{\mathbf{b}}_i$ to D, who then responds with $\widehat{\mathbf{a}}_i^1, i = 1, \ldots, n$. [3] Then, $\mathcal{A}_2(\sigma)$ computes some responses u_i^1 for $i = 1, \ldots, n$. The distinguisher D rewinds $\mathcal{A}_2(\sigma)$ but this time inputs a different $\widehat{\mathbf{a}}_i^2$ and observes the responses u_i^2 for $i = 1, \ldots, n$.

The distinguisher D proceeds to rewind and run $\mathcal{A}_2$ for M times and inputs different random $\widehat{\mathbf{a}}_i^j, i = 1, \ldots, n$, and $j = 1, \ldots, M$ and obtains u_i^j for $i = 1, \ldots, n$ and $j = 1, \ldots, M$.

Finally, D runs through all $1 \leq j_1 < j_2 \leq M$ and computes $\widehat{u}_i = u_i^{j_1} - u_i^{j_2} \bmod p$ for $i = 1, \ldots, n$ and $\widehat{u}_i' = \langle(\widehat{\mathbf{a}}_i^{j_1} - \widehat{\mathbf{a}}_i^{j_2}), \mathbf{y}\rangle$ for $i = 1, \ldots, n$. D checks if there exist two different such j values, j_1, j_2 for which we have that for all $i = 1, \ldots n$, $\exists e_i, e_i' \in \mathbb{E}$ such that $e_i - e_i' = \widehat{u}_i - \widehat{u}_i'$. If this is the case, D outputs 1 ($\Lambda_k(\mathbf{s})$), otherwise 0 (U_{k+1}).

- Case 1: Assume D gets samples from U_{k+1} in Phase 1. Then, the values $\widehat{u}_i - \widehat{u}_i'$ are uniformly and independently distributed. That is, D will make an error and output 1 iff $\widehat{U}_i = \widehat{u}_i - \widehat{u}_i'$ can be written as the subtraction of two elements in $\mathbb{E}$. This happens iff $\widehat{U}_i$ is in the set $\mathbb{D}$. A pair of j_1, j_2 then gives all n responses from a legit tag with probability $\alpha_{\mathbb{E}}^n := \left(\frac{|\mathbb{D}|}{p}\right)^n$. In other words, running through all pairs gives an error probability of D as most as $\alpha_{\mathbb{E}} \cdot \binom{M}{2}$. In other word,

$$\Pr\left[D^{U_{k+1}} = 1\right] = \alpha_{\mathbb{E}}^n \cdot \binom{M}{2}.$$

- Case 2: Assume D gets samples from $\Lambda_k(\mathbf{s})$. Then, in Phase 1, we simulated the transcript as if $\mathcal{A}$ interacted with $\mathcal{T}_{\mathbf{y},\mathbf{s}}$. Indeed,

$$u_i = u_i' + \langle \mathbf{a}_i, \mathbf{y}\rangle \bmod p = \langle \mathbf{b}_i, \mathbf{s}\rangle + \langle \mathbf{a}_i, \mathbf{y}\rangle + e_i.$$

  By definition, $\mathcal{A}_2$ produces a correct response with probability at least ϵ. We now show that for a well-chosen M, at least a pair of correct responses exists with high probability. $\mathcal{A}$ fails to produce any correct pair of responses with probability

$$M \cdot \epsilon \cdot (1-\epsilon)^{M-1} + (1-\epsilon)^M \approx$$
$$M \cdot \epsilon \cdot \exp(-\epsilon \cdot (M-1)) + \exp(-\epsilon \cdot M).$$

[3] The superscript $1, 2, \ldots$ in $\widehat{\mathbf{a}}^1, \widehat{\mathbf{a}}^2$ are not exponentiation but corresponds to the whole authentication attempt where $\mathcal{A}_2$ sends challenges.

That is,

$$\Pr\left[\mathbf{s} \leftarrow \mathbb{E}^k : D^{\Lambda_k(\mathbf{s})} = 1\right] = \\ 1 - M \cdot \epsilon \cdot \exp(-\epsilon \cdot (M-1)) + \exp(-\epsilon \cdot M).$$

We want the probability of D being successful to be close to 1; hence, one chooses $M = c \cdot \frac{1}{\epsilon}$, for some constant $c > 1$. This proves the theorem.

Hence, if c is big enough, D can tell if the samples are from $\Lambda_k(\mathbf{s})$ with probability very close to 1.

□

Example 4. *Let $\epsilon = 2^{-20}$, we only need $M = \frac{5}{\epsilon}$ to have probability 0.96 to have at least one correct pair.*

3.2 A MitM Attack Strategy

Here, we point out that the proposed protocol from Figure 6 is not secure against a MitM attacker. The attack depends on the additive structure of the restricted set $\mathbb{E}$. Consider an adversary that observes one sample in the first round of the protocol as before:

$$u = \langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle + e \bmod p$$

and tries to guess $e \in \mathbb{E}$. Recall that in the MitM model, $\mathcal{A}$ are allowed to interfere and alter $(\mathbf{b}, \mathbf{a}, u)$ before forwarding to either $\mathcal{T}$ or $\mathcal{R}$. The adversary $\mathcal{A}$ first picks an element $e' \in \mathbb{E}$ and assumes that $e = e'$. Then pick another element $e'' \neq e'$, such that $e'' \in \mathbb{E}$. Then substitute u with u', which is calculated as

$$u' = \langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle + e - e' + e'' \bmod p$$

in this first round. In all other rounds $i = 2, \ldots, n$, the adversary forwards the responses u without modifications.

Assume u' passes the verification check, which can only occur when $e - e' + e'' \in \mathbb{E}$, and depending on the choice of $\mathbb{E}$, $\mathcal{A}$ can obtain some information regarding e or even correctly guess e.

Example 5. *Let $p = 127$ and $\mathbb{E} = \{1, 2, 4, 8, 16, 32, 64\}$. It is then easily checked that if $e' \neq e'' \in \mathbb{E}$ and $e - e' + e'' \in \mathbb{E}$ then $e = e'$. Equivalently, when u' passes the check, $\mathcal{A}$ correctly guess $e = e'$. Knowing e, the adversary can compute $u - e$ and conclude that*

$$u - e = \langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle \bmod p.$$

This gives a linear equation in the unknown key variables $\mathbf{x}, \mathbf{y}$. The Reader accepts the corrupted response with probability $1/|\mathbb{E}|$ (for each authentication). Thus, it takes the attacker on average $|\mathbb{E}|$ attempts to reveal a noise-free value $\langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle$, or at

most $(n+k)|\mathbb{E}|$ *attempts to reveal the entire secret key by solving a system of linear equations through Gaussian elimination.*

Example 6. *For our later choice of* $\mathbb{E}$, $p = 127$ *and* $\mathbb{E} = \{(-2)^i, i = 0, \ldots, 13\}$. *Guessing* e *is more challenging as there are many* e' *and* e'' *that yield a seemingly valid* u'. *However, by definition, the design is not MitM-secure as it means that* $\mathcal{A}$ *can forge a response* $\mathbf{u}' \neq \mathbf{u}$ *(with non-negligible probability) that will be authenticated by* $\mathcal{R}$.

Going further, $\mathcal{A}$ *can also perform a key recovery attack. In contrast to Example 5, for fixed* e' *and* e'', *there are more than one possible values for* e *(that makes* u' *pass). As an example,* $(\mathrm{e}', \mathrm{e}'') = (1, 8)$ *yields* $\mathrm{e} = 1$ *or* $\mathrm{e} = 19$, *which allows* $\mathcal{A}$ *to form a quadratic equation as* $(\mathrm{e}-1)(\mathrm{e}-8) = 0 \bmod p$. *By changing different* $(\mathrm{e}', \mathrm{e}'')$ *and going through* n *responses coming from the Tag,* $\mathcal{A}$ *obtains more information (and equations) about each error position, which could be used to speed up combinatoric solvers (or algebraic).*

4 Parameters for the proposed protocol

We propose parameters for 3 security levels, namely $80, 112$, and 128 bits. The parameters are selected based on the performance of the various RSDP-solving algorithms.

As in Remark 1, an active attacker using RSDP solvers to recover the secret key(s) amounts to solving an RSDP instance with parameter n and secret length $k_{\mathbf{y}}$. When $n < k_{\mathbf{y}}$, there are exponentially many solutions. However, one has to find the exact $e_1, \ldots, e_n$ produced by $\mathcal{T}$ to recover $\mathbf{x}$. The attacker can indeed observe many authentications to get more than n samples to guarantee that the solution from the solver is indeed correct. However, once the average number of solutions is 1, Stern/Dumer and BJMM do not seem to take advantage of the extra samples.

We stress that the performance of Stern/Dumer or BJMM takes into account the cost of accessing huge memory under the log model. Indeed, such algorithms require such high memory usage that can not be disregarded. For example, Shifted-BJMM needs 2^{70} bits of memory for 80-bit security parameters.

All proposed instances use $p = 127$ and $\mathbb{E} = \{(-2)^i, i = 0, \ldots 13\}$ as the multiplicative subgroup in $\mathbb{F}_p$, so $z = 14$.

Table 2: Parameters recommendation for the RSDP HB+ protocol with n as the number of authentication steps, $k_{\mathbf{x}}$, $k_{\mathbf{y}}$ as the length of two secret keys, and p as the field size. The restricted set is set to be $\mathbb{E} := \{(-2)^i, i = 0, \ldots z-1\}$.

Security Level	80	112	128
(p, z)	(127,14)	(127,14)	(127,14)
$(k_{\mathbf{x}}, k_{\mathbf{y}})$	(22,34)	(30,54)	(34,70)
n	26	36	41

We present the cryptanalysis on the 80-bit security parameters with $\mathbb{E} = \{\pm 1, \pm 2, \ldots, \pm 64\}$ Interestingly, the biggest threat to this parameter set is the algebraic attack. Since the algebraic approach cost can be lower-bounded by roughly $\binom{k}{z}^3$, it prevents $k_{\mathbf{y}}$ from being too small. Since there are better methods compared to Gaussian elimination that can lower the exponent, we conservatively choose $k_{\mathbf{y}} = 34$ as $\binom{34}{14}^3 \approx 2^{91}$. For a d-bit security level, we require the soundness error $\left(\frac{14}{127}\right)^n \leq 2^{-d}$. For example, $n = 26$ for 80-bit security level.

Now we consider the combinatorial approach, such as Stern and BJMM, assuming an adversary using such solvers is allowed multiple interactions (to guarantee the correctness of the found solution). In that case, it does not seem to yield improvements as the size of $\mathbb{E}$ is quite big, thus significantly increasing enumerating efforts. For example, the Stern(BJMM) in Section 2.4 requires 2^{105} (2^{110}, resp.) operations and 2^{94} (2^{98}, resp.) bits in memory. In contrast to the CROSS parameter, shifted-BJMM does not improve over Stern for our choice of $\mathbb{E}$. In addition, for higher security-level parameters, the gap between algebraic and combinatorial solvers gets significantly wider.

Other settings

The most well-studied setting of RSDP, as can be seen in CROSS, is $p = 127, z = 7$ ($\mathbb{E} = \{2^i, i = 0, \ldots, 6\}$). In particular, it is employed in all targeted NIST security levels (up to 256-bit). However, this choice of z makes our protocol vulnerable to the algebraic attack by [AG11]. Table 3 shows the change of $k_{\mathbf{y}}$ the keys with $z = 7$.[4] Such a drastic change can make our protocol unreasonably expensive regarding storage in an environment-constrained device. In this case, the adversary needs to observe 2^{30} (80-bit) to 2^{48} (128-bit) authentications to success. In summary, this setting is not secure unless there is a reasonable countermeasure to thwart excessive queries. Therefore, our choice of parameters in Table 2 can

Table 3: The restricted set is set to be $\mathbb{E} := \{2^i, i = 0, \ldots z - 1\}$.

Security Level	80	112	128
(p, z)	(127,7)	(127,7)	(127,7)
$(k_{\mathbf{x}}, k_{\mathbf{y}})$	(22,71)	(30,210)	(34,400)

be seen as a simple and effective way to achieve better security without adding substantial overhead (only additional $\log 2$ bits for each key symbol stored).

4.1 Performance

In this section, we discuss the proposed protocol with different benchmarks such as computation, communication, and key size(s).

[4] Again, combinatoric approaches are not competitive to algebraic attacks.

Key size

The protocol employs two keys of lengths $k_{\mathbf{x}}, k_{\mathbf{y}}$ each, of which entries are drawn from the restricted set $\mathbb{E}$; therefore, we need roughly $(k_{\mathbf{x}} + k_{\mathbf{y}}) \cdot \log(z)$ bits to store the keys.

Communication

One authentication consists of n 3-round authenticating step. In each step, the Tag and the Reader exchange $(\mathbf{a}, \mathbf{b}, u) \in \mathbb{F}_p^{k_{\mathbf{x}}} \times \mathbb{F}_p^{k_{\mathbf{y}}} \times \mathbb{F}_p$, which incurs $n \times (k_{\mathbf{x}} + k_{\mathbf{y}} + 1) \times \log(p)$ bits in communication.

Example 7. *80-bit security parameters of LPN-based HB+ are roughly* $(512, 1/8)$ *where* 512 *is the length of the key and* $1/8$ *is the noise rate. To achieve good completeness and soundness probability (*2^{-40} *and* 2^{-80}*, respectively), one needs to repeat an authentication step by roughly* $n = 441$ *times.*

Table 4: Key size (in bits) of different protocols.

Security Level	80	112	128
RSA	1024	2048	3072
DSA	1024	2048	3072
LPN-based	592	880	896
RSDP HB+	217	320	396

Table 4 shows the key sizes of our protocol in comparison with some traditional cryptographic primitives and LPN-based protocols. Note that there are several options for LPN-based protocols to achieve one security level. For instance, one can select $(k_{\mathbf{x}} = 112, k_{\mathbf{y}} = 512, \tau = 0.49)$ for 112-bit security. However, such a high noise level will be detrimental to both the completeness and soundness of the scheme, thus implying a very high number of authentication steps, i.e., high communication cost. Therefore, for LPN-based protocols, we pick values of k that correspond to "reasonable" noise rates (typically ≤ 0.3).[5]

4.2 Hardware Implementation

The most critical task that affects performance in Figure 6 is computing

$$u = \langle \mathbf{a}, \mathbf{x} \rangle + \langle \mathbf{b}, \mathbf{y} \rangle + e \bmod p$$

This operation is implemented using hardware in the tag and needs to be as lightweight as possible. Our choice of the error set $\mathbb{E}$, and the choice of p ensures that the operation remains cheap. We implemented two strategies for both protocols based

[5] Table in Section 5.2 in [LF06b] can serve as a rough guideline for LPN parameters.

on RSDP and LPN, with parameters targeting 80-bit security. The implementations focus on the inner product operation, which is the most costly operation implemented on the tag hardware. In both strategies, the cost is dominated by the circuits needed to store the key and the challenge. In the case of LPN, both variables consist of 512 bits, while in the case of RSDP, the key consists of 136 bits, while the challenge consists of 238 bits. In the case of LPN, one vector coordinate consists of 1 bit, while in the case of RSDP, one challenge vector coordinate consists of 7 bits, and 1 key vector coordinate consists of 4 bits in a signed integer format. Each multiplication, in the case of RSDP, consists of two steps:

1. Multiplication of the challenge coordinate by the unsigned value of the key coordinate $\mod 127$, which boils down to a rotation based on the key values.

2. Multiplication by either 1 or $-1 \mod 127$, which boils down to a conditional bitwise negation.

In strategy A, we implement the operation serially, as depicted in Figure 7. Every clock cycle, we perform one multiplication operation followed by an accumulation. The inner product operation takes several clock cycles. In strategy B, we implement the inner product operation in one clock cycle, and add all the products using an addition tree. Strategy A targets the minimum possible area (storage with a minimal combinational circuit), while strategy B targets the highest possible speed (1 clock cycle per inner product). We synthesized the circuits for both the Artix 7 FPGA using Xilinx Vivado and FDSOI 28nm using Synopsys Design Compiler, and the results are provided in Tables 5 and 6, respectively. We observe that on FPGA, strategy B works better for LPN than strategy A, but the RSDP-based protocol is smaller and faster for both strategies. Similar trends emerge in ASIC implementations, the RSDP circuit being 60% and 30% smaller for strategies A and B, respectively. We note that LPN favors strategy B over strategy A due to its very lightweight combinational logic compared to the storage cost, while RDSP offers a somewhat linear trade-off due to its more involved combinational/arithmetic logic. However, it still experiences a significant gain in all cases.

Table 9 shows the comparison in performance between RSDP-based and LPN-based authentication protocols. Note that the number of rounds for LPN-based protocol is chosen so that they have acceptable correctness and soundness probabilities ($\leq 2^{-40}$, and $\leq 2^{-80}$) (Figure 2, [LF06b]). We then extrapolate the overall performance into Table 1.

As we can see from Table 9, the RSDP-based approach provides remarkable benchmarks compared to its LPN-based counterpart. The RSDP-based protocol requires very modest keys compared to LPN-based counterparts. The reason is that RSDP offers a much higher security level for the same key length, and we only need to use $\log z$ bits for each key symbol. In addition, we also see reduced communication cost (column 2) per round for $\mathcal{T}$ and $\mathcal{R}$. Most importantly, the

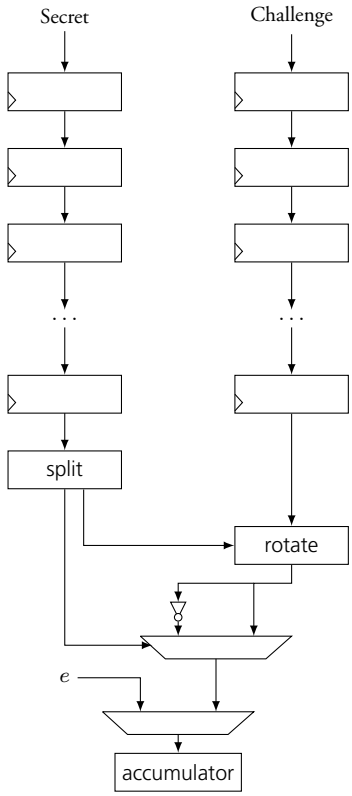


Figure 7: Serialized implementation of the inner product operation (strategy A).

Table 5: FPGA Resource Utilization for 80-bit security using Xilinx Artix 7 and Xilinx Vivado. LUTs: Look-Up Tables, FFs: Flip-Flops.

Design	LUTs	FFs	Cycles
LPN-based	1028	1025	512
	1032	1025	64
	780	1025	1
RSDP-based	399	381	34
	421	381	18
	678	266	1

RSDP-based protocol needs very few authentication rounds: 17 times and 28 times less for 80-bit and 128-bit parameters, which yields very competitive total bandwidth. Although round parallelization is an option, it burdens power and hardware utilization on $\mathcal{T}$, which can be unreasonable for devices with low power or constrained environment, such as an RFID tag. As hinted in the introduction, due to elementary arithmetic (XOR), the GEs in LPN do not change much if

Table 6: ASIC area for 80-bit security using FDSOI 28nm. GEs: Gate Equivalents.

Design	GEs	Cycles
LPN-based	12243.75	512
	12262.5	64
	12985	1
RSDP-based	4678	34
	4797.18	18
	8614.69	1

Table 7: FPGA Resource Utilization for 128-bit security using Xilinx Artix 7 and Xilinx Vivado. LUTs: Look-Up Tables, FFs: Flip-Flops.

Design	LUTs	FFs	Cycles
LPN-based	1540	1537	768
	1544	1537	96
	1169	1537	1
RSDP-based	619	601	70
	641	601	35
	999	394	1

Table 8: ASIC area for 128-bit security using FDSOI 28nm. GEs: Gate Equivalents.

Design	GEs	Cycles
LPN-based	18349.13	768
	18389.25	96
	19428.97	1
RSDP-based	7304.22	70
	7421.47	35
	13249.47	1

Table 9: Performance comparison in 1 round of RSDP-based and LPN-based authentication protocol for 128-bit level security

Design	Key size	Communication	Rounds
RSDP-based	396	735	41
LPN-based	896	897	1164

we follow Strategy A or B. In contrast, RSDP allows optimizations/trade-offs depending on the hardware at hand. Therefore, we argue that RSDP is a much more friendly and flexible candidate for a lightweight authentication protocol.

5 A MitM-secured proposal based on RSDP

In this section, we present an extended protocol that provides security also in the MitM model. We adopt the approach of Lyubashevsky et al. [LM13a] to achieve this, with the addition of a few additional assumptions.

Definition 17. *A family function $\mathcal{F} : \mathbb{D} \to \mathbb{F}$ is said to be a* weak pseudorandom functions *(wPRFs) if for $f \xleftarrow{\$} \mathcal{F}$, it is computationally infeasible to distinguish input-output pairs $(x_i, f(x_i))$, where x_i are chosen uniformly at random from $\mathbb{D}$, from random pairs $(x_i, y_i) \in (\mathbb{D}, \mathbb{F})$.*

The condition of wPRFs can be further relaxed by a family of functions $\mathcal{F}_\chi$, provided the output of $f \in \mathcal{F}$ is indistinguishable after perturbation of *noise* characterized by a distribution χ. Such a family is called *randomized wPRFs*.

Definition 18. *$\mathcal{H} : \mathbb{D} \to \mathbb{F}$ is a pairwise independent function family if*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x_1) = y_1 \wedge h(x_2) = y_2] = 1/|\mathbb{F}|^2,$$

for all $x_1 \neq x_2$, $y_1 \neq y_2$.

In brevity, the design in [LM13a] relies on a (randomized) wPRFs $\mathcal{F}_\chi : \mathbb{D} \to \mathbb{F}$, a pairwise independent function family $\mathcal{H} : \mathbb{D} \to \mathbb{F}$, where $\mathbb{F}$ is a finite field. Importantly, a weight function $|\cdot|$ (defined on $\mathbb{F}$), the field $\mathbb{F}$, and χ have to satisfy certain (reasonable) properties. In essence, together, they have to provide good completeness (close to 1) and soundness probability. A natural instantiation for LPN is $\mathbb{F} = \mathbb{Z}_2[x]/\langle g(x) \rangle$, for some irreducible polynomial $g(x)$, and χ as the Bernoulli distribution Ber_η^n, and the Hamming weight function. Despite such requirements not applying to our situation, we have seen in Section 3 that RSDP naturally yields good completeness and soundness probability. Hence, we propose a protocol that achieves MitM security in Figure 8.

The following instantiation of the protocol in Figure 8 can be seen as an RSDP version of the protocol in [LM13a]. Note that the protocol now runs only a single round as the final response $\mathbf{u}$ from the Tag is a vector (or, equivalently, an element in a finite field).

Public parameters

The following are public parameters where k, n, p and z depends on the security parameter λ.

- $\mathbb{Z}_p$: integers modulo p.
- $\mathbb{E}$: a multiplicative subgroup of order z in $\mathbb{Z}_p$.

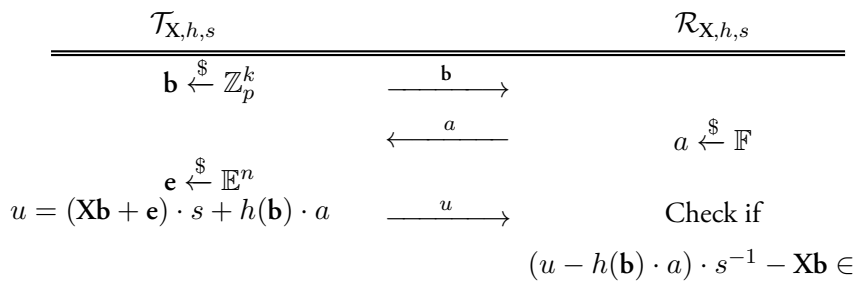


Figure 8: An RSDP adaptation to Lyubashevski et al. for MitM security. Here, we use the vector form of $(u - h(\mathbf{b}) \cdot a) \cdot s^{-1}$ in the check.

- $\mathbb{F} = \mathbb{Z}_p[x]/\langle g(x)\rangle$ for some irreducible polynomial $g(x)$ over $\mathbb{Z}_p$ of degree n. Multiplications, denoted by $\cdot$ in $\mathbb{F}$, are seen as polynomial multiplications. Operations between a vector and an element in $\mathbb{F}$ are assumed to be done after converting the vector to a corresponding element. For instance, a vector can be seen as coefficients of a polynomial in $\mathbb{F}$, and vice versa.

Secret keys

The Tag and the Reader both share secret keys $\mathbf{X} \in \mathbb{E}^{n\times k}$, $h \in \mathcal{H}$, and $s \in \mathbb{F}$. The function h is defined as $h(\mathbf{x}) = h_1 \cdot x + h_2$, for $h_i \in \mathbb{F}$ and x is the corresponding polynomial to vector $\mathbf{x}$ in $\mathbb{F}$.

Hardness assumptions

Having $\mathbf{X}$ as a secret matrix is equivalent to having many RSDP instances with different secrets $\mathbf{x}_i$ (columns of $\mathbf{X}$). However, as we have discussed, RSDP is not hard when an adversary $\mathcal{A}$ has access to an unlimited number of queries. Therefore, for the reduction proof of this design, we have to assume that $\mathcal{A}$ is allowed up to Q interactions with the Tag, where Q is a fixed value. Storing a matrix $\mathbf{X} \in \mathbb{E}^{k\times n}$ could pose a practical challenge for a low-cost RFID Tag. Therefore, similar to previous work, one can consider a version where $\mathbf{X}$ is a Toeplitz matrix.

5.1 Security reduction of the MitM proposal

We now want to prove security for the protocol in Figure 8. We are inspired by the methodology from [LM13a]. Their work considers a slightly different MitM adversary where it does not operate in two phases. Instead, a MitM attacker interacts with $\mathcal{T}$ and $\mathcal{R}$ for Q times and modifies the communications $(\mathbf{b}, a, u) \rightarrow (\mathbf{b} + \mathbf{b}', a + a', u + u')$. The attacker wins the game if one out of Q interaction, a non-trivial change in the communication, i.e., $(\mathbf{b}', a', u')$ are not simultaneously 0, yields an accept from $\mathcal{R}$. However, if the attacker wins in this scenario, it also wins in the 2-stage model.

$(\mathbf{b}_i, \mathbf{y}_i) \rightarrow$ $\mathcal{T}$ (sim.)	$\mathcal{A}$	$\mathcal{R}$ (sim.)
choose some random s, h	$\xrightarrow{\mathbf{b}_i + \mathbf{b}'_i}$	
	$\xleftarrow{a_i + a'_i}$	$a_i \xleftarrow{\$} \mathbb{F}$
$u_i = \mathbf{y}_i \cdot s + h(\mathbf{b}_i) \cdot (a_i + a'_i)$	$\xrightarrow{u_i + u'_i}$	Check if $(\mathbf{b}'_i, a'_i, u'_i) \neq (\mathbf{0}, 0, 0)$, reject else, accept

Figure 9: Simulating $\mathcal{T}$ and $\mathcal{R}$ before the winning query.

In this reduction, we assume the existence of an adversary $\mathcal{A}$ that breaks the design after Q authentication queries with probability ϵ. It means that we assume that $\mathcal{A}$ makes $Q-1$ unsuccessful attempts followed by a Q-th query, where the Reader output accept for $(\mathbf{b}'_i, a'_i, u'_i) \neq (\mathbf{0}, 0, 0)$ with probability ϵ. Obviously, this implies that an adversary $\mathcal{A}$ that is allowed to win in any query has success probability at most ϵQ.

Theorem 5. *Assume that the protocol in Figure 8 is not (t, Q, ϵ)-secured, that is, there exists an active adversary $\mathcal{A}$, running in time t, interacting with the Tag and Reader in at most Q executions, and can produce $(\mathbf{b}', a', u') \neq (\mathbf{0}, 0, 0)$ that is authenticated with probability at least ϵ. Then there exists an algorithm D running in time $\mathcal{O}(t)$, making $Q \cdot n$ queries to an RSDP oracle (with secret as a matrix), and*

$$\left|\Pr\left[\mathbf{X} \leftarrow \mathbb{E}^{k \times n} : D^{\Lambda_k(\mathbf{X})} = 1\right] - \Pr\left[D^{U_{k+1}} = 1\right]\right| \geq \left(\epsilon - 1/p^k\right)^2 - \alpha_{\mathbb{E}}^n.$$

Proof. Let $\mathcal{A}$ be a MitM attacker as described above. In each authentication query, $\mathcal{A}$ changes $\mathbf{b} \rightarrow \mathbf{b} + \mathbf{b}'$, $a \rightarrow a + a'$, $u \rightarrow u + u'$.

We now describe a distinguisher for the (vectorized) RSDP oracle using $\mathcal{A}$ as a part. The Challenger sends to the distinguisher pairs $(\mathbf{b}_i, \mathbf{y}_i)$ where $\mathbf{b}_i$ are uniformly random and $\mathbf{y}_i$ are either uniformly random or $\mathbf{y}_i = \mathbf{X}\mathbf{b}_i + \mathbf{e}_i$, for some random matrix $\mathbf{X}$. Now, the distinguisher simulates the Tag and the Reader for all queries. First, random values of the secrets h, s are chosen. For execution i, D receives $(\mathbf{b}_i, \mathbf{y}_i)$ from the challenger and chooses a random a_i. Then $(\mathbf{b}_i, a_i)$ is input to $\mathcal{A}$ who responds with $(\mathbf{b}'_i, a'_i)$. D computes u_i from its known values as

$$u_i = \mathbf{y}_i \cdot s + h(\mathbf{b}_i) \cdot (a_i + a'_i),$$

and gives to $\mathcal{A}$, who responds with u'_i. Finally, we simulate the response from the reader by simply rejecting if $(\mathbf{b}'_i, a'_i, u'_i) \neq (\mathbf{0}, 0, 0)$. This is the simulation before the winning query, and it is depicted in Figure 9.

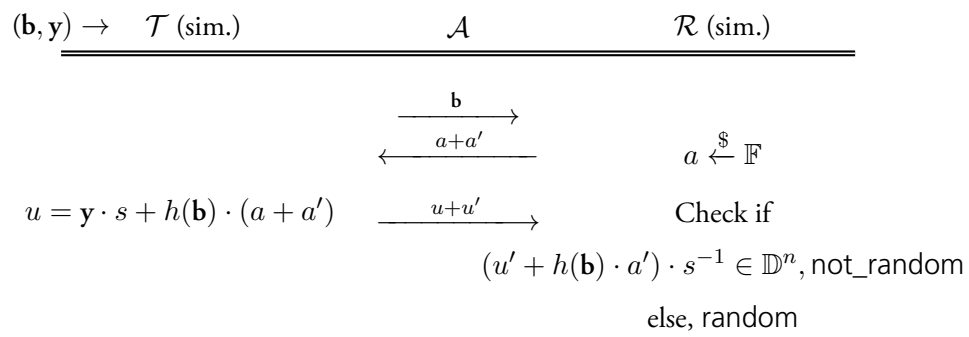


Figure 10: Response with the winning query when $\mathbf{b}' = \mathbf{0}$.

We argue that this is a correct simulation of the protocol up to $Q-1$ queries. If $\mathbf{y}_i = \mathbf{X}\mathbf{b}_i + \mathbf{e}_i$, then $\mathcal{T}$ and $\mathcal{R}$ behaves as if they have secrets $\mathbf{X}$, s and h. Then, for $(\mathbf{b}'_i, a'_i, u'_i) = (\mathbf{0}, 0, 0)$, $\mathcal{R}$ outputs accept correctly. If $(\mathbf{b}'_i, a'_i, u'_i) \neq (\mathbf{0}, 0, 0)$, $\mathcal{R}$ outputs reject correctly since the winning query has not been reached.

Next, we will show how to use the Q-th winning query to output the answer to the Challenger correctly. The distinguished acts differently in the two cases: $\mathbf{b}' = \mathbf{0}$ and $\mathbf{b}' \neq \mathbf{0}$. From now on, we denote the winning query by $(\mathbf{b}', a', u') \neq (\mathbf{0}, 0, 0)$.

The case $\mathbf{b}' = \mathbf{0}$

The response to the challenger is as given in Figure 10.

- Case 1: Assume that the challenger sends RSDP pairs. Then, since (a', u') is the winning response, the following must be accepted by the Reader.

  $$(u + u' - h(\mathbf{b}) \cdot a) \cdot s^{-1} - \mathbf{X}\mathbf{b} \in \mathbb{E}^n.$$

  On the other hand, since the pairs $(\mathbf{b}, \mathbf{y})$ from the Challenger is an "RSDP" pair, we also have

  $$(u - h(\mathbf{b}) \cdot (a + a')) \cdot s^{-1} - \mathbf{X}\mathbf{b} \in \mathbb{E}^n.$$

  Subtracting the two equations, one obtains

  $$(u' + h(\mathbf{b}) \cdot a') \cdot s^{-1} \in \mathbb{D}^n,$$

  i.e., in this case, the response is always not_random.

- Case 2: Assume that the challenger sends a random pair. Then, the simulator responds not_random if and only if

  $$(u' + h(\mathbf{b}) \cdot a') \cdot s^{-1} \in \mathbb{D}^n$$

with randomly chosen h and s. Moreover, previous unsuccessful queries do not leak information about these secrets, So $\mathcal{A}$ behaves as if u', a' are chosen before h and s. As $\mathbf{b}' = \mathbf{0}$, we have that u', a' cannot both be zero. This leads to

$$\Pr[(u' + h(\mathbf{b}) \cdot a') \cdot s^{-1} \in \mathbb{D}^n] = \alpha_{\mathbb{E}}^n,$$

for any choice of u', a'.

In summary, if $\mathbf{b}' = \mathbf{0}$, then one has

$$\left| \Pr\left[\mathbf{X} \leftarrow \mathbb{E}^{k \times n} : D^{\Lambda_k(\mathbf{X})} = 1\right] - \Pr\left[D^{U_{k+1}} = 1\right] \right| = \epsilon - \alpha_{\mathbb{E}}^n.$$

The case $\mathbf{b}' \neq \mathbf{0}$

The response to the challenger in this case is as given in Figure 11. Here we first run $\mathcal{A}$ in the winning query with random input $\mathbf{a_0}$, then we rewind $\mathcal{A}$ and run it again with another random input $\mathbf{a_1}$.

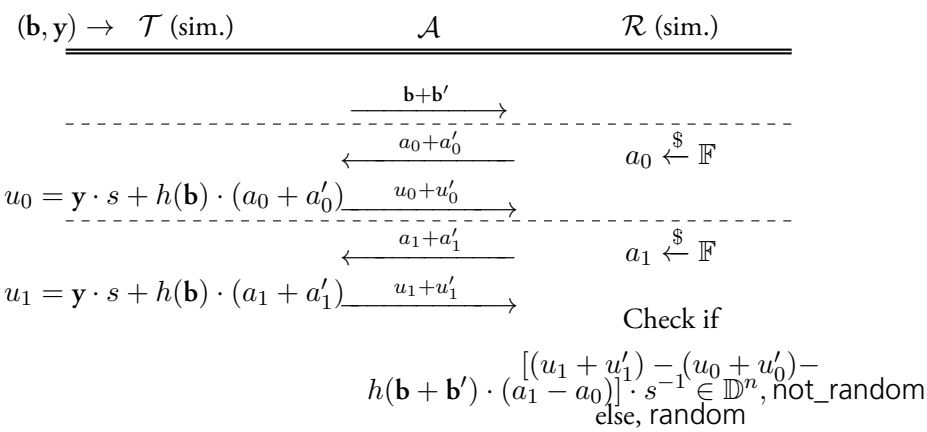


Figure 11: Response with the winning query when $\mathbf{b}' \neq \mathbf{0}$.

- Case 1: Assume that the challenger sends an RSDP pair. To detect if $\mathcal{A}$ responds correctly without knowing the secret $\mathbf{X}$, one needs to get rid of $\mathbf{X}(\mathbf{b}+\mathbf{b}')$. Therefore, we have to *rewind* $\mathcal{A}$ to produce $\mathbf{X}(\mathbf{b}+\mathbf{b}')$ twice with different challenges a_0 and a_1 (which happens with probability $1 - 1/p^k$). In particular, the simulated Reader sends a_0 and the Tag computes u_0 as in Figure 11. Then we rewind $\mathcal{A}$ back to the point it sends $\mathbf{b} + \mathbf{b}'$. This time, a different challenge a_1 is sent, and similarly, a new u_1 is computed.

The distinguisher now has two responses $u_0 + u'_0$ and $u_1 + u'_1$. Both of them are simultaneously correct with different challenges with probability

$$\left(\epsilon - 1/p^k\right)^2,$$

and it follows that

$$[(u_1 + u'_1) - (u_0 + u'_0) - h(\mathbf{b} + \mathbf{b}') \cdot (a_1 - a_0)] \cdot s^{-1} \in \mathbb{D}^n,$$

meaning that in this case the distinguisher always gives the correct response not_random.

- Case 2: Assume that the challenger sends random pairs. We now investigate

$$\Pr\left[[(u_1 + u'_1) - (u_0 + u'_0) - h(\mathbf{b} + \mathbf{b}') \cdot (a_1 - a_0)] \cdot s^{-1} \in \mathbb{D}^n\right]. \tag{4}$$

Let $\tilde{a} = a_1 - a_0$ (and correspondingly for $\tilde{a}', \tilde{u}'$), we can rewrite Equation 4 as

$$\Pr\left[[\tilde{u}' + h(\mathbf{b}) \cdot (\tilde{a} + \tilde{a}') - h(\mathbf{b} + \mathbf{b}') \cdot \tilde{a}] \cdot s^{-1} \in \mathbb{D}^n\right].$$

As previous unsuccessful queries do not leak information about h and s, $\mathcal{A}$ can be considered choosing $\tilde{u}', \tilde{a}'$ before h and s. For all $\mathbf{t} \in \mathbb{D}^n$,

$$\Pr[h(\mathbf{b} + \mathbf{b}') = h(\mathbf{b}) \cdot (\tilde{a} + \tilde{a}') + \tilde{u}' - \mathbf{t} \cdot s)] = \frac{1}{p^n},$$

by definition of pairwise independent function. Therefore,

$$\Pr\left[[\tilde{u}' + h(\mathbf{b}) \cdot (\tilde{a} + \tilde{a}') - h(\mathbf{b} + \mathbf{b}') \cdot \tilde{a}] \cdot s^{-1} \in \mathbb{D}^n\right] = \alpha_{\mathbb{E}}^n.$$

In summary, in this case, one has

$$\left|\Pr\left[\mathbf{X} \leftarrow \mathbb{E}^{k \times n} : D^{\Lambda_k(\mathbf{X})} = 1\right] - \Pr\left[D^{U_{k+1}} = 1\right]\right| = \left(\epsilon - 1/p^k\right)^2 - \alpha_{\mathbb{E}}^n.$$

One can extend the proof to use multiple rewinding as in Theorem 4 to achieve tighter reduction. □

6 Parameters for the MitM design

In contrast to the active-secured design, the RSDP sample is now masked with a secret polynomial s, which makes it challenging to apply any RSDP solvers to retrieve the secret $\mathbf{X}$ (even in the case that $\mathbf{X}$ is a Topeliz matrix). It is reasonable to assume that an Adversary will face a harder problem than just RSDP. Therefore,

one can use the same parameters proposed in Table 10 for this design if the cost factor is paramount.

Table 10: Agressive parameters for the MitM-secured RSDP authentication protocol with p as the field size, and the restricted set is set to be $\mathbb{E} := \{(-2)^i, i = 0, \dots z-1\}$.

Security Level	80	112	128
(p, z)	(127,14)	(127,14)	(127,14)
k	34	54	70
n	26	36	41

To be more conservative, one can select parameters based on the security reduction. Theorem 5 has tightness $\sqrt{\epsilon}$. In other words, the protocol is only half as secure as the RSDP problem and n has to be big enough so that $\alpha_{\mathbb{E}}^n$ can be deemed negligible. In particular, for d-bit security level, we ask for $\alpha_{\mathbb{E}}^n \leq 2^{-d}$.

Table 11: Conservative parameters recommendation for the MitM-secured authentication protocol with p as the field size, and the restricted set is set to be $\mathbb{E} := \{(-2)^i, i = 0, \dots z-1\}$.

Security Level	80	112	128
(p, z)	(127,14)	(127,14)	(127,14)
k	55	79	91
n	96	134	153

There are several ideas when it comes to reducing the cost even more. For instance, selecting the coefficient of s, or secret polynomial h_1, and h_2, to be also in the restricted set. However, more careful work has to be done to understand the security implications of such risky options.

Example 8. *Let us look at some efficiency comparisons between LPN and RSDP instantiation for* 80*-bit security. Since it is unclear how the steepness of reduction translates to security in practice, one can choose the parameters that yield* 80*-bit security for both the RSDP and LPN problem as in Section 4. In particular, (roughly)* $(n, k, \epsilon) = (441, 512, 1/8)$ *for LPN and* $(n, k, p, z) = (26, 34, 127, 14)$ *for RSDP.*

- *In terms of key storage, assuming we use Toepliz matrices for both cases, LPN uses* $k + 4n = 2276$ *bits, while RSDP uses* $(k + n) \cdot \log(z) + 3n \cdot \log(p) = 770$ *bits.*
- *LPN instantiation uses* $3k = 1323$ *and RSDP uses* $3k \cdot \log(p) = 545$ *bits in communication.*

Therefore, we can see that RSDP, in the MitM-secured construction, still offers significant advantages over traditional LPN-based protocols in terms of key storage and communication costs.

7 Conclusion

In this paper, we have presented novel authentication protocols based on the Restricted Syndrome Decoding Problem. We show a natural adaptation of the new problem in two constructing directions: HB-family and wPRFs-based authentication protocols. From a theoretical viewpoint, security reductions from the previous works translate to RSDP (with a few additional assumptions, in the case of wPRFs-based). For practical interests, we show that with a well-chosen restricted set $\mathbb{E}$, the proposed protocol yields impressive performance well suited to low-cost cryptographic primitives. More importantly, such a choice of $\mathbb{E}$ does not compromise the security regarding available RSDP solvers.

Acknowledgement

The work was supported by the Swedish Civil Contingencies Agency (grant number 2020-11632); the Swedish Foundation for Strategic Research (Grant No. RIT17-0005); and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Mustafa Khairallah is funded by the Wallenberg-NTU Presidential Postdoctoral Fellowship.

References

[AFS05] D. Augot, M. Finiasz, and N. Sendrier. "A Family of Fast Syndrome Based Cryptographic Hash Functions". In: *Progress in Cryptology - Mycrypt 2005, First International Conference on Cryptology in Malaysia, Kuala Lumpur, Malaysia, September 28-30, 2005, Proceedings*. Ed. by E. Dawson and S. Vaudenay. Vol. 3715. Lecture Notes in Computer Science. Springer, 2005, pp. 64–83.

[AG11] S. Arora and R. Ge. "New algorithms for learning in presence of errors". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2011, pp. 403–415.

[App+09a] B. Applebaum et al. "Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems". In: *Advances in Cryptology - CRYPTO 2009*. Ed. by S. Halevi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 595–618.

[Ara+21] N. Aragon et al. "BIKE". In: (2021). available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions`.

[Bal+20a] M. Baldi et al. "A New Path to Code-based Signatures via Identification Schemes with Restricted Errors". In: *CoRR* abs/2008.06403 (2020). arXiv: `2008.06403`.

[Bal+24a] M. Baldi et al. “CROSS”. In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2024).

[Bal+24c] M. Baldi et al. “Zero Knowledge Protocols and Signatures from the Restricted Syndrome Decoding Problem”. In: *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part II*. Ed. by Q. Tang and V. Teague. Vol. 14602. Lecture Notes in Computer Science. Springer, 2024, pp. 243–274.

[Ban+23] G. Banegas et al. “Wave”. In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2023).

[BC08] J. Bringer and H. Chabanne. “Trusted-HB: a low-cost version of HB+ secure against Man-in-The-Middle attacks”. In: *CoRR* abs/0802.0603 (2008). arXiv: 0802.0603.

[Bec+12b] A. Becker et al. “Decoding Random Binary Linear Codes in 2n/20: How 1 + 1 = 0 Improves Information Set Decoding”. In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by D. Pointcheval and T. Johansson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 520–536.

[Ber+11] D. J. Bernstein et al. “Really Fast Syndrome-Based Hashing”. In: *Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings*. Ed. by A. Nitaj and D. Pointcheval. Vol. 6737. Lecture Notes in Computer Science. Springer, 2011, pp. 134–152.

[Ber+20] D. J. Bernstein et al. “Classic McEliece: conservative code-based cryptography”. In: *NIST submissions* (2020).

[Bit+23] S. Bitzer et al. “Generic Decoding of Restricted Errors”. In: *IEEE International Symposium on Information Theory, ISIT 2023, Taipei, Taiwan, June 25-30, 2023*. IEEE, 2023, pp. 246–251.

[BM24] W. Beullens and P. B. andIn contrast Morten Øygarden. “A Security Analysis of Restricted Syndrome Decoding Problems”. In: *IACR Cryptol. ePrint Arch.* (2024), p. 611.

[BMT78b] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. “On the inherent intractability of certain coding problems (Corresp.)” In: *IEEE Trans. Inf. Theory* 24.3 (1978), pp. 384–386.

[BØ23] P. Briaud and M. Øygarden. "A New Algebraic Approach to the Regular Syndrome Decoding Problem and Implications for PCG Constructions". In: *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V*. Ed. by C. Hazay and M. Stam. Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 391–422.

[CFS01] N. T. Courtois, M. Finiasz, and N. Sendrier. "How to Achieve a McEliece-Based Digital Signature Scheme". In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. Ed. by C. Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 157–174.

[DK07] D. N. Duc and K. Kim. *Securing* HB+ *against GRS man-in-the-middle attack*. 2007.

[FS96] J. Fischer and J. Stern. "An Efficient Pseudo-Random Generator Provably as Secure as Syndrome Decoding". In: *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*. Ed. by U. M. Maurer. Vol. 1070. Lecture Notes in Computer Science. Springer, 1996, pp. 245–255.

[GRS05] H. Gilbert, M. Robshaw, and H. Sibert. *An Active Attack Against* HB+ *- A Provably Secure Lightweight Authentication Protocol*. `https://eprint.iacr.org/2005/237`. 2005.

[GRS08a] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "Good Variants of HB + Are Hard to Find". In: *Financial Cryptography and Data Security*. Ed. by G. Tsudik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 156–170.

[GRS08b] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "HB$^{\#}$: Increasing the Security and Efficiency of HB^{+}". In: *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*. Ed. by N. P. Smart. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[GRS08c] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. "HB#: Increasing the Security and Efficiency of HB+". In: *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[HB01a] N. J. Hopper and M. Blum. "Secure Human Identification Protocols". In: *Advances in Cryptology — ASIACRYPT 2001*. Ed. by C. Boyd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 52–66.

[Hey+12] S. Heyse et al. "Lapin: An Efficient Authentication Protocol Based on Ring-LPN". In: *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*. Ed. by A. Canteaut. Vol. 7549. Lecture Notes in Computer Science. Springer, 2012, pp. 346–365.

[JW05a] A. Juels and S. A. Weis. "Authenticating pervasive devices with human protocols". In: *Advances in Cryptology–CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005. Proceedings 25*. Springer. 2005, pp. 293–308.

[Kil+11] E. Kiltz et al. "Efficient Authentication from Hard Learning Problems". In: *Advances in Cryptology – EUROCRYPT 2011*. Ed. by K. G. Paterson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 7–26.

[KS06b] J. Katz and J. S. Shin. "Parallel and Concurrent Security of the HB and HB + Protocols". In: *Advances in Cryptology - EUROCRYPT 2006*. Ed. by S. Vaudenay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 73–87.

[LF06b] É. Levieil and P.-A. Fouque. "An improved LPN algorithm". In: *Security and Cryptography for Networks: 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006. Proceedings 5*. Springer. 2006, pp. 348–359.

[LGQ13a] Z. Li, G. Gong, and Z. Qin. "Secure and Efficient LCMQ Entity Authentication Protocol". In: *IEEE Transactions on Information Theory* 59.6 (2013), pp. 4042–4054.

[LGQ13b] Z. Li, G. Gong, and Z. Qin. "Secure and efficient LCMQ entity authentication protocol". In: *IEEE transactions on information theory* 59.6 (2013), pp. 4042–4054.

[LM13a] V. Lyubashevsky and D. Masny. “Man-in-the-Middle Secure Authentication Schemes from LPN and Weak PRFs”. In: *Advances in Cryptology – CRYPTO 2013*. Ed. by R. Canetti and J. A. Garay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 308–325.

[Mel+23a] C. A. Melchor et al. “Hamming quasi-cyclic (HQC)”. In: *NIST PQC* (2023).

[Mel+23b] G. A. Melchor et al. “SDitH”. In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2023).

[MP07] J. Munilla and A. Peinado. “HB-MP: A further step in the HB-family of lightweight authentication protocols”. In: *Computer Networks* 51.9 (2007). (1) Advances in Smart Cards and (2) Topics in Wireless Broadband Systems, pp. 2262–2267.

[NJG24] V. Nguyen, T. Johansson, and Q. Guo. “A Key-Recovery Attack on the LCMQ Authentication Protocol”. In: *2024 IEEE International Symposium on Information Theory (ISIT)*. 2024, pp. 1824–1829.

[OOV08] K. Ouafi, R. Overbeck, and S. Vaudenay. “On the Security of HB# against a Man-in-the-Middle Attack”. In: *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*. Vol. 5350. Lecture Notes in Computer Science. Springer, 2008, pp. 108–124.

[Reg09] O. Regev. “On lattices, learning with errors, random linear codes, and cryptography”. In: *J. ACM* 56.6 (Sept. 2009).

[Ste88] J. Stern. “A method for finding codewords of small weight”. In: *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*. Ed. by G. D. Cohen and J. Wolfmann. Vol. 388. Lecture Notes in Computer Science. Springer, 1988, pp. 106–113.

[Ste93] J. Stern. “A New Identification Scheme Based on Syndrome Decoding”. In: *Advances in Cryptology - CRYPTO ’93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*. Ed. by D. R. Stinson. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 13–21.

[Vér96] P. Véron. “Improved identification schemes based on error-correcting codes”. In: *Appl. Algebra Eng. Commun. Comput.* 8.1 (1996), pp. 57–69.

A BKW-Style Solver for the Restricted Syndrome Decoding Problem

Abstract

The Restricted Syndrome Decoding Problem (RSDP) is a variant of the well-known syndrome decoding problem. It has recently been turned into a post-quantum signature scheme named CROSS by Baldi et al.. It is a scheme highlighted for being computationally friendly and providing a compact signature and public key size. This paper investigates an Oracle-based definition of the RSDP that has already proved useful in constructing other cryptographic primitives. We propose a new solving algorithm for this novel and interesting problem. Our approach is to first introduce a new weight definition for vectors over $\mathbb{F}_p$ and then develop a solving algorithm similar to the BKW algorithm that includes finding many such low-weight vectors in a dual space. We make use of several advanced techniques, such as *Covering codes* and *Subspace Hypothesis Testing*. We show that when there are many samples, our algorithm can be more advantageous than prior work based on Information-set decoding adaptations for RSDP or algebraic approaches.

1 Introduction

Recently, as an attempt to diversify our cryptographic portfolio, the National Institute for Standard and Technology (NIST) announced the Post-Quantum Ad-

Thomas Johansson, Qian Guo, Vu Nguyen. "A BKW-Style Solver for the Restricted Syndrome Decoding Problem". In *International Workshop on Code-Based Cryptography 2025, Madrid, Spain*.

ditional Signature Schemes call, where many *code-based* proposals were featured, such as CROSS [Bal+24c], WAVE [Ban+23], LESS [Bal+24b], SDitH [Mel+23b] to name a few. Code-based cryptography has always been at the forefront of post-quantum cryptography research as extensive studies for more than half a century have accumulated a good understanding and high confidence in code-based hardness assumptions. At the heart of code-based cryptography lies the well-known *Syndrome Decoding Problem* (SDP), which has been proven useful in many cryptographic applications. Most notable are KEMs such as McEliece [Ber+20], BIKE [Ara+21], HQC [Mel+23a]), as well as other constructions, e.g., zero-knowledge proof systems [Ste93; Vér96], signatures [CFS01], hash functions [AFS05; Ber+11], stream ciphers [FS96], and so on.

Many code-based constructions rely on $\mathcal{NP}$-complete variants of the original SDP, especially when performance is a limiting factor. Numerous works have been conducted in this direction, such as employing "structured" noise: the *Regular Decoding Problem* [AFS05; CCJ23; Cui+24], *Permuted Kernel Problem* [Sha90; Beu+19], etc., or SDP in different metrics (rank metric, Lee metric) other than the Hamming weight. In particular, the second round of the NIST signature call has seen the advancement of LESS and CROSS, whose security revolves around the *Code Equivalence Problem*, and the new and novel *Restricted Syndrome Decoding Problem* (RSDP).

The RSDP was first introduced in [Bal+20a], where the author introduced an efficient zero-knowledge identification protocol. Since then, many cryptanalytic studies have been done to improve our understanding of this interesting problem. Solvers for RSDP fall into two categories: adaptations of the well-known SDP algorithm *Information-Set Decoding* (ISD), and algebraic solvers.

Recently, another class of SDP algorithms, called *Statistical Decoding* [Car+22], has been revisited, and it seems promising in certain code rate regimes. In particular, when the code rate can be made arbitrarily small, which effectively turns (binary) SDP into the *Learning Parity with Noise* (LPN) problem, there exists another algorithm called BKW [BKW03b] that can solve LPN in sub-exponential complexity. Therefore, the natural question arises of whether applying the BKW algorithm to solve RSDP is viable, especially when the code rate is also small.

Contributions

In this work, we reintroduce the RSDP problem in its "oracle" form, allowing an adversary to obtain an arbitrary number of RSDP samples. We propose a BKW-style approach to solve the RSDP problem that resembles the BKW algorithm for LPN and Statistical Decoding for the classical Syndrome Decoding Problem. Our approach is to first introduce a new weight definition for vectors over $\mathbb{F}_p$ and then develop a solving algorithm that includes the process of finding many low-weight vectors in a dual space. We incorporate several state-of-the-art techniques that improve the basic BKW algorithm, such as *covering codes* and *subspace hypothesis*

testing [GJL14; GJS15]. In contrast to ISD solvers, our algorithm can benefit from additional samples and bridge the gap between combinatorial and algebraic attacks.

On the one hand, we enrich the cryptanalysis for RSDP primitives, which is an interesting and novel topic in the post-quantum cryptography landscape. On the other hand, our take can serve as a vital tool for designing new RSDP-based cryptographic primitives. We believe that, as the concrete hardness of RSDP is better understood, it opens the doors for new applications, such as RSDP-based authentication similar to protocols based on the LPN problem [JKN25].

In addition, we extend our results to RSDP instances that have not been studied, e.g., when the restricted set is larger than that of CROSS. The dual-style solver can be adapted to such a scenario, yielding significant improvements.

Organization

In Section 2, we go through notions in coding theory, revisit the RSDP, and introduce a new Oracle-form definition of RSDP. Then, we summarize the state-of-the-art cryptanalysis of this new problem. Section 3 explains our BKW-style approach in clear, identifiable steps, and in Section 4, we analyze the computation cost of our algorithm. Section 5 presents several case studies regarding the algorithm's performance when applied to different RSDP settings. In particular, we study the CROSS RSDP setting and investigate the scenario with a larger restricted set. We then verify heuristic arguments made throughout the paper by simulations with small RSDP parameters in Section 6. Lastly, we conclude our paper in Section 7.

2 Preliminaries

In this section, we briefly go through the mathematical background of our work. Throughout the paper, we use the following notations:

- $a, \mathbf{a}, \mathbf{A}$ for single elements, vectors, and matrices, respectively.
- $\mathbf{I}_n$ the identity matrix of size $n \times n$.
- $\mathbb{F}_p$ the finite field of order p, where p is a prime and $\mathbb{F}_p^n$ is the corresponding vector space of dimension n.
- $X \xleftarrow{\$} \chi$ a random variable following the distribution χ.
- $\mathcal{U}(\mathcal{X})$ is the uniform distribution on the set $\mathcal{X}$.
- If $X \xleftarrow{\$} \chi$ and χ is not the uniform distribution, then X is said to be *biased*. The size of the bias, a measure of how much χ differs from the uniform distribution, can be measured in different ways and will be detailed later.

Let p be a prime number and $\mathbb{F}_p$ be a finite field. A $[n, k]$-linear code $\mathcal{C}$ over $\mathbb{F}_p$ ($k \leq n$) is a vector subspace of dimension k in $\mathbb{F}_p^n$. A full-rank matrix $\mathbf{G} \in \mathbb{F}_p^{k\times n}$ is said to be the generator of $\mathcal{C}$ if $\mathcal{C} = \{\mathbf{uG} : \mathbf{u} \in \mathbb{F}_p^k\}$, i.e., the rows of $\mathbf{G}$ form a basis of $\mathcal{C}$. A matrix $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$ is said to be a parity-check matrix for the code $\mathcal{C}$ if $\mathbf{GH}^\intercal = \mathbf{0}$. Let $\mathbf{y} \in \mathbb{F}_p^n$, then $\mathbf{s} = \mathbf{yH}^\intercal$ is called its *syndrome* (w.r.t $\mathbf{H}$). We often assume $\mathbf{G}$ and $\mathbf{H}$ to be in their *systematic form*, that is $\mathbf{G} = \begin{pmatrix}\mathbf{I}_k & \mathbf{A}\end{pmatrix}$ for some $\mathbf{A} \in \mathbb{F}_p^{k\times(n-k)}$, and $\mathbf{H} = \begin{pmatrix}-\mathbf{A}^\intercal & \mathbf{I}_{n-k}\end{pmatrix}$. The Hamming weight of a vector $\mathbf{x} \in \mathbb{F}_p^n$, denoted by $\omega_H(\mathbf{x})$, is defined as the number of non-zero coordinates in $\mathbf{x}$.

2.1 Restricted Syndrome Decoding Problem (RSDP)

Let $\mathbb{F}_p$ be a finite field. Consider $g \in \mathbb{F}_p^*$ of order z, and denote the subgroup generated by g as $\mathbb{E} = \{g^i, i \in \{1, \ldots, z\}\} \subset \mathbb{F}_p^*$. The RSDP, first introduced in [Bal+20a] in a slightly different form, is here defined as follows:

Problem 8 (Restricted Syndrome Decoding Problem). *Given* $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$ *and a syndrome* $\mathbf{s} \in \mathbb{F}_p^{n-k}$. *Find* $\mathbf{e}$ *(if any) where* $\mathbf{e} \in \mathbb{E}^n$, *and* $\mathbf{eH}^\intercal = \mathbf{s}$.

To ease the notation, we sometimes omit the transposition notation when the sizes of matrices and vectors are evident from the context.

Alternative definitions could include to allow entries in $\mathbf{e}$ to be also zero. In this latter case, a restriction of the weight of $\mathbf{e}$ can additionally be added. The RSDP is related to other well-known hard problems. For example, it has been shown in [Bal+20a] that RSDP is NP-complete for any choice of $\mathbb{E}$. In this section, we redefine the RSDP in a slightly different fashion that is similar to the *Learning with Errors* (LWE) problem [Reg09] and the LPN problem, as originally proposed in [JKN25].

Problem 9 (RSDP Oracle). *Let* k *and* p *be positive integers. An RSDP oracle* $\mathcal{O}_{\mathbf{u},\mathbb{E}}$ *for a secret vector* $\mathbf{u} \in \mathbb{F}_p^k$ *and a multiplicative subgroup* $\mathbb{E} \subset \mathbb{F}_p^*$ *is an oracle returning*

$$(\mathbf{h}, b = \langle \mathbf{h}, \mathbf{u}\rangle + e),$$

where $\mathbf{h} \xleftarrow{\$} \mathcal{U}(\mathbb{F}_p^k)$, $e \xleftarrow{\$} \mathcal{U}(\mathbb{E})$ *each time the oracle is called.*

Calling the RSDP Oracle n times gives the returned values

$$(\mathbf{h}_1, b_1 = \langle \mathbf{h}_1, \mathbf{u}\rangle + e_1), \ldots, (\mathbf{h}_i, b_i = \langle \mathbf{h}_i, \mathbf{u}\rangle + e_i), \ldots, (\mathbf{h}_n, b_n = \langle \mathbf{h}_n, \mathbf{u}\rangle + e_n).$$

In other words, we observe noisy codewords from a q-ary code, where the noise comes from the restricted set $\mathbb{E}$. With a fixed number of queries, recovering the secret key $\mathbf{u}$ amounts to solving the RSDP, where the parity-check matrix can be deduced from $\mathbf{h}_i$, $i = 1, \ldots, n$. The difference is that an adversary can choose

n and work with a code with an arbitrarily small code rate. In contrast to LWE using discrete Gaussian noise, as we will see later, RSDP in this form is not hard when we have an unlimited amount of queries as it can be vulnerable to algebraic attacks.

The conversion steps are as follows. Let $\mathbf{G} = [\mathbf{h}_1\mathbf{h}_2 \dots \mathbf{h}_n]$. Then the oracle responses give $\mathbf{uG} + \mathbf{e} = \mathbf{b}$, where $\mathbf{e} = (e_1, e_2, \dots, e_n)$ and $\mathbf{b} = (b_1, b_2, \dots, b_n)$. The parity check matrix for $\mathbf{H}$ is a matrix such that $\mathbf{GH}^\intercal = \mathbf{0}$. Multiplying with $\mathbf{H}^\intercal$ gives $\mathbf{uGH}^\intercal + \mathbf{eH}^\intercal = \mathbf{bH}^\intercal$ and we have $\mathbf{eH}^\intercal = \mathbf{s}$, where $\mathbf{s} = \mathbf{bH}^\intercal$ is the syndrome. This shows the equivalence between the two descriptions of RSDP, one as a syndrome decoding problem and the oracle version as a "learning problem". We summarize the main problem for our work.

Definition 19. *(Search-RSDP). The Search-RSD problem is the problem of recovering the hidden secret* $\mathbf{u}$ *given* n *queries with* $(\mathbf{h}_i, b_i) \in \mathbb{F}_p^k \times \mathbb{F}_p, i = 1, \dots, n$ *obtained from* $\mathcal{O}_{u,\mathbb{E}}$.

Secret-Noise transformation

In the definition of RSDP (original and Oracle form), the secret $\mathbf{u}$ is drawn from the uniform distribution. Similarly to the case of LWE, through a transformation as in [App+09a], we can transform the secret $\mathbf{u}$ into a "new" secret $\overline{\mathbf{u}}$, where the secret itself is drawn from $\mathbb{E}^k$. Therefore, from now on, we simply assume that $\mathbf{u} \in \mathbb{E}^k$ for the sake of simplicity.

For completeness, we describe this secret-noise transformation for the RSDP problem step by step. Suppose we have access to the RSDP oracle $\mathcal{O}_{u,\mathbb{E}}$ which, upon query, provides samples of the form $(\mathbf{h}_i, b_i) = (\mathbf{h}_i, \langle \mathbf{h}_i, \mathbf{u} \rangle + e_i)$, where $\mathbf{u}$ is the secret vector, $\mathbf{h}_i$ is chosen uniformly at random, and $e_i \xleftarrow{\$} \mathcal{U}(\mathbb{E})$. Our objective is to manipulate these samples to create new samples where the 'secret' component effectively behaves like noise drawn from $\mathbb{E}^k$.

The process begins by collecting k samples from the RSDP oracle. Let us arrange the query vectors $\mathbf{h}_i$ from these k samples as the rows of a $k \times k$ matrix $\mathbf{H}_0$, and the corresponding responses b_i as a vector $\mathbf{b}_0$. We can then represent these k samples in matrix form as $(\mathbf{H}_0, \mathbf{b}_0) = (\mathbf{H}_0, \mathbf{H}_0\mathbf{u} + \mathbf{e}_0)$, where $\mathbf{e}_0$ is a vector comprised of the k error terms e_i. With a certain probability, the matrix $\mathbf{H}_0$ will be invertible over the field $\mathbb{F}_p$. Assuming $\mathbf{H}_0$ is indeed invertible, we can precompute its inverse, $\mathbf{H}_0^{-1}$.

To generate a transformed sample, we make a new query to the RSDP oracle, obtaining a sample $(\mathbf{h}_1, b_1) = (\mathbf{h}_1, \langle \mathbf{h}_1, \mathbf{u} \rangle + e_1)$. Now, we compute a new value $b' = -\mathbf{h}_1\mathbf{H}_0^{-1}\mathbf{b}_0 + b_1$, where,

$$\begin{aligned} b' &= -\mathbf{h}_1\mathbf{H}_0^{-1}\mathbf{b}_0 + b_1 \\ &= -\mathbf{h}_1\mathbf{H}_0^{-1}(\mathbf{H}_0\mathbf{u} + \mathbf{e}_0) + (\langle \mathbf{h}_1, \mathbf{u} \rangle + e_1) \\ &= -\mathbf{h}_1\mathbf{H}_0^{-1}\mathbf{H}_0\mathbf{u} - \mathbf{h}_1\mathbf{H}_0^{-1}\mathbf{e}_0 + \langle \mathbf{h}_1, \mathbf{u} \rangle + e_1 \end{aligned}$$

$$= -\langle \mathbf{h}_1, \mathbf{u} \rangle - \mathbf{h}_1 \mathbf{H}_0^{-1} \mathbf{e}_0 + \langle \mathbf{h}_1, \mathbf{u} \rangle + e_1$$
$$= -\mathbf{h}_1 \mathbf{H}_0^{-1} \mathbf{e}_0 + e_1$$

Let $\mathbf{e} = \mathbf{e}_0$ and $e' = e_1$. Then $b' = \langle \mathbf{h}'_1, \mathbf{e} \rangle + e'$. Thus, we have constructed a new sample $(\mathbf{h}'_1, b')$ which is of the form $(\mathbf{h}'_1, \langle \mathbf{h}'_1, \mathbf{e} \rangle + e')$. In this form, $\mathbf{h}'_1 = -\mathbf{h}_1 \mathbf{H}_0^{-1}$ is the new query vector (still uniformly distributed), and $\mathbf{e} = \mathbf{e}_0$ is derived from the original error distribution $\mathbb{E}^k$, and $e' = -e_1$ is also an error term from $\mathbb{E}$. Effectively, the "secret" part in the transformed sample is now related to the noise distribution.

After this transformation, the RSDP problem can be viewed in terms of recovering a "secret" $\mathbf{e}$ that is itself drawn from a distribution derived from $\mathbb{E}^k$, rather than the original secret $\mathbf{u}$. We lose k samples after the transformation.

2.2 Solvers for RSDP

Several cryptanalytic attempts have recently been made to solve RSDP, which can be categorized into two classes of algorithms: combinatorial and algebraic. On the one hand, combinatorial attacks are derivations of ISD algorithms such as Stern and BJMM. For instance, the authors of CROSS [Bal+24c] relied on the so-called shifted-Stern/BJMM to obtain the parameters for their scheme. On the other hand, algebraic solvers try to exploit the structured noise. In essence, the RSDP is, in this approach, modeled as a system of equations. This was first examined in [Bal+24c], and tighter measurements/bounds were provided in [BM24]. However, both works suggest that, at this point, it is not straightforward to conceive an algebraic solver that outperforms ISD variants (especially, regarding CROSS parameters). It is worth noting that BJMM-like algorithms can be further tailored to the structure of the restricted set $\mathbb{E}$, which results in significant speed-ups in some peculiar cases as in [Bit+23].

Stern-like algorithms

In the CROSS proposal, the authors adapt the Stern/Dumer algorithm [Ste88; Dum91] to solve the RSDP. Let an RSDP instance be given by $(\mathbf{H}, \mathbf{s}, \mathbb{E})$, where $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$ and $\mathbb{E}$ is a multiplicative subgroup of order z in $\mathbb{F}_p$. We are tasked to find $\mathbf{e} \in \mathbb{E}^n$, such that $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$. First, a partial Gaussian Elimination yields $\mathbf{H} \leftarrow \mathbf{UHP}$, with an invertible matrix $\mathbf{U}$ and some permutation $\mathbf{P}$, and we obtain $\mathbf{eH} = \mathbf{s}$ in the form

$$(\mathbf{e}_1, \mathbf{e}_2) \begin{pmatrix} \mathbf{I}_{n-k-\ell} & \mathbf{H}_1 \\ \mathbf{0} & \mathbf{H}_2 \end{pmatrix} = (\mathbf{s}_1, \mathbf{s}_2),$$

where $\mathbf{eP}^{-1} = (\mathbf{e}_1, \mathbf{e}_2) \in \mathbb{F}_p^{k+\ell} \times \mathbb{F}_p^{k+\ell}$ (where ℓ is a parameter that can be optimized). We now have two equations that can be used to construct the solution,

which are

$$\mathbf{e}_1 + \mathbf{e}_2\mathbf{H}_1 = \mathbf{s}_1, \tag{1}$$

$$\mathbf{e}_2\mathbf{H}_2 = \mathbf{s}_2. \tag{2}$$

We now enumerate vectors $\mathbf{e}_2 \in \mathbb{E}^{k+\ell}$ that satisfy (2). Equation (1) verifies that $\mathbf{e}_1$ is part of the permuted (original) solution, that is, $\mathbf{s}_1 - \mathbf{e}\mathbf{H}_1$ is also a vector in $\mathbb{E}^{k+\ell}$. Candidates $\mathbf{e}_2$ can be constructed by a Meet-in-the-Middle approach. In particular, let $\mathbf{e}_2 = (\mathbf{x}_1, \mathbf{x}_2)$ where $\mathbf{x}_i \in \mathbb{E}^{(k+\ell)/2}$, $i = 1..2$, and $\mathbf{H}_2 = \begin{pmatrix}\mathbf{H}_{21}\\ \mathbf{H}_{22}\end{pmatrix}$. We then construct two lists

$$\mathcal{L}_1 = \{(\mathbf{x}_1, \mathbf{x}_1\mathbf{H}_{21}), \mathbf{x}_1 \in \mathbb{E}^{(k+\ell)/2}\},$$

$$\mathcal{L}_2 = \{(\mathbf{x}_2, \mathbf{s}_2 - \mathbf{x}_2\mathbf{H}_{22}), \mathbf{x}_2 \in \mathbb{E}^{(k+\ell)/2}\},$$

and find collisions between them, i.e., $\mathbf{x}_1\mathbf{H}_{21} = \mathbf{s}_2 - \mathbf{x}_2\mathbf{H}_{22}$. Without loss of generality, assume $|\mathcal{L}_1| = |\mathcal{L}_2| = z^{(k+\ell)/2}$. The cost associated with constructing them, denoted by C_1 and C_2, are lower bounded by

$$C_1 = C_2 \geq |\mathcal{L}_1| \cdot \left(\frac{k+\ell}{2} \cdot \log(z) + \ell \cdot \log(p)\right),$$

On average, we find $|\mathcal{L}_1|^2 \cdot p^{-\ell}$ collisions (i.e., candidates) between the two lists. Since we are checking all of them, the cost, denoted by C_{coll}, is estimated to be

$$C_{\text{coll}} \geq |\mathcal{L}_1|^2 \cdot p^{-\ell} \cdot (k+\ell) \cdot \log(p).$$

The algorithm is considered successful if it finds any solutions for the instance. If the problem instance was constructed in such a way that it is known that there is at least one solution, then the expected number of solutions is roughly given by

$$1 + |\{\mathbf{e} \in \mathbb{E}^n : \mathbf{e}\mathbf{H} = \mathbf{s}\}| \approx 1 + z^n p^{k-n}.$$

Similarly to *enumeration-based* ISD algorithms in classical SDP, this adaptation also uses excessive memory for the lists. Therefore, it is reasonable to consider the penalty for accessing such memory. For instance, the author of CROSS employs the log penalty model, which is $\log(\max(|\mathcal{L}_1|, |\mathcal{L}_2|))$. In summary, in this case the complexity of the Stern-like algorithm is estimated as

$$C_{\text{Stern/Dumer}} \geq \frac{C_1 + C_2 + C_{\text{coll}}}{(\text{number.of.solutions})} \times (\text{memory.access.cost}). \tag{3}$$

Recall that, in our Oracle model, the adversary has to find the correct secret (not just any). Moreover, additional queries ensure that the number of solutions is close to 1 (as $z^n p^{k-n}$ gets 'arbitrarily small'). Therefore, we assume that, when applying the Stern algorithm to RSDP Oracle samples, the number of solutions

is 1.

BJMM-like algorithm

The authors of [Bit+23] and [Bal+24c] show that the *representation technique* can also be employed in the enumeration steps. Interestingly, the performance of such adaptations depends on the additive structure of the restricted set $\mathbb{E}$. We will briefly explain the idea and skip the details of the approach. We refer to [Bal+24c] for a more thorough understanding.

Suppose, in Equation 2, we can decompose $\mathbf{e}_2 = \mathbf{e}_1^{(1)} + \mathbf{e}_2^{(1)}$. In contrast to the Stern-like approach, the support of $e^{(1)}$ and $e^{(2)}$ can overlap. Since $\mathbb{E}$ is a multiplicative subgroup, it is not closed under addition. Therefore, within the overlapping position, we have to sample $\mathbf{e}_i^{(1)}$ from some larger search domain defined as $\mathbb{E} \cup \mathbb{D} \cup \{0\}$, where $\mathbb{D}$ is defined as

$$\mathbb{D} := \{e - e' | e, e' \in \mathbb{E}\} \backslash (\mathbb{E} \cup \{0\})$$

The cost of this approach depends on the so-called "linearity" of $\mathbb{E}$ and $\mathbb{D}$. In particular, for a random element $a \in \mathbb{E}$, we define the two quantities

$$\ell_{\mathbb{E}}(a) = |[b \in \mathbb{E} : \exists c \in \mathbb{E}, b + c = a]| \,,$$

$$\ell_{\mathbb{D}}(a) = |[b \in \mathbb{E} : \exists c \in \mathbb{D}, b + c = a]| \,.$$

These values will determine how many representations r for an element during the enumerating process. In particular, an entry of $\mathbf{e}_2$ can be represented as a sum of two elements from $\mathbb{E}$, or those from $\mathbb{E}$ and $\mathbb{D}$. Hence, it implies how much we can save by only enumerating a $1/r$-fraction. It is worth noting that, for certain of p and $\mathbb{E}$ (e.g., in CROSS parameters), the above quantities are independent of a. Similar to the SDP case, one can repeat this step for a vector $\mathbf{e}^{(1)} = \mathbf{e}_1^{(2)} + \mathbf{e}_2^{(2)}$ and so on, increasing the depth of the tree. However, for a fixed code rate of k/n and in the full-weight RSDP setting (as in CROSS parameters [Bal+24c]), this approach does not seem to yield improvements over the Stern/Dumer algorithm.

To improve the performance, the BJMM approach is further tailored: one can solve for a *shifted* instance of RSDP as $(\mathbf{e}+\mathbf{x})\mathbf{H} = \mathbf{s}+\mathbf{s}_\mathbf{x}$, where $\mathbf{s}_\mathbf{x}$ is the syndrome of an element $\mathbf{x} \in \mathbb{E}^n$). In particular, it was found that for the CROSS parameter, shifted-BJMM yields smaller spaces for *shifted* $\mathbb{E}_x$ and $\mathbb{D}_x$ (while having the same linearity), which slightly outperforms Stern/Dumer.

As CROSS have advanced to the second round of the NIST additional post-quantum signature call, it is interesting to perform attacks on the RSDP in broader contexts, as it can inspire many future cryptographic primitives. Therefore, we later investigate our algorithm when the restricted set is, for example, $\mathbb{E} = \{(-2)^i, i = 0, \ldots, 13\}$. It is unclear how the shifted-BJMM proposed by [Bal+24c] will perform due to its reliance on the additive structure of the restricted set (particularly,

the "linearity" is no longer constant). A similar attempt to study RSDP with special restricted error sets has also been made in [Bit+23]. Studying RSDP with different settings improves our understanding and confidence in employing RSDP in cryptography.

The algebraic approach with linearization

The basic Aurora-Ge approach [AG11] involves forming nonlinear equations for each sample of the form

$$\prod_{e\in\mathbb{E}}(\langle \mathbf{h}_i, \mathbf{u}\rangle + e - b_i) = 0,$$

which will be of degree z. Then, the system of equations is transformed into a linear one with around $\sum_{i=1}^{z}\binom{k}{i}$ unknowns (each is a monomial up to degree z of variables u_i in $\mathbf{u}$). Assuming in the order of $\binom{k}{z}$ oracle calls and $z < k/2$, one can then solve for the unknowns by Gaussian elimination in time around $\binom{k}{z}^3$. Better algorithms for solving linear systems of equations can lower the exponent.

Gröbner basis method

More advanced algebraic attacks have been recently studied as an alternative to the traditional ISD algorithms, particularly for specific variants of SDP such as the so-called *Regular* SDP [BØ23]. Therefore, it is natural to extend the investigation to RSDP. Recall that in an RSDP instance, the error $\mathbf{e} = (e_1, \ldots, e_n)$ is drawn from a restricted set $\mathbb{E}$ of order z. Hence, we can solve for

$$\begin{cases} \mathbf{eH} = \mathbf{s}, \\ e_i^z = 1, \quad \text{for } i = 1, \ldots, n. \end{cases}$$

Such a system can be solved using *Gröbner bases* , e.g., via F5 algorithm [BFS15]. The complexity of such an approach depends on the *degree of regularity* d_{reg} of the above system. In [Bal+24c], the author heuristically estimates d_{reg} to be linear in n, and the cost of finding a Gröbner basis is exponential. Additional simulations and tighter analysis by [BM24] further support their claim.

3 The new BKW-style approach

Recall that our problem is given in the RSDP-Oracle model as

$$\mathbf{u}\begin{pmatrix}\mathbf{h}_1 & \ldots & \mathbf{h}_n\end{pmatrix} + \mathbf{e} = \mathbf{b}, \tag{4}$$

where $\mathbf{h}_i \in \mathbb{F}_p^k, i = 1, \ldots, n$ and $\mathbf{b} \in \mathbb{F}_p^n$ are known vectors and $\mathbf{u} \in \mathbb{E}^k, \mathbf{e} \in \mathbb{E}^n$ are the unknowns. We observe that since coordinates in $\mathbf{u}$ and $\mathbf{e}$ belong to

the multiplicative subgroup $\mathbb{E}$, the vector $\mathbf{b}$ will, in general, be biased. Before explaining the steps of our new algorithm, let us introduce additional definitions.

Definition 20. *($\mathbb{E}$-weight) Let p be a prime number, $\mathbb{E}$ be a multiplicative subgroup of $\mathbb{F}_p$, and $-\mathbb{E} := \{-e : e \in \mathbb{E}\}$. For $x \in \mathbb{F}_p$, we define its $\mathbb{E}$-weight, denoted by $\omega_{\mathbb{E}}(x)$, as*

$$\omega_{\mathbb{E}}(x) = 0, \quad \text{if } x = 0,$$

$$\omega_{\mathbb{E}}(x) := \text{argmin}_i\{\sum_{j=1}^{i} e_j = x : e_j \in \mathbb{E} \cup -\mathbb{E}\}, \quad \text{if } x \neq 0,$$

i.e., $\omega_{\mathbb{E}}(x)$ is the smallest number of elements from $\mathbb{E} \cup -\mathbb{E}$ that sums to x. For convenience, let us introduce $\mathbb{E}' = \mathbb{E} \cup -\mathbb{E}$. We can naturally extend this notion to the $\mathbb{E}$-weight of a vector $\mathbf{x} \in \mathbb{F}_p^n$ as

$$\omega_{\mathbb{E}}(\mathbf{x}) = \sum_{i=1}^{n} \omega_{\mathbb{E}}(x_i).$$

This definition also induces a distance, $d_{\mathbb{E}}(\mathbf{x}, \mathbf{y})$, by defining $d_{\mathbb{E}}(\mathbf{x}, \mathbf{y}) = \omega_{\mathbb{E}}(\mathbf{x} - \mathbf{y})$. This defines a metric as $d_{\mathbb{E}}(\mathbf{x}, \mathbf{y}) \geq 0$ with equality only when $\mathbf{x} = \mathbf{y}$; $d_{\mathbb{E}}(\mathbf{x}, \mathbf{y}) = d_{\mathbb{E}}(\mathbf{y}, \mathbf{x})$; $d_{\mathbb{E}}(\mathbf{x}, \mathbf{y}) \leq d_{\mathbb{E}}(\mathbf{x}, \mathbf{z}) + d_{\mathbb{E}}(\mathbf{z}, \mathbf{y})$.

Example 9. *Let $p = 127$ and $\mathbb{E} = \{(2)^i, i = 0, \ldots 6\}$. We have $\omega_{\mathbb{E}}(0) = 0$, and 14 elements of weight 1, $\omega_{\mathbb{E}}(1) = 1, \omega_{\mathbb{E}}(2) = 1, \ldots, \omega_{\mathbb{E}}(63) = 1$. The $\mathbb{E}$-weight of an element x in $\mathbb{F}_p$ is 1 if the Hamming weight of the binary representation of $\mathbf{x}$ or $-\mathbf{x}$ (complemented bits) is 1. Otherwise, it is less straightforward. For example $x = 15$ has $\omega_{\mathbb{E}}(15) = 2$ as $15 = 16 - 1$, but the binary representation of 15 has Hamming weight 3. The maximum weight of an element in $\mathbb{F}_p$ when $p = 127$ is 3.*

Example 10. *Let $p = 127$ and $\mathbb{E} = \{(-2)^i, i = 0, \ldots, 13\}$. In this case $\mathbb{E}' = \mathbb{E} \cup -\mathbb{E} = \mathbb{E}$. The weight of elements in $\mathbb{F}_p$ is the same as above.*

Our new approach is based on two observations related to the new metric.

Observation 1. *Let $e \in \mathbb{E}, h \in \mathbb{F}_p$. Then*

$$\omega_{\mathbb{E}}(eh) = \omega_{\mathbb{E}}(h).$$

Therefore, if we let $\mathbf{e} \in \mathbb{E}^n, \mathbf{h} \in \mathbb{F}_p^n$, we have

$$\omega_{\mathbb{E}}((e_1h_1, e_2h_2, \ldots, e_nh_n)) = \omega_{\mathbb{E}}(\mathbf{h}).$$

The second observation is that

Observation 2. *If $\omega_{\mathbb{E}}(\mathbf{h})$ is low, then a random variable X created from the parity check equations and $\mathbf{e} \in \mathbb{E}^n$, as*

$$X = \sum_{i=1}^{n} e_i h_i = \mathbf{1} \cdot (e_1 h_1, e_2 h_2, \ldots, e_n h_n) = \mathbf{eh},$$

will be biased. The size of the bias will depend on the $\mathbb{E}$-weight of the vector $\mathbf{h}$, to be detailed later.

We will sometimes talk about the weight of a sum, by which we mean the sum of the weights of the terms in the sum, in the above case the same as $\omega_{\mathbb{E}}(\mathbf{h})$.

The basic strategy of the new approach is now to guess a few of the u_i entries in $\mathbf{u}$, recompute the oracle calls based on this guess, and then generate many samples of the form $\sum_{j=1}^{t} b_{i_j} = \mathbf{u} \cdot \sum_{j=1}^{t} \mathbf{h}_{i_j} + \sum_{j=1}^{t} e_{i_j}$. If the weight of $\sum_{j=1}^{t} \mathbf{h}_{i_j}$ is low then the sampled value will show a detectable bias.

The following description of the algorithmic steps leans towards the RSDP setting of CROSS. In particular, we consider $\mathbb{E} = \{2^i, i = 0, \ldots, z\}$, for which we have $\mathbb{E} \neq -\mathbb{E}$. Our algorithmic description follows this slightly more complicated case. Some algorithmic steps can be simplified in the case $\mathbb{E} = \{(-2)^i, i = 0, \ldots, z\}$, for which we have $\mathbb{E} = -\mathbb{E}$. We extend the basic BKW-style approach in several ways, as described in the following subsections.

3.1 Creating many samples and reducing guessing positions.

For every coordinate b_i of $\mathbf{b}$, it is a noisy dot product of the secret $\mathbf{u}$ and a sample $\mathbf{h}_i$. This step aims to create more samples at the cost of introducing more error terms to each equation. It is similar to the reduction step of the BKW algorithm.

Let t, ν and $\delta^{(i)}, i = 1, \ldots, \nu$ be positive integers that are algorithm parameters. Note that ν is analogous to the amount of reduction steps in the BKW algorithm. We first construct 2^ν lists $\mathcal{L}_j^{(0)}$, $j = 1, \ldots, 2^\nu$ which contain combinations of t samples $\mathbf{h}_{i_j}$. The corresponding sum of t values $b_{i_j}, j = 1, \ldots, t$ is updated accordingly. Consequently, it is equivalent to combining t errors for each new equation as

$$\mathbf{u} \cdot (\sum_{j=1}^{t} \mathbf{h}_{i_j}) + \sum_{j=1}^{t} e_{i_j} = \sum_{j=1}^{t} b_{i_j}. \tag{5}$$

To make the idea and later analysis straightforward, as well as to avoid any possible dependency, we assume, for simplicity, that the lists $\mathcal{L}_j^{(0)}$, $j = 1, \ldots, 2^\nu$ are each constructed from M different samples from oracle calls (i.e., we have $2^\nu \cdot M$ oracle calls). Let us call the newly obtained combinations as t-error samples.

Note that we can benefit from the multiplicative structure of $\mathbb{E}$ to increase the amount of t-error samples. For every sample $\mathbf{h}_{i_j}$ in Equation (5), we can multiply with any fixed element in $\mathbb{E}$ as our corresponding noise terms are still seen as

elements of $\mathbb{E}$. So we then get samples

$$\mathbf{u} \cdot (\sum_{j=1}^{t} d_j \mathbf{h}_{i_j}) + \sum_{j=1}^{t} d_j e_{i_j} = \sum_{j=1}^{t} d_j b_{i_j}, \tag{6}$$

where $d_j \in \mathbb{E}, j = 1, \ldots, t$.

Ultimately, we aim to generate $t \cdot 2^\nu$-error samples, of which $\sum_{i=1}^{\nu} \delta^{(i)}$ fixed positions in the parity check part are all zeroes. We achieve this by a tree-like procedure depicted in Figure 1. In particular, the vectors in the top lists $\mathcal{L}_j^{(0)}$, $j = 1, \ldots, 2^\nu$, are sorted by their first $\delta^{(1)}$ positions. Then, we merge pairs, i.e. sum (or subtract) one vector in $\mathcal{L}_1^{(0)}$ and one in $\mathcal{L}_2^{(0)}$, etc., so that the elements in the merged lists $\mathcal{L}_j^{(1)}$, $j = 1, \ldots, 2^{\nu-1}$, are all 0 in their last $\delta^{(1)}$ positions. The same procedure can be repeated (on next layers) with $\delta^{(2)}, \delta^{(3)}$, and so on, and the number of lists in each layer is halved. In summary, this procedure, that we denote create_sample(), takes $2^\nu \cdot M$ RSDP-Oracle calls as input and it outputs a list $\mathcal{L} = \{(\widehat{\mathbf{h}}, \widehat{b})\}$ of $2^\nu \cdot t$-error samples with their corresponding updated values of the noisy product.

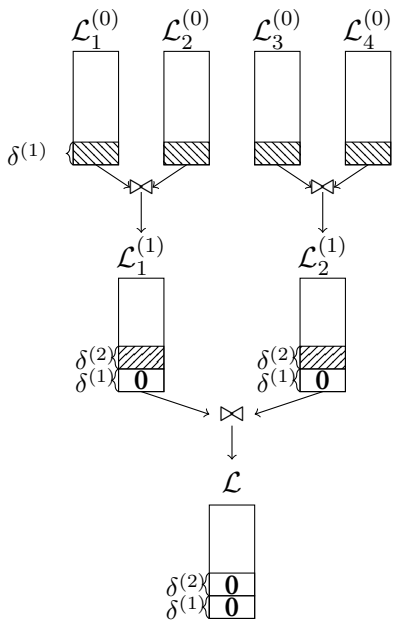


Figure 1: create_sample() with $\nu = 2$ reduction steps. After step i, we cancel $\delta^{(i)}$ last positions of samples in the lists.

We have achieved two things: With proper choice of $\delta^{(i)}$ and ν, we inflate the number of available samples for guessing in later stages, and we have reduced the

dimension of the guess space of the secret from k to $k - \sum_{i=1}^{\nu} \delta^{(i)}$.

Dependencies

Assume $\nu = 2$ as in Figure 1. Due to the multiplicative structure, any $2t$ combination in the middle layer $\mathcal{L}_i^{(1)}$ will have z collinear copies (i.e., they only differ by a scalar in $\mathbb{E}$). Hence, the number of "effective" samples in this list is only a $1/z$-fraction of all combinations. This can be avoided by precomputing all the t-tuples of coefficients in a table and only combining samples in, e.g., $\mathcal{L}_i^{(0)}$, if their corresponding coefficient vectors are not collinear. Note that once we eliminate such redundant copies in $\mathcal{L}_i^{(1)}$, the same kind of dependency cannot occur further down the tree.

Remark 2. *In the merge step $\bowtie$, subtracting samples from lists yields errors from both $\mathbb{E}$ and $-\mathbb{E}$. However, it does not pose a problem for us. As we will see later, what is crucial is that we have the same error profile in each $2^\nu \cdot t$-error sample. This allows us to gauge the noise bias in the end precisely.*

Remark 3. *Since the top lists are assumed to be independent batches of samples, it is possible to have $d_j \in \mathbb{E} \cup -\mathbb{E}$ (i.e., more samples). Following Remark 2, we want to have homogeneity of error profiles in the final samples in $\mathcal{L}$, which requires a subroutine to ensure. Therefore, we restrict ourselves to $\mathbb{E}$, as described above, and investigate this possibility in Section 5.*

3.2 The Covering Code Method

After create_sample(), every equation of type (4) now has the form

$$\mathbf{u} \cdot \widehat{\mathbf{h}} + \mathcal{N} = \widehat{b}, \tag{7}$$

where $\mathcal{N}$ denotes the error term coming from the sum of $2^\nu \cdot t$ unknown elements in $\mathbb{E}$ with known coefficients in $\{-1, 1\}$ and

$$\widehat{\mathbf{h}} = \big(\underbrace{* \quad \cdots \quad *}_{k-\delta} \quad \underbrace{0 \quad \cdots \quad 0}_{\delta}\big),$$

where $*$ means that the position can take any value in $\mathbb{F}_p$. As we have canceled out the contribution of some secret values in $\mathbf{u}$, we are tasked with determining only $k - \delta$ values of $\mathbf{u}$. Assume we know the bias coming from $\mathcal{N}$. Then, one can estimate roughly how many samples $\widehat{\mathbf{h}}$ are needed to uniquely determine the secret using guessing and then verifying the bias. This is most efficiently done through the *Discrete Fourier Transform* (DFT). However, a direct application of the Discrete Fourier Transform with the current dimension can be prohibitively expensive in many cases.

Moving forward, we now assume that we remove the zero parts from (7) without changing our notation. So assume now that $\mathbf{u} \in \mathbb{F}_p^{k'}$, where $k' = k - \delta$ and each $\widehat{\mathbf{h}} \in \mathbb{F}_p^{k'}$.

Instead of direct guessing, we proceed with the help of the *covering codes* technique. We project the coordinates of $\widehat{\mathbf{h}}$ onto a $\mathbb{F}_p$ linear code $\mathcal{C}$ of length k'. In particular, we use here a direct sum of $[\ell, 1]$ repetition codes over $\mathbb{F}_p$ (length ℓ and dimension 1), assuming k' is a multiple of ℓ. Codewords in a repetition code are of the form $(c_1, c_1, \ldots, c_1)$ for $c_1 \in \mathbb{F}_p$. For a vector $\widehat{\mathbf{h}}$, let $\widehat{\mathbf{h}} = \widehat{\mathbf{c}} + \widehat{\mathbf{e}}$, where $\widehat{\mathbf{c}}$ is the closest codeword in $\mathcal{C}$ and $\widehat{\mathbf{e}}$ is the corresponding error, measured using $\omega_{\mathbb{E}}(\widehat{\mathbf{e}})$. We can rewrite (7) as

$$\mathbf{u} \cdot \widehat{\mathbf{c}} + \mathbf{u} \cdot \widehat{\mathbf{e}} + \mathcal{N} = \mathbf{u} \cdot \widehat{\mathbf{c}} + \mathcal{N}' + \mathcal{N} = \widehat{b}, \tag{8}$$

where $\mathcal{N}' = \mathbf{u} \cdot \widehat{\mathbf{e}}$ is a low weight sum of elements from $\mathbb{E}$. Let $\ell' = k'/\ell$, we have the following lemma.

Lemma 9. *Let $\widehat{\mathbf{c}}$ be a codeword in $\mathcal{C}$ from an $[\ell, 1]$ repetition code and $\mathbf{c} = (c_1, c_2, \ldots, c_{\ell'})$ be its corresponding information word. There exists a unique $\mathbf{u}' \in \mathbb{F}_p^{\ell'}$ such that*

$$\mathbf{u} \cdot \widehat{\mathbf{c}} = \mathbf{u}' \cdot \mathbf{c}.$$

Proof. For $i = 1, \ldots, \ell'$, define

$$u'_i = u_{i+\ell(i-1)} + u_{(i+1)+\ell(i-1)} + \ldots + u_{(i+\ell'-1)+\ell(i-1)}. \tag{9}$$

Then

$$\mathbf{u} \cdot \widehat{\mathbf{c}} = \mathbf{u} \cdot (c_1, \ldots, c_1 || \ldots || c_{\ell'}, \ldots, c_{\ell'}) = \sum_{i=1}^{\ell'} u'_i c_i = \mathbf{u}' \cdot \mathbf{c}.$$

□

The guessing problem is now reduced to finding $\mathbf{u}' \in \mathbb{F}_p^{\ell'}$ such that it supports a large set of samples of the form

$$\mathbf{u}' \cdot \mathbf{c} + \mathcal{N}' + \mathcal{N} = \widehat{b}, \tag{10}$$

where $\mathbf{c}$ and $\widehat{b}$ are known values and $\mathcal{N}' + \mathcal{N}$ are biased noise variables for each created sample.

To summarize this step, denoted by covering_codes(), we have further reduced the cost of guessing while introducing additional noise $\mathcal{N}'$. Assume that the average $\mathbb{E}$-weight of $\widehat{\mathbf{e}}$ is t'. Since $\mathbf{u} \in \mathbb{E}$, we can treat $\mathcal{N}'$ as the sum of t' elements in $\mathbb{E}'$, as $\mathbb{E}' = \mathbb{E} \cup -\mathbb{E}$ is also closed under multiplication. Therefore, in total, $\mathcal{N} + \mathcal{N}'$ correspond to summing $\omega = 2^\nu \cdot t + t'$ elements from $\mathbb{E}$ with

coefficients in $\{-1, 1\}$. Consider a biased random variable X constructed as

$$X = \mathcal{N} + \mathcal{N}' = \sum_{j=1}^{2^\nu \cdot t} d_j e_{i_j} + \sum_{j=1}^{t'} \widehat{e}_{i_j} u_{i_j},$$

where $e_{i_j}, u_{i_j} \in \mathbb{E}$ are the unknowns and $d_j, \widehat{e}_{i_j}$ are fixed known values. For each term in $\mathcal{N} + \mathcal{N}'$, it may come from different "error patterns" by which we mean how they are decomposed into elements of $\mathbb{E}$ and $-\mathbb{E}$. Consequently, different error patterns have different final error distributions and make things harder to analyze. We will now investigate the bias for the overall noise more closely.

3.3 Estimation of the noise

The description now follows the case $\mathbb{E} \neq -\mathbb{E}$. For the other case, $\mathbb{E} = -\mathbb{E}$, there is only a single distribution to consider.

Noise distribution

To guess the correct $\mathbf{u}'$ in Equation (8), one needs to know the exact distribution of the noise term $\mathcal{N} + \mathcal{N}'$. In particular, we are investigating the distribution of summing $\omega = 2^\nu \cdot t + t'$ elements from $\mathbb{E}$ and $-\mathbb{E}$. Moreover, assume that the combined noise can be represented as the sum of ω_1 and ω_2 elements of $\mathbb{E}$ and $-\mathbb{E}$, respectively. Let X be a random variable defined as

$$X = \sum_{\substack{i=1 \\ n_i \in \mathbb{E}}}^{\omega_1} n_i + \sum_{\substack{i=1 \\ n'_i \in -\mathbb{E}}}^{\omega_2} n_i, \tag{11}$$

where $n_i \in \mathbb{E}$ for $i = 1, \ldots, \omega_1$ and $n'_i \in -\mathbb{E}$ for $i = 1, \ldots, \omega_2$. We can tune create_sample() and covering_codes() so that we only consider a particular (e.g., most probable) pattern by fixing ω_1 and ω_2. Knowing the exact pattern, we can rely on computer simulation to estimate precisely the accumulated distribution after summing ω_1 and ω_2 random elements from $\mathbb{E}$ and $-\mathbb{E}$, respectively.

Let χ be the distribution for X in Equation (11). In addition to the noise distribution, it is crucial to estimate the deviation of χ from the uniformly random distribution U. To achieve this goal, we use the following result used also in [BJV04].

Lemma 10 (Square Euclidean Imbalance). *Let χ be a distribution on $\mathbb{Z}_p$ and U be the uniform distribution. For $X \leftarrow \chi$, we write $\Pr_\chi[x]$ as the probability of $X = x$. Define $\epsilon_z = \Pr_\chi[x] - \frac{1}{p}$ for $x \in \mathbb{Z}_p$. The* Squared Euclidean Imbalance *(SEI)*

$\Delta(\mathcal{D}_0)$ of a distribution $\mathcal{D}_0$ from the uniform distribution over $\mathbb{Z}_p$ is defined as

$$\Delta(\chi) = p \sum_{z \in \mathbb{Z}_p} \epsilon_z^2.$$

Then, to distinguish χ from U, we need

$$n = \mathcal{O}(\frac{1}{\Delta(\chi)})$$

samples from χ.

Furthermore, assume that we are considering a process that generates independent random variables depending on some variable $K \in \mathcal{K}$. We assume that for one unknown value, $K = K_0$, all generated samples follow distribution χ, whereas when $K \neq K_0$ all samples follow the uniform distribution. From the previous lemma, we are required to have

$$n = \frac{d}{\Delta(\chi)},$$

where d is a constant factor, samples to distinguish K_0 from all other "wrong" $K \in \mathcal{K}$. Let $\Phi(t)$ denote the distribution function of the standard normal distribution, i.e.,

$$\Phi(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} e^{-\frac{x^2}{2}}\, dx.$$

Then, from [BJV04], we can successfully differentiate K_0 from all other candidates with probability[1]

$$P_{\text{success}} = \left(1 - \Phi(\sqrt{d/2})\right)^{|\mathcal{K}|-1}. \tag{12}$$

3.4 Subspace Hypothesis Testing

Until now, we presumably know the noise distribution of Equation (8). To identify the correct key, we employ the p-ary subspace hypothesis testing technique introduced in [GJS15]. This approach extends the binary subspace hypothesis testing method proposed in [GJL14; GJL20], which was originally developed to address the LPN problem.

Similarly to the technique used in [GJS15], we can group all processed samples $(\widehat{\mathbf{h}}_i, \widehat{b}_i)$ in sets $L(\widehat{\mathbf{c}})$ according to their nearest codeword $\widehat{\mathbf{c}}$ and define the function $f_L^{\widehat{\mathbf{c}}}(X)$ as

$$f_L^{\widehat{\mathbf{c}}}(X) = \sum_{(\widehat{\mathbf{h}}_i, \widehat{b}_i) \in L(\widehat{\mathbf{c}})} X^{\widehat{b}_i \bmod p}.$$

[1] It can be approximated as $\exp(-|\mathcal{K}| \cdot e^{-d/4}/\sqrt{2\pi})$, when d is large.

According to Lemma 9, the function $f_L^{\widehat{\mathbf{c}}}(X)$ can be re-written to be a function of the information word $\mathbf{c}$, denoted by $h_{\mathbf{c}}(X) = f_L^{\widehat{\mathbf{c}}}(X)$. We define

$$H_{\mathbf{y}}(X) = \sum_{\mathbf{c} \in \mathbb{F}_p^{\ell'}} h_{\mathbf{c}}(X) \cdot X^{-\langle \mathbf{y}, \mathbf{c} \rangle}.$$

and exhaust all the $p^{\ell'}$ possible values of the vector $\mathbf{y}$. There exists a unique vector $\mathbf{y} \in \mathbb{F}_p^{\ell'}$ corresponding to the unknown $\mathbf{u}'$ such that $\langle \mathbf{y}, \mathbf{c} \rangle = \langle \mathbf{u}, \widehat{\mathbf{c}} \rangle$. For a correct guess, the polynomial $H_{\mathbf{y}}(X)$ thus captures the occurrences of the error symbols, which are distributed according to the assumed error distribution. Otherwise, it is presumed to be uniformly distributed.

The computation of the polynomial $H_{\mathbf{y}}(X)$ can be efficiently performed using the Fast Fourier Transform (FFT). Let ω be a primitive p-th root of unity in the complex field $\mathbb{C}$. The polynomial $H_{\mathbf{y}}(X)$ can be reconstructed if its values at p distinct points $(1, \omega, \omega^2, \ldots, \omega^{p-1})$ are known, with a computational complexity of approximately $\mathcal{O}(p \log_2(p))$. Consequently, the problem is reduced to evaluating the polynomial at these points.

Initially, we evaluate $p^{\ell'}$ polynomials $h_{\mathbf{c}}(X)$ at the p points $(1, \omega, \omega^2, \ldots, \omega^{p-1})$, which requires a complexity of $\mathcal{O}(p^{\ell'} \cdot p \log_2(p))$. After these values are precomputed and stored, the polynomial $H_{\mathbf{y}}(X)$ can be evaluated using p FFTs, each with a complexity of $\mathcal{O}(p^{\ell'} \log_2(p^{\ell'}))$. Once the occurrences are determined, statistical tests can be performed for all $p^{\ell'}$ hypotheses, and the one most aligned with the assumed noise distribution is selected. This step incurs an additional cost of $p^{\ell'+1}$ operations over $\mathbb{F}_p$.

Algorithm description

We describe our algorithm using pseudocode in Algorithm VI.1. The Subspace Hypothesis Testing in the previous Section is denoted as sht().

The algorithm recovers only a part of the key $\mathbf{u}'$. However, if this part is correctly recovered, one can repeat the procedure on the same problem but now with a lower original dimension. This is clearly much easier to solve compared to the original problem, so the complexity of this first run will dominate the overall complexity of recovering the full secret.

4 Complexity Analysis

In this section, we go through our algorithm complexity by revisiting each step described in Section 3. Let $t, \nu, \delta^{(i)}$ and ℓ be parameters specified in Algorithm VI.1. The complexity mainly consists of the costs of its sub-procedures: sample_create(), covering_codes(), and sht(). For a fair comparison, we present our analysis in a similar fashion as the method employed by the authors in [Bal+24c]. In

Algorithm VI.1: RSD solver

Input: R-SDP Oracle queries: $(\mathcal{M} := \{\mathbf{h}_i, b_i = \mathbf{h}_i \cdot \mathbf{u} + e_i)\}$, where $\mathbf{u} \in \mathbb{E}^k, \mathbf{h}_i \in \mathbb{F}_p^k, e_i \in \mathbb{E}$.

Parameters: $\nu, t, (\delta^{(i)})_{i=1}^{\nu}, \ell$,
$|\mathcal{M}| := 2^\nu \cdot M$,
$\delta := \sum_{i=1}^{\nu} \delta^{(i)}$,
$\ell' := (t - \delta)/\ell$ as an integer.

Output: A guess of the partial secret $\mathbf{u}'$.

$\mathcal{L} \leftarrow$ sample_create$(\mathcal{M}, t, \nu, (\delta^{(i)}))_{i=1}^{\nu} = \{(\widehat{\mathbf{h}}_i, \widehat{b}_i)\}$;
$\mathcal{C} \leftarrow$ Direct sum of ℓ' $[\ell, 1]$ repetition codes over $\mathbb{F}_p$ with an average decoding error weight t';
$\omega := 2^\nu \cdot t + t' = \omega_1 + \omega_2$; // ω_1 terms from $\mathbb{E}$, ω_2 terms from $-\mathbb{E}$.
$\Delta \leftarrow$ Estimated Square Euclidean Distance when combining $\omega_1 + \omega_2$ elements in $\mathbb{E}$ and $-\mathbb{E}$, respectively;
$P \leftarrow$ the distribution obtained when combining $\omega_1 + \omega_2$ elements in $\mathbb{E}$ and $-\mathbb{E}$, respectively;
forall $i = 1, \ldots, |\mathcal{L}|$ *with* $(\widehat{\mathbf{h}}_i, \widehat{b}_i) \in \mathcal{L}$ **do**
  $(\widehat{\mathbf{c}}_i, \widehat{\mathbf{e}}_i) \leftarrow$ covering_codes$(\widehat{\mathbf{h}}_i, \mathcal{C})$; $\mathcal{L}_{\mathsf{sht}} \leftarrow \emptyset$; **if** $\omega_{\mathbb{E}}(\widehat{\mathbf{e}}_i) = t'$ *and* $\widehat{\mathbf{e}}_i$ *is correctly decomposed in* $\mathbb{E}$ *and* $-\mathbb{E}$ **then**
    $\mathbf{c}_i \leftarrow$ information word of $\widehat{\mathbf{c}}_i$;
    $\mathcal{L}_{\mathsf{sht}} \leftarrow \mathcal{L}_{\mathsf{sht}} \cup \{\mathbf{c}_i, \widehat{b}_i\}$;

Return: $\mathbf{u}' \leftarrow \mathsf{sht}(\mathcal{L}_{\mathsf{sht}}, P, \Delta)$

particular, we also employ the log memory cost model to penalize our algorithm analysis.

4.1 The sample_create() step

The sample_create() first create 2^ν lists $\mathcal{L}_j^{(0)}$ of t-error samples, then use a hashing strategy repeatedly to get the lists $\mathcal{L}_j^{(i)}$, $i = 1, \ldots, \nu$, of $2^{(i)} \cdot t$-error samples of which $\delta = \sum_{i=1}^{\nu} \delta^{(i)}$ positions are canceled out. Without loss of generality, we can assume that, in each layer of the tree, the size of lists are roughly equal and denoted by $|\mathcal{L}^{(i)}|, i = 1, \ldots \nu - 1$, and the final list is simply $\mathcal{L} := \mathcal{L}^{(\nu)}$.

The computational complexity of sample_create(), denoted by $C_{\mathsf{sample\_create}}$,

equivalently to building the tree of depth ν, is lower bounded by

$$C_{\mathsf{sample\_create}} \geq \sum_{i=0}^{\nu-1} 2^{\nu-i} \cdot |\mathcal{L}^{(i)}|(k \cdot \log p + \delta^{(i+1)} \cdot \log p) + |\mathcal{L}^{(\nu)}| \cdot k \cdot \log p. \quad (13)$$

where $|\mathcal{L}^{(0)}| = |\mathbb{E}|^t \cdot \binom{M}{t}$, $|\mathcal{L}^{(1)}| = |\mathcal{L}^{(0)}|^2 \cdot p^{\delta_1} \cdot z^{-1}$ (recall Section 3.1), and for $i = 2, \ldots, \nu$, $|\mathcal{L}^{(i)}| = |\mathcal{L}^{(i-1)}|^2 \cdot p^{-\delta^{(i)}}$.

Indeed, the cost of building 2^ν initial lists of t-error samples is lower bounded by

$$2^\nu \cdot |\mathcal{L}^{(0)}| \left(k \cdot \log p + \delta^{(1)} \cdot \log p\right),$$

where $|\mathcal{L}^{(0)}| = |\mathbb{E}|^t \cdot \binom{M}{t}$ as each t-sample is obtained from combining t oracle calls and multiplying with elements in $\mathbb{E}$. Each sample is of size k, and they are sorted according to $\delta^{(1)}$ positions; hence, we need at least $k \cdot \log p + \delta^{(1)} \cdot \log p$ binary operations. On average, we have roughly $|\mathcal{L}^{(1)}| = |\mathcal{L}^{(0)}|^2 \cdot p^{-\delta^{(1)}} \cdot z^{-1}$ collisions for each pair, and we again sort them according to $\delta^{(2)}$ positions. The same arguments apply to later layers. Lastly, for the bottom layer, we have the list $\mathcal{L}$, and the cost here is at least

$$|\mathcal{L}^{(\nu-1)}|^2 \cdot p^{-\delta^{(\nu)}} \cdot k \cdot \log p.$$

4.2 The covering_codes() step

In this step, we look at every sample of $\mathcal{L}$ and replace them with the closest codeword in the concatenated repetition code $\mathcal{C}$. Recall that in Section 3.2, we decode blocks of length ℓ to its nearest codeword(s). To efficiently decode our samples, we can construct, in advance, a query table that stores the nearest codeword(s) of $\mathbb{F}_p^\ell$ vectors. Then, we can retrieve the nearest codeword for every sample by parsing them into $\ell' = k'/\ell$ blocks and checking the table for each block.

The cost of this step, denoted by $C_{\mathsf{covering\_codes}}$, revolves around building the table of p^ℓ items and decoding the list $\mathcal{L}$. For each item in the table, one goes through all p codewords in the $[\ell, 1]$ repetition code over $\mathbb{F}_p$ and computes the error and its weight. Therefore, the cost of constructing the table is, up to a constant, $p \cdot |\mathbb{F}_p^\ell| = p^{\ell+1}$.

For decoding on samples in $\mathcal{L}$, we need to decode ℓ' blocks for each sample. Hence, it takes $\ell' \cdot |\mathcal{L}|$. In conclusion, the complexity of the covering_code step is

$$C_{\mathsf{covering\_codes}} = \mathcal{O}(p^{\ell+1} + \ell' \cdot |\mathcal{L}|). \quad (14)$$

Again, in terms of concrete complexity, decoding the list $\mathcal{L}$ has no additional cost. However, when constructing the table, computing the error and its weight constitutes $2 \cdot \ell \cdot \log(p)$ operations.

The number of samples for guessing

As we have discussed, the average weight of the error in covering_codes() is t'. Recall that, in the described algorithm, we assume to select codewords of which the corresponding error has $\mathbb{E}$-weight t' and can be decomposed equally to $\mathbb{E}$ and $-\mathbb{E}$. Assume each entry of an error is decomposed into $\mathbb{E}$ and $-\mathbb{E}$ with probability $1/2$ each. Let us denote by $\mathcal{L}_{\mathsf{sht}}$ the list of candidates for sht(). Then, on average, we can keep

$$|\mathcal{L}_{\mathsf{sht}}| = |\mathcal{L}| \cdot \binom{t'}{t'/2} \cdot 2^{-t'} \cdot \Pr[\text{error weight} = t']$$

samples for the guessing steps. To compute the probability of having errors of weight t', we can do as follows: first, we compute the distribution of error weight in the $[\ell, 1]$ block code, and we can apply the convolution of probability mass functions to have the error weight distribution when concatenating ℓ' blocks.

4.3 The sht() step

As described before, in this step, we use the DFT transform and statistical tests to recover the most likely guess for the (partial) secret $\mathbf{s}$. Since we are looking at $\mathbf{u} \in \mathbb{F}_p^{\ell'}$, the cost of this step is

$$C_{\mathsf{sht}} = C_{\mathrm{FFT}} \cdot p^{\ell'+1}(\ell'+1)\log_2 p + p^{\ell'+1}, \tag{15}$$

where C_{FFT} is constant for one FFT and is typically set to 1.

4.4 The success probability of our algorithm

From Definition 10, we can estimate our success probability, called P_{success} as follows. The number of samples for the subspace hypothesis testing is $|\mathcal{L}_{\mathsf{sht}}|$. Assuming the Square Euclidean Distance between the uniform distribution and the distribution when randomly combining $\omega = 2t + t'$ elements in $\mathbb{E}'$ is Δ. Then, let

$$d = |\mathcal{L}_{\mathsf{sht}}| \cdot \Delta$$

and one obtains the success probability given as in Equation (12).

Theorem 6. *The complexity of the described algorithm for RSDP is lower bounded as*

$$C_{\mathsf{RSD\_solver}} \geq P_{\text{success}}^{-1} \cdot (C_{\mathsf{sample\_create}} + C_{\mathsf{covering\_code}} + C_{\mathsf{sht}}) \cdot \log(\mathsf{memory}) \tag{16}$$

where P_{success}^{-1}, $C_{\mathsf{sample\_create}}$, $C_{\mathsf{covering\_code}}$, C_{sht}, *are defined in Equations (12), (13), (14), (15), and*

$$\mathsf{memory} := \max_{i=}(|\mathcal{L}^{(i)}|).$$

5 Applications

This section presents a case study of the CROSS NIST-I parameters, which are $k = 76$, $p = 127$ and $\mathbb{E} = \{2^i, i = 0, \ldots, 6\}$. Moreover, the number of Oracle calls for this parameter set in CROSS is fixed as $M = 126$, which can pose a significant challenge for the dual approach. Therefore, we consider a variable number of oracle calls and investigate the threshold of M when our approach can offer an improvement over the combinatoric approach. According to [Bal+24c], the best algorithm for this parameter set is shifted-BJMM, which requires at least 2^{143} bit operations.

As discussed in Section 2.2, a threshold (for the number of oracle calls) exists above which algebraic solvers are the best option to solve RSDP. Therefore, we are interested in the scenario where the number of allowed oracle calls is under the said threshold.

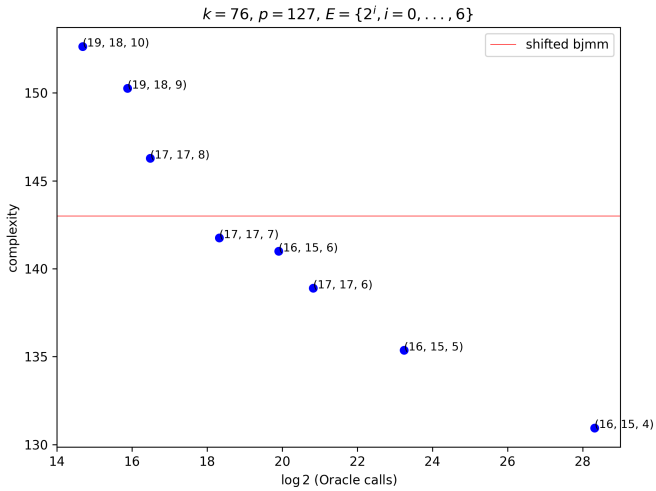
Example 11. *Let $\nu = 1$, t, and δ be our algorithm parameters, and we employ a direct sum of $[3, 1]$ repetition codes for* covering_codes*(). Simulation shows that the average error weight in the covering code steps is 3.95. Therefore, when using the proposed algorithm parameters, we have an expected total weight of $\omega = 2t + 3.95 \cdot \frac{k-\delta}{3}$.*

Let $\delta = 22, t = 6$, resulting in samples of weight $\omega = 83$. This yields the value of SEI as $\Delta \approx 2^{-111.45}$. For these values of parameters, we find that $M \approx 2^{23.1}$ yields the best results at $C \approx 2^{157}$, of which $C_{\text{create\_sample()}}$ is the dominating term.

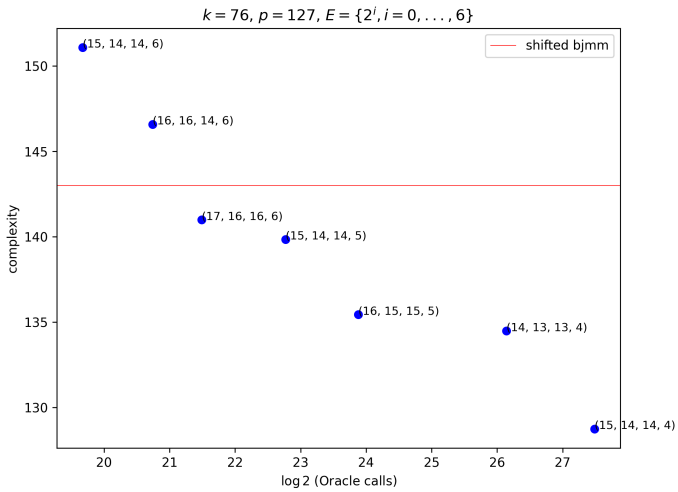
Example 12. *In the previous example, we have an "imbalanced" error profile as the covering code steps contribute 71 out of 83 error terms. Moreover, a high value for δ imposes a high requirement for the number of samples as the SEI is typically very small. Therefore, an extra reduction step, e.g., $\nu = 2$ with smaller $\delta^{(1)}$ and $\delta^{(2)}$, allows the algorithm to work with fewer oracle calls, which can offer significant improvements as the costs of building lists are the main contributor to the complexity.*

- *Let $\delta^{(1)} = 17, \delta^{(2)} = 17$, and $t = 6$. In the covering code step, we continue to employ a direct sum of 14 $[3, 1]$ repetition codes, giving a total error weight of $\omega = 4 \cdot t + 14 \cdot 3.95 \approx 79$ and $\Delta \approx 2^{-105.9}$. We find that $M \approx 4 \cdot 2^{18.81}$ yields $C \approx 2^{138.97}$, where $C_{\text{create\_sample}} = 2^{131.89}, C_{\text{covering\_codes}} = 2^{122.61}, C_{\text{sht}} = 2^{111.55}$, and using 2^{119} bits in memory. The success probability is roughly 1 as the ratio of samples and Δ is 106.*

In Figure 2, we demonstrate how the complexity of our algorithm with $\nu = 2, 3$ as a function of the allowed Oracle calls for $k = 76, z = 127$, and $\mathbb{E} = \{2^i, i = 0, \ldots, 6\}$. Similarly to the example, we also use a direct sum of $[3, 1]$ repetition codes. The label, e.g., $(16, 15, 6)$, represents the optimized values of $(\delta^{(1)}, \delta^{(2)}, t)$. One can see a trend that more oracle calls allow for smaller t, which yields a smaller noise through a decreased ω. We notice a time-memory trade-off between depth-2 and depth-3. Moreover, the difference in improvement is



(a) $\nu = 2$



(b) $\nu = 3$

Figure 2: Scatter plot of our algorithm complexity (with $\nu = 2, 3$ reduction steps) as a function of allowed Oracle calls.

less significant than that of depth-2 and depth-1. An algebraic solver requires roughly $\sum_{i=1}^{7}\binom{k}{i} \approx 2^{31}$ oracle calls and solves the RSDP instance in at most 2^{93} operations with plain Gaussian elimination. Hence, our range of investigation is below 2^{31} oracle calls.

We observe that the memory needed for the algorithm decreases as the number of available oracle calls increases (due to t becoming smaller). For example, for the depth-3 instantiation, it ranges from 2^{124} bits to 2^{108} bits in the regime where our approach outperforms shifted-BJMM, which uses roughly 2^{116} bits in memory.

5.1 RSDP instances with larger restricted error sets

We now study the case when $\mathbb{E} = \{(-2)^i, i = 0, \ldots, 13\}$. As discussed in Section 2.2, applying shifted-BJMM may not be as straightforward. Moreover, the algebraic approach also suffers due to the increase in polynomial degree. Therefore, the Stern-like algorithm seems to be a more reliable tool for estimating the problem hardness in this scenario.

Our algorithm is adaptable to the new situation as we are only concerned with the error distribution through various steps of our algorithms. For example, a change from $z = 7$ to $z = 14$ as above yields a more straightforward analysis as we are no longer requiring a particular error pattern (equal number of positive and negative coefficients) in covering_codes() (since now $\mathbb{E} = -\mathbb{E}$); hence, more samples for testing. A larger error set makes the guessing more challenging as the noise distribution is much closer to the uniform distribution. However, it also allows us to create more samples with create_sample(). Figure 3 shows the performance of our algorithm within this new setting. For example, we choose an RSDP instance with $z = 14$ that requires roughly 147-bit complexity with the Stern algorithm. As a result of the bigger restricted set, one can achieve this hardness with smaller k, e.g., $k = 50$.

We can see that the proposed BKW-style algorithm enjoys even more impressive improvement here. For example, it starts to outperform Stern with fewer oracle calls (compared to Figure 2b) and drops to 110-bit with 2^{28} Oracle calls. Note that, $\sum_{i=1}^{z}\binom{k}{i} \approx 2^{40}$ is the limit when known algebraic attacks can be applied, so we are well below this limit in our investigation.

6 Verification with a small experiment.

We present our experimental results with very small parameters to rectify our heuristic arguments. In particular, it is paramount that the final noise distribution in Equation (10) is modelled correctly. Moreover, in covering_code(), the samples we obtained with predetermined error patterns must be consistent with our theory.

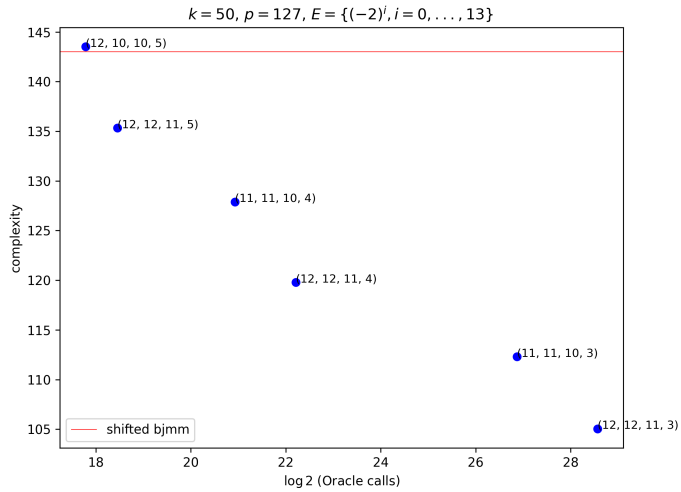


Figure 3: The scatter plot of our algorithm with $\nu = 3$ when $z = 14$.

Experiment 1.

We set up a known-answer test. Let $k = 7, p = 127$ and $\mathbb{E} = \{1, 2, 4, 8, 16, 32, 64\}$. The algorithm is instantiated with $t = 2, \nu = 1, \delta = 1$, which yields weight 4 error samples after create_sample() . Note that these newly created samples all have 0 in their last position. In the covering_code() step, we employ two $[3, 1]$ repetition codes. On average, each error block in the code adds 3.95 $\mathbb{E}$-weight; hence, we select codewords for which the corresponding error has weighted 8. Adding that the coefficients in the error are equally decomposed into $\mathbb{E}$ and $-\mathbb{E}$, this configuration yields a noise term that can be written as a sum of 8 elements from $\mathbb{E}$ and 4 from $-\mathbb{E}$. The probability of obtaining weight-8 errors in covering_codes() with the desirable error pattern is 0.273 and 0.352, respectively.

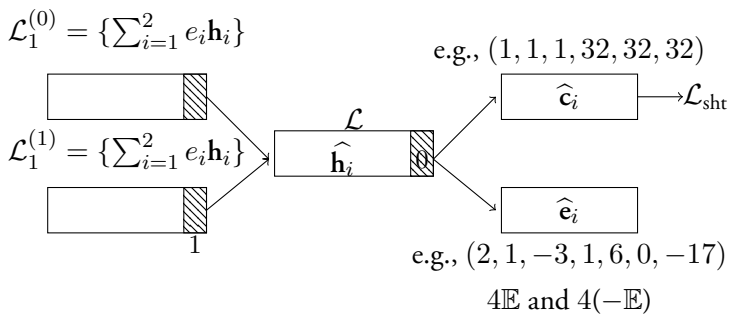


Figure 4: Visualization of create_sample() and covering_codes() steps

	$M = 45$	$M = 50$	$M = 55$
$\lvert L^{(0)}\rvert$	$15.56(15.56)$	$15.87(15.87)$	$16.15(16.15)$
$\lvert L\rvert$	$21.33(21.33)$	$21.95(21.95)$	$22.51(22.51)$
$\lvert L_{\mathsf{sht}}\rvert$	$17.83(17.95)$	$18.39(18.57)$	$18.89(19.12)$

Table 1: The sizes (in log) of samples in various steps of our algorithm according to heuristic estimation and experiments.

Let the (partial) secret vector be $\mathbf{u} = (u_1, \ldots, u_6)$. Then, with the correct guess of $\mathbf{u}' = (u'_1, u'_2) = (u_1 + u_2 + u_3, u_4 + u_5 + u_6)$, we expect to obtain a noise distribution $\mathbf{p}_{\text{correct}}$ that can be distinguished from those that come from all other (wrong) guesses corresponding to $\mathbf{p}_{\text{random}}$.

In particular, for a noise term as described above, we obtain an "ideal" distribution denoted by $\mathbf{p}_{\text{theory}}$ of which the Square Euclidean Imbalance is approximately $\Delta = 2^{-12.85}$. The distribution is calculated as

$$P(X = x) = \sum P(n_1, \ldots, n_4, n'_1, \ldots, n'_4 | \sum_{i=1}^{4} n_i + \sum_{i=1}^{4} n'_i = x \bmod 127),$$

where $P(n_i) = 1/7$ if $n_i \in \mathbb{E}$ and $P(n_i) = 0$ otherwise, and similar for $P(n'_i)$. As discussed in Section 3.3, we need $\mathcal{O}(\frac{1}{\Delta})$ to be able to separate the correct guess from the wrong ones. Let $\mathbf{p}_{\text{correct}}$ be the sample distribution (type) for the correct guess and let $\mathbf{p}_{\text{random}}$ be the sample distribution for an incorrect guess. We expect as the number of available samples increases that $\mathbf{p}_{\text{correct}}$ becomes closer to $\mathbf{p}_{\text{theory}}$ than $\mathbf{p}_{\text{random}}$. Vice versa, $\mathbf{p}_{\text{random}}$ should be closer to uniform distribution than $\mathbf{p}_{\text{correct}}$. Once we have enough samples (as the success probability in Equation (12) approaches 1), the correct guess can be determined with a high level of confidence.

Table 1 shows (in log) the list sizes in different steps of our algorithm compared to theoretical predictions in parentheses. As $\nu = 1$, we instantiate the algorithm with 2 batches of M samples to create t-samples (i.e., $2 \cdot M$ RSDP oracle calls). We see that the heuristic argument for $\mathcal{L}_{\mathsf{sht}}$, i.e., samples for guessing, is reasonable.

On the other hand, Table 2 shows the "score" of the correct guess versus random guesses. Recall that the SEI of the correct guess is $\Delta(\mathbf{p}_{\text{theory}}) = 2^{-12.85}$. We can clearly see that as the number of available samples increases, so does the quality of the correct guess compared to the random guess as expected. Note that, in the case $M = 45$, several wrong guesses score better than the correct one. In particular, they can be "farther" from $\mathbf{U}$ than the correct guess, but none is closer to $\mathbf{p}_{\text{theory}}$. However, when $M = 50$, or $M = 55$, we see that the correct guess is always the best candidate. For completeness, we also set up a number generator that produces sums of 8 elements in $\mathbb{E}$ and $-\mathbb{E}$ and measure the distance between the obtained distribution and $\mathbf{p}_{\text{theory}}$. We observe it matches fairly well

	$M = 45$	$M = 50$	$M = 55$
$(\mathbf{p}_{\text{correct}}, \mathbf{U})$	$2^{-10.56}$	$2^{-10.92}$	$2^{-11.41}$
$(\mathbf{p}_{\text{random}}, \mathbf{U})$	$2^{-11.03}$	$2^{-11.68}$	$2^{-12.30}$
$(\mathbf{p}_{\text{correct}}, \mathbf{p}_{\text{theory}})$	$2^{-11.00}$	$2^{-11.69}$	$2^{-12.32}$
$(\mathbf{p}_{\text{random}}, \mathbf{p}_{\text{theory}})$	$2^{-10.67}$	$2^{-11.15}$	$2^{-11.55}$

Table 2: The SEI of $\mathbf{p}_{\text{correct}}$ compared to the average $\mathbf{p}_{\text{random}}$, as well as Squared Euclidean distance to $\mathbf{p}_{\text{theory}}$.

with $(\mathbf{p}_{\text{correct}}, \mathbf{p}_{\text{theory}})$ from the experiment.

7 Conclusion and Discussion

In this work, we have presented a dual-style approach to solve the newly proposed and interesting Restricted Syndrome Decoding Problem. Our algorithm significantly outperforms combinatorial alternatives when the adversary can obtain many oracle samples and exhibits behaviours similar to those of BKW algorithms for LPN or coded-BKW for LWE. In particular, the improvement occurs well before the threshold at which an algebraic attack is a viable option to solve RSDP. Therefore, our approach is a relevant cryptanalysis tool for future RSDP-based applications where such scenarios can be applied. In addition, we implemented our algorithm on small parameter sets, verifying our heuristic arguments.

Limited amount of samples

In scenarios, e.g., CROSS, where the adversary only has a fixed number of samples, it is still possible to apply the BKW-style algorithm. In particular, in the create_sample() step, we can use higher t values and combine all available samples (instead of separating them into batches) to have a sufficient amount for testing. However, one has to take into consideration dependencies between t-error samples. This resembles the situation for the LPN case and the improvement made by Leveil and Fouque [LF06b] that did not use formal independence between created samples, but still showed performance as independent samples. Moreover, in later recombining steps of the procedure, it is inevitable that one creates duplicates of $2t$-error samples ($4t$ and so on). On the one hand, the algorithm efficiency suffers due to duplicates. On the other hand, the quality of the samples also degrades after each combination and affects the validity of guessing in sht(). Altogether, further research that addresses the above issues more rigorously must be done to apply our algorithm in such a setting.

Using other codes

We instantiate the employed codes with the (concatenated) repetition codes. Although they are adequate for our needs and the heuristics can be validated with simulations, these codes are not optimal as covering codes. Specifically, they do not achieve the best possible covering radius or density for a given code rate. Consequently, exploring the integration of more advanced covering codes represents a promising avenue for future research.

References

[AFS05] D. Augot, M. Finiasz, and N. Sendrier. "A Family of Fast Syndrome Based Cryptographic Hash Functions". In: *Progress in Cryptology - Mycrypt 2005, First International Conference on Cryptology in Malaysia, Kuala Lumpur, Malaysia, September 28-30, 2005, Proceedings*. Ed. by E. Dawson and S. Vaudenay. Vol. 3715. Lecture Notes in Computer Science. Springer, 2005, pp. 64–83.

[AG11] S. Arora and R. Ge. "New algorithms for learning in presence of errors". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2011, pp. 403–415.

[App+09a] B. Applebaum et al. "Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems". In: *Advances in Cryptology - CRYPTO 2009*. Ed. by S. Halevi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 595–618.

[Ara+21] N. Aragon et al. "BIKE". In: (2021). available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions`.

[Bal+20a] M. Baldi et al. "A New Path to Code-based Signatures via Identification Schemes with Restricted Errors". In: *CoRR* abs/2008.06403 (2020). arXiv: `2008.06403`.

[Bal+24b] M. Baldi et al. "LESS: Linear Equivalence Signature Scheme". In: *Round 2 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2024).

[Bal+24c] M. Baldi et al. "Zero Knowledge Protocols and Signatures from the Restricted Syndrome Decoding Problem". In: *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part II*. Ed. by Q. Tang and V. Teague. Vol. 14602. Lecture Notes in Computer Science. Springer, 2024, pp. 243–274.

[Ban+23] G. Banegas et al. “Wave”. In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2023).

[Ber+11] D. J. Bernstein et al. “Really Fast Syndrome-Based Hashing”. In: *Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings*. Ed. by A. Nitaj and D. Pointcheval. Vol. 6737. Lecture Notes in Computer Science. Springer, 2011, pp. 134–152.

[Ber+20] D. J. Bernstein et al. “Classic McEliece: conservative code-based cryptography”. In: *NIST submissions* (2020).

[Beu+19] W. Beullens et al. “PKP-Based Signature Scheme”. In: *Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings*. Ed. by F. Hao, S. Ruj, and S. S. Gupta. Vol. 11898. Lecture Notes in Computer Science. Springer, 2019, pp. 3–22.

[BFS15] M. Bardet, J.-C. Faugère, and B. Salvy. “On the complexity of the F5 Gröbner basis algorithm”. In: *Journal of Symbolic Computation* 70 (2015), pp. 49–70.

[Bit+23] S. Bitzer et al. “Generic Decoding of Restricted Errors”. In: *IEEE International Symposium on Information Theory, ISIT 2023, Taipei, Taiwan, June 25-30, 2023*. IEEE, 2023, pp. 246–251.

[BJV04] T. Baignères, P. Junod, and S. Vaudenay. “How Far Can We Go Beyond Linear Cryptanalysis?” In: *Advances in Cryptology - ASIACRYPT 2004*. Ed. by P. J. Lee. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 432–450.

[BKW03b] A. Blum, A. Kalai, and H. Wasserman. “Noise-tolerant learning, the parity problem, and the statistical query model”. In: *Journal of the ACM (JACM)* 50.4 (2003), pp. 506–519.

[BM24] W. Beullens and P. B. andIn contrast Morten Øygarden. “A Security Analysis of Restricted Syndrome Decoding Problems”. In: *IACR Cryptol. ePrint Arch.* (2024), p. 611.

[BØ23] P. Briaud and M. Øygarden. “A New Algebraic Approach to the Regular Syndrome Decoding Problem and Implications for PCG Constructions”. In: *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V*. Ed. by C. Hazay and M. Stam. Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 391–422.

[Car+22] K. Carrier et al. “Statistical Decoding 2.0: Reducing Decoding to LPN”. In: *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV*. Ed. by S. Agrawal and D. Lin. Vol. 13794. Lecture Notes in Computer Science. Springer, 2022, pp. 477–507.

[CCJ23] E. Carozza, G. Couteau, and A. Joux. “Short Signatures from Regular Syndrome Decoding in the Head”. In: *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V*. Ed. by C. Hazay and M. Stam. Vol. 14008. Lecture Notes in Computer Science. Springer, 2023, pp. 532–563.

[CFS01] N. T. Courtois, M. Finiasz, and N. Sendrier. “How to Achieve a McEliece-Based Digital Signature Scheme”. In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. Ed. by C. Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 157–174.

[Cui+24] H. Cui et al. “sfReSolveD: Shorter Signatures from Regular Syndrome Decoding and VOLE-in-the-Head”. In: *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part I*. Ed. by Q. Tang and V. Teague. Vol. 14601. Lecture Notes in Computer Science. Springer, 2024, pp. 229–258.

[Dum91] I. Dumer. “On minimum distance decoding of linear codes”. In: *Proc. 5th Joint Soviet-Swedish International Workshop Information Theory*. 1991, pp. 50–52.

[FS96] J. Fischer and J. Stern. “An Efficient Pseudo-Random Generator Provably as Secure as Syndrome Decoding”. In: *Advances in Cryptology - EUROCRYPT ’96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*. Ed. by U. M. Maurer. Vol. 1070. Lecture Notes in Computer Science. Springer, 1996, pp. 245–255.

[GJL14] Q. Guo, T. Johansson, and C. Löndahl. “Solving LPN Using Covering Codes”. In: *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C.,*

December 7-11, 2014. Proceedings, Part I. Ed. by P. Sarkar and T. Iwata. Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 1–20.

[GJL20] Q. Guo, T. Johansson, and C. Löndahl. "Solving LPN Using Covering Codes". In: *J. Cryptol.* 33.1 (2020), pp. 1–33.

[GJS15] Q. Guo, T. Johansson, and P. Stankovski. "Coded-BKW: Solving LWE using lattice codes". In: *Annual Cryptology Conference*. Springer. 2015, pp. 23–42.

[JKN25] T. Johansson, M. Khairallah, and V. Nguyen. *Efficient Authentication Protocols from the Restricted Syndrome Decoding Problem*. Cryptology ePrint Archive, Paper 2025/021. 2025.

[LF06b] É. Levieil and P.-A. Fouque. "An improved LPN algorithm". In: *Security and Cryptography for Networks: 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006. Proceedings 5*. Springer. 2006, pp. 348–359.

[Mel+23a] C. A. Melchor et al. "Hamming quasi-cyclic (HQC)". In: *NIST PQC* (2023).

[Mel+23b] G. A. Melchor et al. "SDitH". In: *Round 1 Additional Signatures to the NIST PostQuantum Cryptography: Digital Signature Schemes Call* (2023).

[Reg09] O. Regev. "On lattices, learning with errors, random linear codes, and cryptography". In: *J. ACM* 56.6 (Sept. 2009).

[Sha90] A. Shamir. "An Efficient Identification Scheme Based on Permuted Kernels (extended abstract)". In: *Advances in Cryptology — CRYPTO' 89 Proceedings*. Ed. by G. Brassard. New York, NY: Springer New York, 1990, pp. 606–609.

[Ste88] J. Stern. "A method for finding codewords of small weight". In: *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*. Ed. by G. D. Cohen and J. Wolfmann. Vol. 388. Lecture Notes in Computer Science. Springer, 1988, pp. 106–113.

[Ste93] J. Stern. "A New Identification Scheme Based on Syndrome Decoding". In: *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*. Ed. by D. R. Stinson. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 13–21.

[Vér96] P. Véron. "Improved identification schemes based on error-correcting codes". In: *Appl. Algebra Eng. Commun. Comput.* 8.1 (1996), pp. 57–69.

Popular Scientific Summary

Popular Scientific Summary

In today's interconnected digital world, cryptography is an essential and indispensable tool underpinning crucial aspects of secure communication, online banking, digital transactions, anonymity, data integrity, and so on. Imagine all of the above (nice) things are no longer possible because of quantum computers, which can undermine the security of our digital lives. Researchers are looking for new ways to protect information - a field called *post-quantum cryptography*. The process ensures long-term security, a smooth transition from classical cryptography, and maintains digital trust.

Among popular approaches, code-based cryptography is a reliable and well-understood candidate, signified by the selection of an algorithm called Hamming-Quasi Cyclic (HQC) as one of the selected few by the National Institute of Standards and Technology (NIST). This standardization organization produces the main cryptographic standards in the world. There are several other code-based promising candidates, for example, those that are still in the NIST Additional Signature Scheme Proposals, an ongoing standardization effort to diversify our post-quantum digital signature portfolio.

This thesis first examines the security of several code-based cryptographic constructions. We then propose novel attacking techniques that can be used to give a better security understanding for code-based cryptography. Lastly, we explore the application of code-based cryptography in *lightweight cryptography*.

Coding theory and code-based cryptography

Coding theory refers to the research area of codes and their applications, such as data transmission and compression or cryptography. In communication (over unreliable media), coding theory is crucial for error detection and correction, allowing the correct retrieval of information. For instance, data on scratched CDs and DVDs can still be recovered. The same idea can be applied to conceal messages, as unscrambling scrambled messages is usually a *very challenging* problem without crucial (trapdoor) information. This is called code-based cryptography.

Cryptanalysis

Although hard problems in coding theory can provide solid security foundations for code-based cryptosystems, certain design aspects can negatively affect security. Cryptanalysis refers to using various techniques to identify weaknesses in cryptographic constructions. This can improve our understanding and help us avoid pitfalls in the future. The weaknesses often arise when we prioritize performance, using several (mathematical) features an adversary can exploit. We analyzed several code-based cryptographic constructions and improved their security estimates. In addition, we also proposed novel generic cryptanalysis algorithms for relevant problems. Other than providing valuable alternatives in suitable scenarios, our work gives a more complete view of the security picture for novel problems that can be useful for future applications.

Lightweight cryptography

Lightweight cryptography refers to applications with limitations on hardware resources, power consumption, or minimal processing capacity and memory. From IoT devices and embedded systems to RFID tags and smart sensors, they proposed unique challenges to efficient cryptography that can protect sensitive data or ensure authentication. In particular, we are interested in applying code-based cryptography *efficiently* and *securely*, which is shown to be possible and can be an excellent alternative by our works.

Why it matters

Proactive post-quantum cryptography and cryptanalysis research is essential to develop and standardize new, stable cryptographic methods, whether or not quantum computers come to be. Our work contributes to a smooth transition to the post-quantum era, where we, hopefully, have a trustworthy cryptographic portfolio.