JEFFERSON EVANDI RICARDINI FERNANDES DE OLIVEIRA

qSCMS: POST-QUANTUM SECURITY CREDENTIAL MANAGEMENT SYSTEM FOR VEHICULAR COMMUNICATIONS

Doctoral Thesis submitted to Escola Politécnica da Universidade de São Paulo in fulfillment of the requirements for the degree of Doctor in Sciences.

São Paulo
2019

JEFFERSON EVANDI RICARDINI FERNANDES DE OLIVEIRA

qSCMS: POST-QUANTUM SECURITY CREDENTIAL MANAGEMENT SYSTEM FOR VEHICULAR COMMUNICATIONS

Doctoral Thesis submitted to Escola Politécnica da Universidade de São Paulo in fulfillment of the requirements for the degree of Doctor in Sciences.

Concentration area:

Computer Engineering

Supervisor:

Paulo S. L. M. Barreto

São Paulo
2019

Este trabalho é dedicado à minha amada esposa sem a qual nada seria possível.

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Ricardini Fernande de Oliveira, Jefferson Evandi

qSCMS: Post-Quantum Security Credential Management System for Vehicular Communications / J. E. Ricardini Fernande de Oliveira -- versão corr. -- São Paulo, 2019.

118 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Criptologia 2.Reticulados 3.Algoritmos 4.Privacidade I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

ACKNOWLEDGEMENTS

To my supervisor, Paulo S. L. M. Barreto, for all your help. Many Thanks!

I would like to thank my fellow lab co-workers for their feedback, cooperation and of course friendship.

My thanks to LARC and CAPES and LG Electronics for granted scholarship and their technical and financial support.

The UBK study was financed in part by the Brazilian CAPES (Finance Code 001) and CNPq (grant 301198/2017-9), and by LG Electronics.

The qSCMS was done under the LGE US support.

Finally, I must express my gratitude to my parents and my family for providing me with support and continuous encouragement throughout my years of study and through the process of researching and writing this dissertation. Thank you.

RESUMO

Com o aumento da demanda por Sistemas de Transporte Inteligentes (ITS – *intelligent transportation systems*), requisitos de segurança de informação e privacidade são primordiais. Isso levou a muitas propostas visando a criação de uma infraestrutura de chave pública veicular (VPKI – *Vehicular Public Key Infrastructure*) capaz de atender esses requisitos. Entre estes, o Sistema de Gerenciamento de Credenciais de segurança (SCMS – *Security Credential Management System*) é particularmente promissor. Ele provê autenticação de dados de uma maneira a preservar a privacidade e também suporta revogação de veículos que apresentem comportamento inadequado. Especificamente, um dos principais benefícios do SCMS é o chamado processo de *butterfly key expansion*, que emite lotes arbitrariamente grandes de certificados para pseudônimos a partir de única requisição. Embora este protocolo originalmente exija que o veículo forneça dois pares de chaves públicas/privadas separadas para as autoridades de registro, aqui é proposta uma abordagem aprimorada que as unifica em um único par de chaves. Também é mostrado esse ganho de desempenho não causa nenhuma deterioração em termos de segurança, flexibilidade ou escalabilidade quando comparado ao SCMS original. Além das melhorias no protocolo original baseado em curvas elípticas, aqui é apresentada uma versão pós-quântica do protocolo usando a hipótese de segurança R-LWE (*Ring Learning-with-errors*). Este novo protocolo tem o mesmo formato e características do original, mas usa assinatura e cifração baseada em R-LWE como esquemas subjacentes e operações em reticulados para o processo de emissão de chaves em vez de curvas elípticas.

ABSTRACT

With the increasing demand for intelligent transportation systems (ITS), security and privacy requirements are paramount. This demand led to many proposals aimed at creating a Vehicular Public Key Infrastructure (VPKI) able to address such prerequisites. Among them, the Security Credential Management System (SCMS) is particularly promising, providing data authentication in a privacy-preserving manner and supporting the revocation of misbehaving vehicles. Namely, one of the main benefits of SCMS is its so-called butterfly key expansion process, which issues arbitrarily large batches of pseudonym certificates through a single request. Despite SCMS's appealing design, in this document, we show that its certificate issuing process can be improved. Namely, this protocol originally requires the vehicle to provide two separate public/private key pairs to registration authorities; we now propose an improved approach that unifies them into a single key pair. We also show that such performance gains come with no negative impact in terms of security, flexibility or scalability when compared to the original SCMS. Besides the improvement on the initial Elliptic Curve based protocol, we present a post-quantum version of the protocol using Ring Learning-with-errors (R-LWE) assumption. This new protocol has the same shape and features of the original one, but using R-LWE-based signature and encryption as underlying schemes and Lattices operation for the key issuing instead of Elliptic Curves.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS AND ACRONYMS

AES	Advanced Encryption Standard
BCAM	Binary hash tree based Certificate Access Management
BKE	Butterfly Key Expansion
BLISS	Bimodal Lattice Signature Scheme
CAM	Certificate Access Manager
CDT	cumulative distribution table
CRL	Certificate Revocation List
DHAES	Diffie-Hellman Augmented Encryption Scheme
DHIES	Diffie-Hellman Integrated Encryption Scheme
DLAES	Logarithm Augmented Encryption Scheme
DLP	Discrete Logarithm Problem
DSV	Device Specific Value
ECC	Elliptic Curve Cryptography
ECIES	Elliptic Curve Integrated Encryption Scheme
ECDLP	Elliptic Curve Discrete Logarithm Problem
EdDSA	Edwards-curve Digital Signature Algorithm
GPV	Gentry-Peikert-Vaikuntanathan Cryptosystem
IBE	dentity-based encryption
IFAL	Issue First Activate Later

IoT	Internet of Things
ITS	Intelligent Transportation Systems
KA	Key Agreement
KDF	Key Derivation
KEM	Key Encapsulation Mechanism
LD-OTS	Lamport-Diffie One-Time Signature
LPR	Lyubashevsky-Peikert-Regev Cryptosystem
LWE	Learning With Errors
MAC	Message Authentication Code
MitM	Man-in-the-Middle
MPKC	Multivariate quadratic Public Key Cryptosystems
NIST	National Instituteof Standards and Technology
NTRU	N-th degree-truncated polynomial ring units
NTT	Number-Theoretic Transform
OSR	Order of Self-Revocation
OTS	One-Time Signature
PCA	Pseudonym Certificate Authority
PRF	Pseudorandom Function
PQC	Post-Quantum Cryptography
PUCA	Pseudonyms with User Controlled Anonymity
qSCMS	Post-quantum Security Credential Management System
qUBK	Post-quantum Unified Butterfly Key Expansion

PUCA	Pseudonym scheme with User-Controlled Anonymity
SCMS	Security Credential Management System
SIDH	Supersingular Isogeny Diffie-Hellman
SVP	shortest vector problem
RA	Registration Authority
RSA	Rivest-Shamir-Adleman Cryptosystem
R-LWE	Ring Learning With Errors
UBK	Unified Butterfly Key Expansion
V2I	Vehicle-to-Infrastructure communication
V2V	Vehicle-to-Vehicle communication
V2X	Vehicle-to-Everything communication
VPKI	Vehicular Public-Key Infrastructure
W-OTS	Winternitz OTS
XOF	Extendable-Output Function

LIST OF SYMBOLS

G	The generator of an elliptic curve group
	(Bold $\boldsymbol{G}$: Lattice public basis
$\mathtt{sig}$	A digital signature
$\mathtt{cert}$	A digital certificate
$\mathsf{U}, \mathbf{U}, \mathcal{U}$	Public keys
	(stylized $\mathcal{U}$: reserved for PCA)
	(Bold $\mathbf{U}$: Lattice key)
u, u	Private keys corresponding to U, $\mathbf{U}$ and $\mathcal{U}$
S, s	Public and private caterpillar signature keys
E, e	Public and private caterpillar encryption keys
$\hat{\mathsf{S}}, \hat{\mathsf{s}}$	Public and private cocoon signature keys
$\hat{\mathsf{E}}, \hat{\mathsf{e}}$	Public and private cocoon encryption keys
X, x	Public and private unified caterpillar keys
$\hat{\mathsf{X}}, \hat{\mathsf{x}}$	Public and private unified cocoon keys
b	Number of cocoon keys in certificate batch
f, f_s, f_e, g	Pseudorandom functions
	(Subscript $_{\mathcal{D}}$: Gaussian Pseudorandom)
$Enc(\mathcal{K}, s)$	Encryption of bitstring s with key $\mathcal{K}$
$Dec(\mathcal{K}, s)$	Decryption of bitstring s with key $\mathcal{K}$
$Sign(\mathcal{K}, s)$	Signature of bitstring s, using key $\mathcal{K}$

$Ver(\mathcal{K}, s)$	Verification of signature on s, using key $\mathcal{K}$
$\mathcal{H}(s)$	Hash of bitstring s
	($\mathcal{H}(s)_{\mathcal{R}_q}$ Special hash function that maps sto a ring element $\boldsymbol{c} \in \mathcal{R}_q$)
$s_1 \parallel s_2$	Concatenation of bitstrings s_1 and s_2
D_σ	*(centered) discrete Gaussian distribution* over $\mathbb{Z}$ with standard deviation σ
σ	Standard deviation of a Gaussian distribution
$\mathbb{Z}$	Set of the Integer Numbers
$\mathbb{Z}_q$	Set of integers modulo q
$\mathbb{R}$	Set of the Real Numbers

CONTENTS

1 INTRODUCTION

There is a longstanding pursuit for Intelligent Transportation Systems (ITS) (FIGUEIREDO et al., 2001). This pursuit has led the automotive industry to develop and expand the variety of computing and communication capabilities in vehicles (e.g., sensors and actuators). The same happens in the roadside' infrastructure (e.g., cameras, radars, and dynamic displays). Such development follows the trend for digital technologies embedded in physical objects, known as the Internet of Things (IoT).

In particular, the automotive industry has shown a growing interest in Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) interaction models. They are collectively called Vehicle-to-Everything (V2X) communications (HARDING et al., 2014). V2X enables several applications that improve transportation safety, efficiency, and human-machine interaction. Such applications, usually, involve metering road conditions (e.g., slope and current traffic) and vehicles' state (e.g., velocity, acceleration, and position) (PAPADIMITRATOS et al., 2009).

This new paradigm of vehicular communication brings a new set of requirements to provide security, safety, and availability without affecting privacy. Hence, to become largely deployed and achieve their maximum potential, such technologies must fit all those requirements. Namely, they are classified into three categories (SCHAUB; MA; KARGL, 2009; FÖRSTER; KARGL; LÖHR, 2014): communication, security, and privacy requirements.

- **Communication requirements:** They rely on the communication protocol and

the network topology. Besides transparency, the protocols employed for vehicular communications must provide scalability and support real-time delivery of messages. Hence, they should allow the participation of a considerable number of nodes with high mobility while keeping the latency of communications and data processing to a minimum.

- **Security requirements:** They are related to message authentication and certificate provisioning and revocation. They ensure that the only trust parties allowed to participate in the system and guarantee the legitimacy of messages. Furthermore, the authorities need being able to revoke a client's participation rights, especially in the case of misuse.

- **Privacy requirements:** They ensure the users' identities are protected and only the minimum of user information is disclosed. The system must provide anonymity so that the interactions among users or between a user and the authorities are unlinkable to a user's identity (unless the user consents to reveal it or in case of misbehaving of the user). Besides, certificate revocation must not compromise the anonymity of previous messages.

In this work, we focus on the Security and Privacy requirements that are provided by cryptographic protocols as well as by a revocation of a vehicle's certificates whenever misbehavior is detected. Vehicles are then expected to verify the authenticity of received messages and act upon them only if signed by a non-revoked peer.

However, to avoid privacy issues, the certificates employed by vehicles should not have the long lifespan typical of certificates traditionally used on the Internet. Additionally, such certificates are not directly linked to the client identity. Therefore they are called pseudonym certificates. Otherwise, the certificates themselves could be employed by eavesdroppers to track the mobility patterns of vehicles periodically broadcasting their positions. Even when the messages transmitted do not contain a vehicle's

exact location, tracking is still possible. For example, if the eavesdropper pinpoints the origin of each message and the time the message as sent.

As an alternative to pseudonym certificates, one might propose the use of privacy-preserving signature schemes, such as blind, ring, or groups signatures. Although more straightforward in terms of infrastructure (avoiding a complex PKI with additional entities), such approaches have such disadvantages when compared to a (more complex) VPKI base on Pseudonym certificates.

For instance, blind signature usually requires a protocol with multiple exchanges of messages before obtainment a valid signature (CHAUM, 1983). Such a feature is not compatible with the real-time requirements of V2X communication. Additionally, it is hard to get revocable (in case of misbehaving) anonymity and unlikability.

Group signatures, usually do not require protocol with multiple exchanges of messages before obtainment a valid signature. However, such schemes may require a trustable party called *group manager*. Such an entity owns a master key and in charge of adding group members. Additionally, the *group manager* can reveal the original signer in the event of disputes (CHAUM; HEYST, 1991).

The late feature of group signatures is a disadvantage compared to pseudonym based VPKI because it includes a single point of trust in the system. In other words, one single entity could break the anonymity of users by itself. Pseudonym-based VPKI can avoid this issue, as discussed in Chapter 3.

Besides that new paradigm with car communication, in recent years, there has been a substantial amount of research on quantum computers (The National Institute of Standards and Technology (NIST), December, 2016). Such computers would be capable of running Peter Shor's quantum algorithm, which has risen new concerns to cryptography. This algorithm is capable of factoring large integers and computing discrete logarithms in Abelian groups in polynomial time, more precisely

$O(\log^3 n)$ (SHOR, 1997). Since the conventional asymmetric cryptography is based precisely on these or related problems (e.g., RSA, ECC) (RIVEST; SHAMIR; ADLEMAN, 1978; MILLER, 1986) such concerns are justifiable.

Fortunately, there is alternative asymmetric cryptography based on different computational problems, which cannot be solved by quantum computers. Such cryptographic schemes became known as post-quantum, meaning that they would still be secure in a world where quantum computers exist. The main lines of post-quantum cryptography are:

- Lattices-based cryptography (GOLDREICH; GOLDWASSER; HALEVI, 1997)
- Code-based cryptography (MCELIECE, 1978; NIEDERREITER, 1986)
- Multivariate quadratic systems ($\mathcal{MQ}$) signatures (DING; SCHMIDT, 2005; KIPNIS; PATARIN; GOUBIN, 1999)
- Hash-based Signatures (DING; SCHMIDT, 2005; DODS; SMART; STAM, 2005)
- Supersingular isogeny-based key agreement (JAO; FEO, 2011)

It is worth noting that symmetric cryptography algorithms, like block ciphers and hash functions, are considered post-quantum as well (BARRETO et al., 2014). The existence of quantum computers would require an increase in the key, ciphertexts, and hash output sizes, although they are still considered secure.

Finally, it is hard to obtain a post-quantum signature scheme that fits the constrained devices and real-time requirements present on V2X communication (RÜCKERT, 2010). Differently with group signatures, it is possible to obtain relatively efficient post-quantum group signature (BANSARKHANI; MISOCZKI, 2018).

1.1 Motivation

If practical quantum computers are ever built, they can compromise the security of many commonly used asymmetric cryptographic algorithms. Particularly, quantum computers would completely break public-key cryptography used currently, like RSA and Elliptic Curve Cryptography (ECC). Such cryptosystems are used to implement digital signatures and key agreement. Such applications play a crucial role in ensuring the confidentiality and authenticity of communications on the Internet and other networks.

However, then, we are witnessing an expansion and advance on V2X technologies, which has their security and privacy concerns. Such concerns are addressed today with conventional cryptography, like ECC. However, given this possible threat of the quantum computer classical cryptography might be replaced by post-quantum cryptography soon. This paradigm leads to the necessity of a post-quantum solution for V2X technology. Particularly, such a solution has to fit all the V2X security and privacy requirements.

Additionally, even in the classical cryptography setting, there is space for improvement in the current protocols. Given this, we present here improvements on the original VPKI protocol before adapting it to a PQC setting.

1.2 Goals

Our goal is to propose, design, analyze post-quantum solutions for V2X technology. For that, we present intermediary goals:

1. Survey the state-of-the-art of V2X protocols in the literature. Such a survey enables us to detect the main weakness of nowadays protocols and find the most promising one among the available solutions.

2. improve to the current state-of-the-art V2X pseudonym certificate provisioning protocols. Besides getting better performance, such improvements aim to enable us to adapt the conventional cryptography based protocol to the post-quantum protocol.

3. Finally, propose a post-quantum solution that is competitive with the existing alternatives (or at least feasible in a real-world scenario concerning the adopted metrics).

The scope of this work focuses on the V2X pseudonym certificate provisioning protocol that can resist to quantum computer attacks. Given that, the usage and revocation process of the certificates are out of the scope.

1.3 Methodology

To achieve our goals, we initially do a literature review for V2X certificate provisioning solutions, which we present in the section 1.4. Subsequently, we choose the most promising and suitable ones and then look for improvements.

We have chosen Lattice-based cryptography as underlying primitive for adapting the conventional cryptography based VPKI to a post-quantum setting. The main reason for choosing lattices is their high flexibility to support a large variety of cryptographic primitives. Additionally, when defined over ideal lattices, they present high performance coupled with competitive key sizes (e.g., in comparison with other post-quantum cryptosystems).

The evaluation methodology includes both quantitative and qualitative metrics. The quantitative metrics employed are the key and signature sizes, as well as bandwidth occupation and processing times. Key and signature sizes are traditional metrics in signature schemes, and bandwidth occupation is crucial for V2X because this environment may have bandwidth constraints or limitations. Furthermore, the keys are

generated in batches in an interactive protocol among three entities, including the vehicle.

The qualitative metrics involve the scheme security considering both unforgeability and privacy aspects. We base such metrics on other security analyses available in the literature.

1.4 Related Works

Initially, we present and briefly discuss the main solutions for V2X certification provisioning in the literature. Among the present solutions, we highlight the Security Credential Management System (SCMS) (WHYTE et al., 2013; CAMP, 2016), which is one of the leading candidate designs for securing vehicular communications in the United States (CAMP, 2016). We give a more detailed discussion on the SCMS on Chapter 3.

Next, we present and briefly discuss the main related works on post-quantum cryptography. In particular, we briefly present the main PQC lines of research. Subsequently, we justify our choice of Lattice-based cryptography under the Ring Learning with Errors (R-LWE) assumption to our proposal, precisely, due to its high flexibility to support a large variety of cryptographic primitives.

1.4.1 V2X Communication

The emergence of V2X has led to the development of several proposals for addressing security and privacy issues in this scenario. One can find a comprehensive literature review (KHODAEI; PAPADIMITRATOS, 2015) and (PETIT et al., 2015).

One example is PUCA (FÖRSTER; KARGL; LÖHR, 2014), whose primary goal is to ensure the end users' privacy even toward (colluding) system entities. In PUCA, the revocation procedure assumes that each vehicle's trusted module (responsible for

managing its long-term enrollment certificate and other cryptographic keys) suspends its operation after receiving an order of self-revocation (OSR) addressed to one of its pseudonyms. As a result, no pseudonym resolution is necessary and, thus the identity of the users is preserved even in case of misbehavior. Albeit compelling, it is unclear how the system would prevent attackers from filtering out OSR messages addressed to them, thus avoiding revocation.

Also, by design, the authors prevent pseudonym certificates from being linked together. They argue traditional investigation methods should be used in case of misbehavior rather than rely on the V2X system to identify the culprit. This reasoning would be adequate if V2X itself did not introduce new threats and misbehavior capabilities, but that is not the case: after all, malicious users can abuse the system to cause collisions or facilitate robbery (e.g., by inducing other vehicles to slow down or take an alternative route, claiming an accident nearby). Therefore, it is reasonable that the system itself provides mechanisms to solve the issues it creates, which motivates the need for revocable privacy.

Another interesting example is the IFAL scheme (VERHEUL, 2016). It requires vehicles to activate their issued pseudonym certificates need before they can use them. Consequently, only honest clients periodically receive activation codes. Such an approach avoids the growth in the size of Certificate Revocation Lists (CRL). As even if a vehicle receives a large batch of pseudonym certificates valid for a long time, it still needs obtaining the corresponding activation codes. As a result, a revoked vehicle's certificates that are not yet activated do not need to be included in a CRL. Whereas the information identifying already active certificates only needs appearing in a CRL until they expire. This approach forces vehicles to contact the V2X infrastructure periodically; however, activation codes can be very small. So this process should be much less cumbersome than the periodical delivery of small batches of certificates, a possible alternative to avoid issuing certificates to revoked devices. This promising characteristic

of IFAL is, however, counterbalanced by a security issue: the certificate authority that issues pseudonym certificates, even without colluding with any other entity, can link the pseudonym certificates it issues to the corresponding device's enrollment certificates; therefore, the users' privacy depends on that authority's willingness to delete this information.

Like IFAL, the Binary Hash Tree based Certificate Access Management (BCAM) scheme (KUMAR; PETIT; WHYTE, 2017) was also designed to reduce CRL sizes through activation codes. Besides that, BCAM allows clients to recover activation codes from a small piece of information broadcast by a Certificate Access Manager (CAM), so vehicles are not required to request them explicitly. Namely, each batch of certificates issued to a given vehicle is encrypted by CAM, in such manner that the decryption key can be computed from a *device specific value* (DSV) generated by the CAM using a binary tree. By broadcasting the tree's root, all vehicles can decrypt their batches. To revoke a misbehaving vehicle, the CAM does not broadcast nodes of the tree that would allow the corresponding DSVs to be computed, thus, preventing the decryption of that vehicle's certificates.

Among the many pseudonym-based security solutions for V2X (see (PETIT et al., 2015) for a survey), one of the most prominent is the Security Credential Management System (SCMS) (WHYTE et al., 2013; CAMP, 2016). This solution is today one of the leading candidate designs for protecting vehicular communications in the United States (CAMP, 2016). In SCMS, a Registration Authority (RA) creates batches of pseudonym certificates for authorized vehicles from a single request, in the so-called butterfly key expansion process. The RA shuffles those certificates together; so the PCAcannot tell the original order of the batches; and sends them to a Pseudonym Certificate Authority (PCA). Such certificates are then individually signed and encrypted; so the RAcannot learn the content of the certificates; by the PCA before being sent back to the RA, which delivers them to the requesting vehicle.

SCMS was designed in such a manner that, unless RA and PCA collude, they are unable to link a pseudonym certificate to its owner nor learn whether two certificates belong to the same vehicle. Namely, the PCA does not identify the owner of each certificate, whereas the RA that delivers the certificate to the vehicle does not learn its inner contents. In case of abuse, however, the privacy of the misbehaving vehicle is lifted, and its certificates are revoked efficiently, which is done by placing linkage information, which is created by a Linkage Authority, in a Certificate Revocation List (CRL).

1.4.2 Post-quantum Cryptography

As previously mentioned, there is a possible threat to the current public key cryptography due to Shor's algorithm. Such an algorithm is capable of factoring large integers and compute discrete logarithms in finite fields (and in Elliptic Curves) in polynomial time and polynomial quantum space, more precisely in $O(\log^3 n)$ (SHOR, 1997). Since conventional asymmetric cryptosystems (e.g., RSA, ECC) underlie its security on these computational problems, information encrypted under such schemes may become insecure in a future scenario where quantum computers are a technological reality.

Fortunately, certain cryptosystems based on entirely different intractability assumptions are known to resist Shor's attack. These cryptosystems were named as Post-quantum cryptosystems (BERNSTEIN; BUCHMANN; DAHMEN, 2008). Currently, five families are most promising: Hash-based Signatures, Multivariate quadratic cryptosystems, Code-based encryption, Isogeny-based key exchange, and Lattice-based cryptosystems.

Hash-based digital Signatures first appeared on Lamport and Diffie's one-time signature (LD-OTS) scheme (LAMPORT, 1979). Although very efficient in the key and signature generation, the size of the signature is considerably large (BERNSTEIN;

BUCHMANN; DAHMEN, 2008). Aiming at solving this issue, the Winternitz OTS (W-OTS) scheme was proposed, which yields significantly shorter signatures at the price of degrading the processing times (MERKLE, 1979). Subsequently, Merkle extended the W-OTS to a multi-signature scheme use of a tree structure (MERKLE, 1979). The security of these signature schemes depends on the collision resistance and inversion resistance of the hash function used. Even though Merkle has extended the functionality of OTS schemes, they until now present limitations, for instance; the number of signatures is limited and the signatures are large when compared to other post-quantum schemes (BARRETO et al., 2014).

Multivariate Quadratic Public Key Cryptosystems (MPKC) bases its security on the difficulty of solving multivariate systems of *quadratic* equations over finite fields. This problem is known as Multivariate Quadratic Problem ($\mathcal{MQ}$ Problem), and it was shown to be NP-Hard by Patarin (PATARIN; GOUBIN, 1997). Although MPKC has been developed more intensively in the last two decades, they mostly support signatures while encryption/KEM are much less scrutinized for security.

Error-correcting codes have the original application of ensuring the correct transmission of data over a channel subject to noise, which is done by the addition of redundant information to the original message such that, if it reaches the destination with errors, the receiver can still decode it. In the cryptographic context, the first construction was the public key encryption scheme proposed by McEliece in 1978 (MCELIECE, 1978). In this scheme, one adds errors in a codeword and computes a syndrome relative to the parity-check matrix of the underlying code. The private key is a random irreducible error-correcting code, and the public key is a random generator matrix with a permuted version of this code. The ciphertext is then a codeword in which some errors were added in, such that only the owner of the private key can correct these errors and thus decrypt the message.

Isogeny based key exchange is the most recent among the post-quantum cryptosys-

tems (JAO; FEO, 2011). It is also known as Supersingular Isogeny Diffie-Hellman (SIDH) and is based on the hardness of constructing a smooth-degree isogeny between two supersingular elliptic curves defined over a finite field. It mostly supports the key exchange, while signatures are hard to obtain and inefficient when available (JAO; SOUKHAREV, 2014; GALBRAITH; PETIT; SILVA, 2017). Finally, Lattice-based cryptography starts with Ajtai's seminal work (AJTAI, 1996), that proved the existence of one-way functions based on the hardness of the shortest vector problem (SVP). After Ajtai's work, lattices have proven to be a versatile and flexible cryptographic primitive. Their simplicity and a large number of possible applications make lattices one of the most promising lines of research in post-quantum cryptography.

Since the quantum threat is expected for the foreseeable future (The National Institute of Standards and Technology (NIST), December, 2016), proactive countermeasures are being taken. Indeed, this threat has led to the *Call For Proposal* (CFP) for standardization and harmonization of post-quantum cryptosystems (PQC) by NIST (CHEN et al., 2016).

Among the standardization candidates, lattice-based cryptosystems have gained significant importance due to their high flexibility to support a large variety of cryptographic primitives (i.e., we can build encryption, signature and key agreement schemes from the same underlying primitives). Additionally, when defined over ideal lattices where *Ring-Learning With Errors* (R-LWE) is the underlying hard problem, they offer competitive performance in sizes (key, signature, and ciphertext) and processing times. Leveraging these benefits of R-LWE-based cryptosystems, we propose a post-quantum pseudonym-certificate issuance process for V2X communications.

The flexibility of lattices is vital in the context of SCMS' Certificate provisioning. The reason for that is SCMS requires the underlying cryptographic primitives to support both encryption and signing under key pairs that are derived from each other by the addition of some pseudorandomness. We deeply explore this requirement in the

subsequent chapters. While this property the ECC setting trivially achieves, it is not the same for most post-quantum candidate cryptosystems. Specifically, it precludes:

- Hash-based schemes, which only support signatures;
- Multivariate schemes, which mostly support signatures, while encryption/KEM are much less scrutinized for security and require entirely different algorithms and parameters;
- Code-based and Isogeny-based schemes, which mostly support encryption/KEM. While signatures are hard to obtain, inefficient when available, and require parameters and keys of an entirely distinct nature.

However, it does not preclude lattice-based cryptosystems, in particular to the schemes that rely on the R-LWE assumption (specifically where keys have the form $\mathbf{S} \leftarrow \mathbf{s} \cdot \boldsymbol{G} + \mathbf{e}$, for short private vectors $\mathbf{s}$ and $\mathbf{e}$ sampled from a suitable distribution and $\boldsymbol{G}$ is a uniformly random public vector.), since both encryption and signature schemes are known to differ in much less divergent ways than in other families of post-quantum proposals. In that case, even the notation mimics elliptic curve schemes as closely as possible.

Therefore, with a proper choice of algorithms and above all, distributions and parameters, it is possible to attain the same functionality as the ECC protocols (at least in the explicit version). We discuss the challenges of implicit schemes in the lattice-based setting in Section 5.4. Given that we choose lattices as underlying primitive for our post-quantum certificate provisioning process.

1.5 Contribution

Here in we present the first V2X pseudonym certificate provisioning protocol that can resist to quantum computer attacks. We have based our protocol on the SCMS's

pseudonym certificate provisioning protocol, known as butterfly key expansion.

We also show that even though SCMS provides an efficient and scalable security framework for vehicular communications, its design can be improved. Namely, we describe a method that can reduce approximately by half the processing and bandwidth costs of SCMS's certificate provisioning protocol. We obtained this gain when, instead of using separate butterfly keys for encryption and signature, we combine both keys in a unified key derivation process. Besides describing the solution in detail, we show that the performance improvements come with no negative impact in terms of security, flexibility, or scalability when compared to the original SCMS protocol.

We based our post-quantum butterfly key expansion on the unified key derivation process presented here. Since the unified key derivation is more simple and has a more compact definition, it is easier to adapt to a post-quantum setting than the original one. Additionally, the unified key derivation process leads to a more efficient and compact post-quantum version than then using separate butterfly keys for encryption and signature.

1.6 Outline

The remainder of this document is organized as follows. In Chapter 2, we give some preliminary concepts and notation. This chapter presents the most fundamental concepts on Elliptic Curve Cryptography and Lattice-Based Cryptography, as well as, the essential notation used for both. In Chapter 3, we present the original SCMS Vehicular Public Key Infrastructure architecture system and its butterfly key expansion process for certificate issuing. In Chapter 4, we show our Unified Butterfly Key Expansion and discuss its advantages above the original SCMS key expansion process, including key and signature sizes, bandwidth usage, and processing times. In Chapter 5, we show our Post-quantum version of the Unified Butterfly Key Expansion,

based on Lattices. We also discuss the main advantages and limitations of this post-quantum version, as well as, practical parameters; key and signature sizes, bandwidth usage, and processing times. We do our final remarks on Chapter 6.

2 PRELIMINARY CONCEPTS

The Butterfly key expansion and the Unified Butterfly Key expansion (UBK) protocols were originally built upon Elliptic Curve Cryptography (ECC). We chose Lattice-based cryptography as the underlying cryptography family for building the post-quantum version of the UBK expansion. Both concepts, ECC and Lattices, are explained in details in the subsequent sections. Finally, the strategy for assembling a post-quantum version of UBK is to systematically replace the ECC operations for Lattices underlying operations, while checking all the boulders and parameter sizes required by Lattice-based schemes.

2.1 Elliptic Curve Cryptography

Independently Neal Koblitz and Victor Miller (KOBLITZ, 1987; MILLER, 1986) have stated that logarithm discrete based cryptosystems can be defined over Elliptic curve (additive groups) instead of the conventional (multiplicative) finite field groups. According to them, such change has a positive impact on efficiency without degrading security.

Since then, many efforts have been expended on Elliptic Curve Cryptography (ECC) research, and a large number of cryptosystems have been proposed. This effort is because ECC enabled the use of more compact parameters, providing a security level similar to what conventional methods had previously provided.

We give below a formal definition of an elliptic curve:

Definition 1. *Let $\mathbb{Z}_q$ a finite field of size q. The Elliptic curve $E(\mathbb{Z}_q)$ is the set of solutions (x, y) over $\bar{\mathbb{Z}}_q$ for an equation of the form, which is known as Weierstrass equation:*

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{2.1}$$

Where $a_i \in \mathbb{Z}_q$ and the discriminant $\Delta \neq 0$ (HUSEMÖLLER, 2004).

The set of elliptic curve points together with the so-called point at the infinity denoted by ∞ under the addition operations + forms an additive Abelian group. The + operation is defined over the elliptic curve by the method of chord and tangent, where ∞ is the group identity element. (SILVERMAN, 1986).

An elliptic curve E over a finite field $\mathbb{Z}_q$ ($q \neq 2$) can also be defined as the set of solutions in $\bar{\mathbb{Z}}_q \times \bar{\mathbb{Z}}_q$ of the affine equation:

$$ax^2 + y^2 = 1 + dx^2y^2 \tag{2.2}$$

Where $a, d \in \mathbb{Z}_q$ and $\Delta = -16(4a^3 + 27b^2) \neq 0$. As in the definition 1, the curve is defined as the solutions (x, y) of the equation, where $x, y \in \mathbb{Z}_q$. This curve is known as *Twisted Edwards curve*, it was originally proposed by Bernstein et. Al. (BERNSTEIN et al., 2008) which are a generalization of the curves proposed by Edwards (EDWARDS, 2007). Twisted Edwards curves benefits from complete addition formula, which is considered to have faster group operation when compared to Weierstrass curves.

We also can define the scalar multiplication operation, denoted by kP, is the addition of the point P to itself $k - 1$ times. It is worth noting that the scalar multiplication operation encodes the security assumption for ECC protocols, basing their security on the hardness of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP).

The so called ECDLP is defined as follows: Given an elliptic curve E defined over a finite field $\mathbb{Z}_q$, a point $P \in E(\mathbb{Z}_q)$ of order n, the problem amounts to finding the

integer $k \in \mathbb{Z}$ such that $Q = k \cdot P$ (MILLER, 1986).

2.2 ECC Based Cryptographic Schemes

For the ECC version of SCMS key provisioning process (WHYTE et al., 2013) we assume standard algorithms for data encryption, hashing and digital signatures. For instance, the following algorithms are suggested: the AES block cipher (NIST, 2001) or PRESENT (BOGDANOV et al., 2007)) for symmetric encryption, as well as ECIES (IEEE, 2004) for asymmetric encryption; SHA-2 (NIST, 2015) or SHA-3 (NIST, 2015) as hash function; and ECDSA (NIST, 2013) or EdDSA (BERNSTEIN et al., 2012)) for implementing digital signatures.

We further describe EdDSA and ECIES protocols in Sections 2.2.1 and 2.2.2.

2.2.1 Edwards-curve Digital Signature Algorithm

The Edwards-curve Digital Signature Algorithm (EdDSA) (BERNSTEIN et al., 2012) is a digital signature scheme based on the well-known Schnorr signature scheme (SCHNORR, 1990; SCHNORR, 1991). It is instantiated on the so-called Twisted Edwards curves (EDWARDS, 2007). Like any other Schnorr variant signature schemes, EdDSA requires nonce, unique to each signature. However, differently, from other schemes, EdDSA computes the nonce deterministically, as the hash of the message and the private key. This strategy reduces the risk of vulnerabilities on the random number generator, (such as a weak or failure on generating random numbers) because the deterministic nonce is unlikely to be repeated for different signed messages. Although this reduces the attack surface in terms of random number generation and improves nonce misuse resistance during the signing process, high-quality random numbers are still required for key generation (FUJII; ARANHA, 2017).

When instantiated using the curve edwards25519, the EdDSA scheme is called

Ed25519. The edwards25519 is a Edwards curve with parameters $a = -1$ and $d = 121655/121666$, so it follows the equation:

$$edwards25519 : -x^2 + y^2 = 1 + \frac{121665}{121666}x^2y^2$$

The security level target with this configuration is 2^{128}. The Ed25519 has 32-byte public and private keys; and its signatures fit 64 bytes, which is significantly smaller than non-elliptic curve digital signature schemes.

We present below the algorithms EdDSAKeyGen,EdDSASign, and EdDSAVerify, which describe the EdDSA key generation, signature, and verification functions. The EdDSA is used as the underlying signature scheme for the Butterfly key expansion process and in the UBK, specifically for the explicit version of the protocol, which we discuss in Section 3. We also assume the EdDSA as the default signature scheme to be used by the vehicles to guarantee the authenticity of the messages in V2X communication applications.

Algorithm 1 EdDSAKeyGen: EdDSA Key Generation

Input: G ▷ Elliptic curve group generator
Output: A EdDSA valid key pair ▷ A, k
1: Sample a b-bit string k
2: Compute $\mathcal{H}(k) \rightarrow (h_0, h_1, \ldots, h_{2b-1})$ ▷ each h_i is a bit of the Hash output
3: encode $\mathcal{H}(k)$ as an integer of form $a = 2^{b-2} + \sum_{3 \leq i \leq b-3} 2^i h_i \in \{2^{b-2}, 2^{b-2} + 8, \ldots, b^{b-1}\}$
4: Compute $A \leftarrow a \cdot G$
5: **return** The public key is A, and the secret key is k.

Algorithm 2 EdDSASign: EdDSA Signature Generation

Input: $\mathsf{msg} \in \{0,1\}^n$, k ▷ The Private key, A ▷ The Public key
Output: A EdDSA valid signature ▷ $[R, s]$
1: Compute $\mathcal{H}(k) \rightarrow (h_0, h_1, \ldots, h_{2b-1})$ ▷ each h_i is a bit of the Hash output
2: Define r as $\mathcal{H}(h_b, \cdots, h_{2b-1}, \mathsf{msg}) \in \{0, 1, \cdots, 2^{2b-1}\}$
3: Compute $R \leftarrow r \cdot G$
4: encode $\mathcal{H}(k)$ as an integer of form $a = 2^{b-2} + \sum_{3 \leq i \leq b-3} 2^i h_i \in \{2^{b-2}, 2^{b-2} + 8, \ldots, b^{b-1}\}$
5: Compute $s \leftarrow r + \mathcal{H}(R \,\|\, A \,\|\, \mathsf{msg}) \cdot a$
6: **return** The signature is the pair (R, s)

Algorithm 3 EdDSAVerify: EdDSA Signature Verification

INPUT: $\mathsf{msg} \in \{0,1\}^n$, (R, s) ▷ The Signature, A ▷ The Public key
OUTPUT: $\{0,1\}$▷ Rejected, Accepted
1: Compute the $\mathcal{H}(R, A, \mathsf{msg})$
2: **if** $s \cdot G = R + H(R \,||\, A \,||\, \mathsf{msg}) \cdot A$ **then**
3: **return** 0 ▷ Reject Signature
4: **else**
5: **return** 1 ▷ Accept Signature

2.2.2 Elliptic Curve Integrated Encryption Scheme

The Elliptic Curve Integrated Encryption Scheme (ECIES) is a special case of the Diffie-Hellman Integrated Encryption Scheme (DHIES) when it is restricted to the Elliptic curve setting (IEEE, 2004). The DHIES first appeared in the literature named as Discrete Logarithm Augmented Encryption Scheme (DLAES) proposed by Bellare and Rogaway (1997). It was subsequently improved by Abdalla, Bellare and Rogaway (1998), being initially called Diffie-Hellman Augmented Encryption Scheme (DHAES) and later renamed as DHIES (ABDALLA; BELLARE; ROGAWAY, 2001).

It is semantically secure against adversaries capable of perpetrating chosen-plaintext and chosen-ciphertext attacks.

It is an integrated encryption scheme that uses the following functions:

- **Key Agreement (KA):** Function used by two parties for creating a shared secret.
- **Key Derivation (KDF):** Mechanism that produces a set of keys from keying material and some optional parameters.
- **Hash function:** Cryptographically secure message digest function.
- **Encryption:** Symmetric encryption algorithm (e.g. stream cipher or block cipher combined with an operation mode).
- **Message Authentication Code (MAC):** Information used to authenticate a message.

The security of the scheme is based on the computational Diffie–Hellman Problem (SMART, 2001).

We present below the algorithms ECIESKeyGen, ECIESEnc, and ECIESDec, which describe the ECIES key generation, encryption, and decryption functions.The ECIES is used as underlying public key encryption scheme for the Butterfly key expansion process and in the UBK, specifically for the PCA encryption of the vehicle pseudonym certificates, which we discuss in Section 3.

For algorithms ECIESKeyGen, ECIESEnc, and ECIESDec we also consider the Curve25519 in the Twisted Edwards Model. Given that, for ECIES we assume 32-byte public and private keys.

Algorithm 4 ECIESKeyGen: ECIES Key Generation

INPUT: G ▷ Elliptic curve group generator
OUTPUT: A EdDSA valid key pair ▷ A, a
1: Sample a Random integer a.
2: Compute $A \leftarrow a \cdot G$
3: **return** The public key is A, and the secret key is a.

Algorithm 5 ECIESEnc: ECIES Encryption

INPUT: $\mathsf{msg} \in \{0,1\}^n$, A ▷ The Public key
OUTPUT: $R \,\|\, c \,\|\, d$
1: Generate a random number $r \in [1, n-1]$
2: Compute $R \leftarrow r \cdot G$
3: Compute $S \leftarrow r \cdot A$
4: Use a KDF to derive symmetric encryption keys and MAC keys: $k_E \,\|\, k_M \leftarrow KDF(S)$
5: Encrypt the message: $c \leftarrow Enc(k_E; \mathsf{msg})$
6: Compute the tag of encrypted message and $d \leftarrow MAC(k_M, c \,\|\, S)$
7: **return** The output is $R \,\|\, c \,\|\, d$

Algorithm 6 ECIESDec: ECIES Decryption

INPUT: $\mathsf{msg} \in \{0,1\}^n$, a ▷ The Private key
OUTPUT: $R \,\|\, c \,\|\, d$
1: Derive the shared secret: $S \leftarrow a \cdot R$
2: Derive keys the same way as Alice did: $k_E \,\|\, k_M \leftarrow KDF(S)$
3: Use MAC to check the tag and outputs failed if $d \neq MAC(k_M, c \,\|\, S)$
4: uses symmetric encryption scheme to decrypt the message $m \leftarrow Dec(k_E, c)$

2.3 Lattice-Based Cryptography

Lattices have been studied since the 18th century by mathematicians such as Lagrange and Gauss. Lately, they gained attention and become a topic of intense research, thus becoming a powerful algorithmic tool for solving a wide variety of problems (BABAI, 1986; LENSTRA, 1983; LENSTRA; LENSTRA H.W.; LOVÁSZ, 1982). Therefore, the first applications of lattices were in cryptanalysis; one can find some applications of lattices in cryptanalysis in the survey (NGUYEN; STERN, 2001).

However, the interest in lattices cryptography for constructive purposes starts more recently with Ajtai's work (AJTAI, 1996), that proves the existence of one-way functions based on the hardness of the Shortest Vector Problem (SVP). Subsequent works have proved Lattice-based Cryptography versatility and flexibility as cryptographic primitive. Its simplicity and the wide number of possible applications make it one of the most promising lines of research in post-quantum cryptography. Moreover, some Lattice schemes are supported by security demonstrations that rely on the worst-case hardness of certain problems.

Usually Lattice-based cryptosystems are divided in two categories (BARRETO et al., 2014):

1. The schemes that use lattices with a trapdoor, so the one who knows this trapdoor can solve the hard problem (e.g., NTRU cryptosystem, Ajtai's construction). Whereas the trapdoor is the private key.

2. The schemes based on lattices without trapdoors, this is the case of Fiat-Shamir's construction over lattices (LYUBASHEVSKY, 2012) or other cryptosystems based on the LWE Problem.

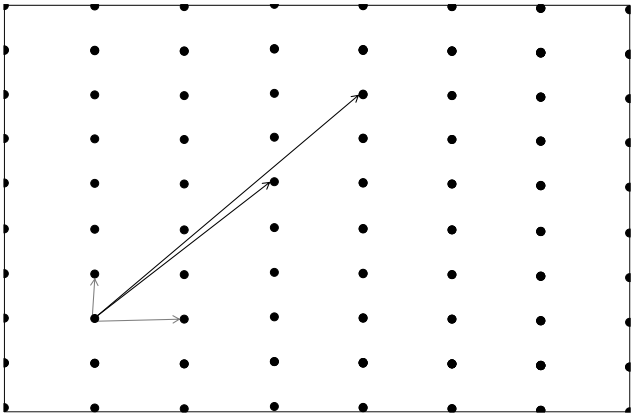
Although hard problems may not be as hard over ideal lattices as over random lattices, no polynomial algorithm is known to solve them, even when considering a polynomial

approximation factor or the utilization of quantum computation (BARRETO et al., 2014).

We give below a definition for Lattice.

A Lattice is a set of linearly independent vectors in an n-dimensional space with a periodic structure. That is, lattices are regular arrangements of points in Euclidean space, as illustrated in figure 1.

Figure 1: A 2-dimensional Lattice and two possible bases



More formally:

Definition 2. (Lattice). *Let it be an n-dimensional Euclidean space over* $\mathbb{R}$*, and let* $\mathcal{B} := \{b_1, \ldots, b_n\}$ *be a set of n linearly independent vectors. A Lattice* $\mathcal{L}$ *in that n-dimensional space is the additive subgroup consisting of all linear combinations of* $\mathcal{B}$ *with integer coefficients, with the form*

$$\mathcal{L}(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i \cdot b_i : x_i \in \mathbb{Z} \right\}. \tag{2.3}$$

$\mathcal{B}$ *is called* Lattice basis*. The same Lattice* $\mathcal{L}$ *can be generated by different bases. A basis composed by short, nearly orthogonal vectors is deemed a* good *basis, whereas a* bad *basis has a big orthogonality defect.*

Throughout this document, vector and matrix components are indexed from zero onward. Let $\mathbb{Z}$ be the set of integer numbers and $q \in \mathbb{N}$; we denote the set of integers

modulo q as $\mathbb{Z}_q$.

We can also define a lattice with matrix notation, where $\mathcal{B}$ is represented by a $B \in \mathbb{R}^{n \times m}$ matrix. The Lattice generated by B is defined as

$$\mathcal{L} = \{x \cdot B \mid x \in \mathbb{Z}^n\}$$

where x is a vector over $\mathbb{Z}$ with n elements. In general, this matrix notation (and polynomial notation of some matrix rings) is widely adopted in cryptographic context. Due to this the basic operations are limited to operations with vectors and matrices (addition and multiplications) and solving linear systems, these operations are simple enough for implementation in constrained devices.

2.3.1 Learning With Error (LWE) and Ring Learning with Error (R-LWE) Problems

Among the many Lattice-based cryptosystems available in the literature, schemes based on the Learning With Errors (LWE) Problem are the most efficient (BERNSTEIN; BUCHMANN; DAHMEN, 2008, Chapter 5.4). The LWE Problem is defined as follows: Let $\mathbb{Z}_q$ be the set of the integers $\mathbb{Z}/q\mathbb{Z}$. Let A be an uniformly distributed matrix of size $n \times m$ with elements in $\mathbb{Z}_q$, and v a vector of size n. Given the pair (A, v), one should tell whether the vector v was chosen uniformly at random or computed by an equation of the form $v = s \cdot A + e$, where s and e are integer vectors of size n sampled from their respective distributions χ_s and χ_e.

LWE can also be instantiated by defining elements as polynomial ring elements, instead of simply matrices and vectors, giving rise to the so-called Ring LWE (R-LWE) problem. Which allows formal security reductions in the random oracle model (LYUBASHEVSKY; PEIKERT; REGEV, 2010). More formally, the polynomial rings are defined as follows: Let $n = 2^k$ for an integer $k > 0$. The ring $\mathcal{R}_q := \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ is isomorphic to $\mathbb{Z}_q^n$, and the ring $\mathcal{R} := \mathbb{Z}[x]/\langle x^n + 1\rangle$ is iso-

morphic to $\mathbb{Z}^n$, where $\mathbb{Z}_q^n$ and $\mathbb{Z}^n$ represent an array of size n with elements in $\mathbb{Z}_q$ and $\mathbb{Z}$, respectively. Therefore, a ring element represented by $a_0 + \cdots + a_{n-1}x^{n-1}$ is associated with the coefficient vector $(a_0, \ldots, a_{n-1})$.

The resulting R-LWE Problem is then: given the pair $(\boldsymbol{A}, \boldsymbol{V}) \in \mathcal{R}_q \times \mathcal{R}_q$, one should tell whether both vectors are randomly generated or whether $\boldsymbol{V}$ satisfies the equation

$$\boldsymbol{s} \cdot \boldsymbol{A} + \boldsymbol{e} = \boldsymbol{V} \in \mathcal{R}_q$$

where $(\boldsymbol{s}, \boldsymbol{e}) \in \mathcal{R}_q \times \mathcal{R}_q$ are sampled from χ_s and χ_e, respectively, and $\boldsymbol{A} \in \mathcal{R}_q$ is uniformly sampled.

We give below the formal definition for R-LWE Problem.

Definition 3. *R-LWE Problem*
Let $k, q \in \mathbb{N}$ be positive integers, $n = 2^k$, and let χ be a probability distribution over $\mathbb{R}$. Let $(\boldsymbol{s}, \boldsymbol{e}) \in \mathcal{R}_q \times \mathcal{R}_q$ be sampled from distributions χ_s and χ_e respectively, and $\boldsymbol{A} \in \mathcal{R}_q$ be uniformly sampled. The ring learning with errors (R-LWE) problem *consists of distinguishing the R-LWE distribution $(\boldsymbol{A}, \boldsymbol{s} \cdot \boldsymbol{A} + \boldsymbol{e}) \in \mathcal{R}_q \times \mathcal{R}_q$ from the uniform distribution $(\boldsymbol{A}, \boldsymbol{U}) \in \mathcal{R}_q \times \mathcal{R}_q$.*

Usually, the distributions χ_s and χ_e are both defined as a discrete Gaussian distribution $\mathcal{D}_\sigma$ with standard deviation σ, discussed in Section 2.3.2.

2.3.2 Discrete Gaussian Distribution

Many modern lattice-based public-key cryptosystems, like encryption and signatures based on R-LWE require sampling from discrete Gaussian distributions (GENTRY; PEIKERT; VAIKUNTANATHAN, 2008; MICCIANCIO; PEIKERT, 2012; LYUBASHEVSKY, 2012). More precisely, most of them need a technique that allows us to make samples with a statistical difference of order 2^{-100} with a true discrete Gaussian (DWARAKANATH; GALBRAITH, 2014).

A *discrete normal distribution* on $\mathbb{Z}$ with mean μ and variance σ^2 , denoted by $\mathcal{N}_\sigma$ for a variable x, is the probability distribution defined by the probability density function

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)}.$$

The *standard normal distribution* is given by taking $\mu = 0$ and $\sigma^2 = 1$ in a general normal distribution, so the equation becomes:

$$P(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$

Moreover, an arbitrary normal distribution can be converted to a standard normal distribution by using a simple variable substitution.

In cryptography, we use a discrete version of Gaussian distribution. We define a *discrete Gaussian distribution* as:

Let $\sigma \in \mathbb{R}_{>0}$. A *discrete Gaussian distribution* denoted by $\mathcal{D}_\sigma$ is defined as follow: Let

$$S = \sum_{k=-\infty}^{\infty} e^{-k^2/(2\sigma^2)} = 1 + 2\sum_{k=1}^{\infty} e^{-k^2/(2\sigma^2)}$$

and let E be the random variable on $\mathbb{Z}$ such that, for:

$$\begin{aligned} x \in \mathbb{Z}, Pr(E = x) &= \rho_\sigma(x) \\ &= \frac{1}{S} e^{-x^2/(2\sigma^2)}. \end{aligned}$$

Some authors write the probability as proportional to $e^{-\pi x^2/s^2}$, where $s = \sqrt{2\pi}\sigma$ is a parameter (DWARAKANATH; GALBRAITH, 2014; LYUBASHEVSKY, 2012).

An essential property of Gaussian distributions is that the sum of two Normal distributions is also a Normal distribution. The mean and variance of the new distribution are the sums of the means and variances of the original distributions.

Polynomials represent lattice points, and the coefficients of secret polynomials are usually normally distributed. Since we are working with discrete distribution, the sam-

pled variables must be within a tail bound of the distribution. One can find more details about these bounds, as to how they are chosen according to the desired security level, in Dwarakanath and Galbraith's work (DWARAKANATH; GALBRAITH, 2014).

Definition 4 (Discrete Gaussian Distribution)**.** *Let $\sigma \in \mathbb{R}$ be positive. The* (centered) discrete Gaussian distribution D_σ *over $\mathbb{Z}$ with standard deviation σ is the unique distribution such that the probability of any $z \in \mathbb{Z}$ is $\rho_\sigma(z)/\rho_\sigma(\mathbb{Z})$, where*

$$\rho_\sigma(z) := e^{-\frac{z^2}{2\sigma^2}}$$

and

$$\rho_\sigma(\mathbb{Z}) := 1 + 2\sum_{z=1}^{\infty} \rho_\sigma(z).$$

The operation of sampling an integer d with discrete Gaussian distribution D_σ is denoted by $d \leftarrow D_\sigma$. Likewise, the coefficient-wise extension of this operation to vectors, i.e. the sampling of a vector $\boldsymbol{d} \in \mathbb{Z}^n$ whose components are all distributed according to D_σ is denoted by $\boldsymbol{d} \leftarrow D_\sigma^n$. For the sake of simplicity, via the identification of polynomials with their coefficient vectors, we also denote by $\boldsymbol{a} \leftarrow D_\sigma^n$ the sampling of a ring element $\mathbf{a} \in \mathbb{R}$ whose polynomial coefficients are all distributed according to D_σ. As side contribution, we present here two Gaussian sampling methods, which we detail in the two next sections (2.3.3 and 2.3.4).

2.3.3 First Gaussian Sampler

In this section, we describe an additional contribution of this work, which is a simplified Gaussian sampler based on a well-known approach due to Ducas *et al.* (DUCAS et al., 2013, Alg. 10–12). The basic idea of Ducas *et al.* is to start from a distribution that approximates the desired Gaussian much more closely than a plain uniform distribution. Namely, We start with a stepwise uniform distribution where the steps have a width ξ and height distributed according to an efficiently implementable, ξ-scaled binary Gaussian. The binary Gaussian sampler, introduced by Ducas et al. (2013), is

a discrete Gaussian with specific variance $\sigma_2 = \frac{1}{\sqrt{2 \ln 2}} \approx 0.849$. The approach focuses only on the positive half of D_{σ_2} denoted by $D^+_{\sigma_2} = \{x \leftarrow D_{\sigma_2} : x \geq 0\}$. From there, a Gaussian distribution D_σ with the desired quality is obtained by rejection sampling guided by Bernoulli distributions $\mathcal{B}_\rho$ with parameter ρ related to the standard deviation σ of the desired Gaussian distribution. That is, $\rho = \exp(-t/2\sigma^2)$ where t is an ℓ-bit integer.

Ducas *et al.* implement the Bernoulli distributions by decomposing them into ℓ base distributions $(\mathcal{B}_{\rho_0}, \mathcal{B}_{\rho_1}, \ldots, \mathcal{B}_{\rho_{\ell-1}})$. Where the ρ constants are pre-computed to the desired accuracy and then sampling from those base distributions at this same accuracy. Even though the Bernoulli decomposition is reportedly quite efficient, its running time highly depends on the private bits. Besides that, each $\mathcal{B}_{c_\rho}$ must be sampled to the same precision as the target distribution.

Consequently, the total amount of entropy needed to obtain one Gaussian sample is higher than theoretically necessary, roughly $O(\ell\lambda)$ bits rather than $O(\lambda)$ for security level λ. Additionally, this Gaussian sampler was originally designed for the signature scheme BLISS, which uses Bimodal Gaussian rather than a simple one. As a consequence, it has the additional complication that Bernoulli distributions with inverse hyperbolic cosine biases $\mathcal{B}_{1/\cosh(t/f)}$ are required as well.

In contrast, qTESLA only needs a basic Gaussian sampler for key generation and, hence, it is possible to use a much simpler construction that is easy to implement in *constant time*. In particular, only one kind of Bernoulli distribution is needed in our case, namely, a distribution $\mathcal{B}_{\exp(-t/2\sigma^2)}$ where t is an ℓ-bit integer. Thus, we can simplify the sampler by Ducas *et al.* by just computing the exponential bias $\rho = \exp(-t/2\sigma^2)$ using well-known exponentiation techniques, where ρ is an approximation of a real number in the interval $[0, 1)$ to the desired precision according to the security level λ. Specifically, we need to perform one uniform sample $y \xleftarrow{\$} \mathbb{Z}$ in $\{0, \ldots, \xi-1\}$, one binary Gaussian sample $x \in \mathbb{Z}$ according to $D^+_{\sigma_2}$, where $D^+_{\sigma_2} = \{x \leftarrow D_{\sigma_2} : x \geq 0\}$ denotes the

positive half of D_{σ_2}, and finally, one uniform sample to the desired precision based on the bias ρ with $t = y(y + 2\xi x)$. At the end of this process we have the desired Gaussian sample with $\sigma = \xi\sigma_2 = \frac{\xi}{\sqrt{2\ln 2}}$, and the total entropy consumption is $O(\lambda)$ bits.

The pseudocode of the algorithms are presented in Algorithms 7 and 8. Specifically, with our method we need to perform one uniform sample $y \xleftarrow{\$} \mathbb{Z}$ in $\{0, \ldots, \xi - 1\}$, one binary Gaussian sample $x \in \mathbb{Z}$ according to $D^+_{\sigma_2}$ and, finally, one uniform sample to the desired precision based on the bias ρ with $t = y(y + 2\xi x)$. At the end of this process we have the desired Gaussian sample with $\sigma = \xi\sigma_2 = \frac{\xi}{\sqrt{2\ln 2}}$, and the total entropy consumption is $O(\lambda)$ bits.

Algorithm 7 Sampling $D_{k\sigma 2}$ for $k \in \mathbb{Z}$

Require: An integer $k \in \mathbb{Z}$
Ensure: An integer $k \in \mathbb{Z}^+$ according $D^+_{k\sigma 2}$

1: Sample $x \in \mathbb{Z}$ according $D^+_{\sigma 2}$ [1]
2: Sample $y \in \mathbb{Z}$ uniformly in $\{0, \ldots, k-1\}$
3: $z \leftarrow kx + y$
4: Samples $b \leftarrow \mathcal{B}_{exp(-y(y+2kx)/(2\sigma^2))}$
5: **if** $\neg b$ **then**
6: restart
7: **if** $z = 0$ **then**
8: restart with probability $1/2$
9: generate a bit $b \xleftarrow{\$} \{0, 1\}$
10: **return** $(-1)^b z$

2.3.4 Second Gaussian Sampler

A well-established technique is based on the cumulative distribution table (CDT) of the normal distribution, which consists of precomputing, to a given β-bit precision, a table $\mathsf{CDT}[i] := \lfloor 2^\beta \Pr[c \leqslant i \mid c \leftarrow D^n_\sigma] \rfloor$, for $i \in [-t + 1 \ldots t - 1]$ with the smallest t such that $\Pr[|c| \geqslant t \mid c \leftarrow D^n_\sigma] < 2^{-\beta}$. To obtain a Gaussian sample, one picks a uniform sample $u \xleftarrow{\$} \mathbb{Z}/2^\beta\mathbb{Z}$, looks it up in the table, and returns the value z such that $\mathsf{CDT}[z] \leqslant u < \mathsf{CDT}[z + 1]$.

[1] In this line of the algorithm $D^+_{\sigma 2}$ is binary discrete Gaussian distribution, which is a discrete Gaussian with specific variance $\sigma_2 = \sqrt{1/(2\ln 2)} \approx 0.849$. We sample from this distribution using a CDT lookup.

Algorithm 8 Sampling $\mathcal{B}_{exp(-t/(2\sigma^2))}$ for $t \in [0, 2^w)$, BerSampler

Input: $r \in \{0,1\}^w$ and $t \in [0, 2^w)$. Set the computer wordsize w, and the precomputed Bernoulli table $\mathcal{B}$, which consists of $\mathcal{B}_{tables}$ sub-tables with $\mathcal{B}_{rows}$ rows each.

1: $c \leftarrow 2^{w-2}$
2: $s \leftarrow t$
3: **for** $i = 0, 1, \ldots, \mathcal{B}_{tables} - 1$ **do**
4: $\quad c \leftarrow c \cdot \mathcal{B}_{i, s \bmod \mathcal{B}_{rows}}$, where an element at the j-th row of the i-th sub-table is represented by $\mathcal{B}_{i,j}$
5: $\quad s \leftarrow s/\mathcal{B}_{rows}$
6: **if** $r \bmod (w-1) \geq \lfloor c \rceil$ **then**
7: $\quad$ **return** 0
8: **else**
9: $\quad$ **return** 1

A CDT-based approach has apparently first been considered for cryptographic purposes by Peikert (PEIKERT, 2010) (in a somewhat more complicated form). The approach was assessed and deemed mostly impractical by Ducas *et al.* (DUCAS et al., 2013), since it would take $\beta t \sigma$ bits. Yet, they only considered a scenario where the standard deviation σ was at least 107, and as high as 271. As a result, table sizes around 78 Kbytes are reported (presumably for $\sigma = 271$ with roughly 160-bit sampling precision). For the qTESLA parameter sets, however, the values of σ are much smaller, making the CDT approach feasible, as one can see in Table 1.

Table 1: CDT dimensions (precision in bits : size in bytes).

qTESLA-I	qTESLA-III-speed	qTESLA-III-size	qTESLA-p-I	qTESLA-p-III
96 : 3072	160 : 2980	160 : 2240	96 : 1152	160 : 2500
64 : 1672	128 : 2160	128 : 1616	64 : 632	128 : 1792

The naïve approach to CDT-based sampling is to perform table lookups via binary search, but this is susceptible to side-channel attacks since the branching depends on the private uniform samples. Two techniques can prevent side-channel attacks:

1. On platforms where a reasonably large number of Gaussian samples can be generated at once, one can sort a list of uniformly random samples together with

the CDT itself, then identify the CDT entries between which each sample is located. The cost of sorting, which can be implemented in a constant-time fashion using, e.g., Batcher's odd-even mergesort (BATCHER, 1968) (also called merge-exchange sorting (KNUTH, 1997, §5.2.2 Alg. M (Merge exchange))), is thus amortized among all samples. Constant-time sorting networks have previously been adopted in a cryptographic context to uniformly sample permutations or fixed weight vectors (BERNSTEIN et al., 2017a; BERNSTEIN et al., 2017b). Its proposed use to sample from a non-uniform distribution (specifically, the Gaussian distribution, though the same idea generalizes to any distribution) appears to be new; and

2. For memory-constrained platforms, where only one or a few samples can be generated at a time, one can adapt the sequential search always to scan the whole table, keeping track of the index z in a constant-time fashion. Interestingly, this approach may even be somewhat faster than the sorting approach when the CDT is very small (see Table 2).

Amortized sorting approach. Assume that BatcherMergeExchange(⟨*sequence*⟩, key:⟨*key*⟩, data:⟨*data*⟩) is a constant-time implementation of the Batcher merge-exchange sorting algorithm for ⟨*sequence*⟩, using the specified ⟨*key*⟩ field of each of its entries for ordering, and carrying the corresponding ⟨*data*⟩ field(s) as associated data. Algorithm 9, generates a chunk of n Gaussian samples in a constant-time fashion.

The advantages of this approach are manifold. This method can be easily written in constant-time, amortizing Batcher's merge-exchange over many samples or resorting to simple sequential search. This flexibility enables its implementation in a wide range of platforms, from desktop/server computers to embedded devices. Moreover, it supports efficient portable implementations without the need of floating-point arithmetic. Methods relying on floating-point arithmetic are more complex to implement

and, more importantly, are typically unsupported on small devices. The technique is also flexible concerning the target security level (tailored tables can be readily precomputed and conditionally compiled) since the sampling precision can be easily adjusted.

For qTESLA implementations the sampling precision is set to $\beta > \lambda/2$. Specifically, for Level-`I` and `III` parameter sets we use $\beta = 64$ and 128, respectively, for platforms with computer word-size $w = 64$.

Table 2 shows the CDT-based methods, sampling precision, and observed speedups when generating chunks of 512 Gaussian samples at a time, compared to the original qTESLA implementation submitted to NIST on November 2017 (STANDARDS; (NIST), 2017), which was based on the Bernoulli-based rejection sampling from (BARRETO et al., 2016) which in turn was based on (DUCAS et al., 2013).

Table 2: CDT speedups compared to Bernoulli-based rejection sampling.

`qTESLA-I` Batcher	`qTESLA-III-speed` Batcher	`qTESLA-III-size` Batcher	`qTESLA-p-I` sequential	`qTESLA-p-III` Batcher
64 bits	128 bits	128 bits	64 bits	128 bits
26%	42%	52%	82%	47%

Algorithm 9 generates a chunk of n Gaussian samples in a constant-time fashion, using the Batcher merge-exchange sorting algorithm (BATCHER, 1968), also called merge-exchange sorting (KNUTH, 1997, §5.2.2 Alg. M (Merge exchange)). Assume that BatcherMergeExchange(⟨*sequence*⟩, key:⟨*key*⟩, data:⟨*data*⟩) is a constant-time implementation of the Batcher merge-exchange sorting algorithm for ⟨*sequence*⟩, using the specified ⟨*key*⟩ field of each of its entries for ordering, and carrying the corresponding ⟨*data*⟩ field(s) as associated data.

Algorithm 9 Constant-time CDT-based Gaussian sampling

INPUT: n: desired number of Gaussian samples.
OUTPUT: z: a sequence of n Gaussian samples.
GLOBAL: cdt_v: the t-entry right-hand-sided, β-bit precision CDT.
LOCAL: *samp*: a list of $n + t$ triples of form (k, s, g). ▷ Denote its i-th entry fields as $samp[i].k$, $samp[i].s$, and $samp[i].g$, respectively.

▷ Prepare a sequence of n uniformly random sorting keys of β-bit precision and keep track if their original sampling order, with an initially null Gaussian index:
1: **for** $0 \leqslant i < n$ **do**
2: $\quad samp[i].k \xleftarrow{\$} \mathbb{Z}/2^{\beta}\mathbb{Z}$
3: $\quad samp[i].s \leftarrow i$
4: $\quad samp[i].g \leftarrow 0$ // placeholder
▷ Append the t entries of the CDT and keep track of the corresponding sequence of the Gaussian indices:
5: **for** $0 \leqslant i < t$ **do**
6: $\quad samp[n + i].k \leftarrow$ cdt_v$[i]$
7: $\quad samp[n + i].s \leftarrow \infty$ // search sentinel
8: $\quad samp[n + i].g \leftarrow i$
▷ Sort *samp* in constant-time according to the k field (the uniformly random samples):
9: BatcherMergeExchange(*samp*, key: k, data: s, g)
▷ Set each entry's Gaussian index, including its sign:
10: $previnx \leftarrow 0$
11: **for** $0 \leqslant i < n + t$ **do**
12: $\quad currinx \leftarrow samp[i].g$
13: $\quad previnx \leftarrow previnx \oplus (currinx \oplus previnx)\ \&\ ((previnx - currinx) \gg RADIX\text{-}1)$
14: $\quad neg \xleftarrow{\$} \{0, 1\}$ // sample the sign
15: $\quad samp[i].g \leftarrow (neg\ \&\ -previnx) \oplus (\sim neg\ \&\ previnx)$
▷ Sort *samp* in constant-time according to the s field (the sampling order):
16: BatcherMergeExchange(*samp*, key: s, data: g) // no need to involve k anymore
▷ Discard the trailing entries of *samp* (corresponding to the CDT):
17: **for** $0 \leqslant i < n$ **do**
18: $\quad z[i] \leftarrow samp[i].g$ // NB: clear *samp* afterwards to avoid memory leaks
19: **return** z

2.4 Lattice-based Cryptographic Schemes

For the post-quantum version here propose, we assume the same symmetric algorithms as the classic version, adjusting the key sizes accordingly. For example, for a 128-bit security level, block ciphers should employ 256-bit keys and the hash function's digest size should be at least 512-bit long for collision and pre-image resistance (BERNSTEIN; LANGE, 2017; BERNSTEIN, 2009). The asymmetric algorithms, on the other hand, are assumed to be: Lyubashevsky-Peikert-Regev (LPR) for asymmetric encryption (LYUBASHEVSKY; PEIKERT; REGEV, 2010); and qTESLA (AKLEYLEK et al., 2017) for digital signatures.

2.4.1 qTESLA Digital Signature Scheme

qTESLA (AKLEYLEK et al., 2017) is a signature scheme submitted to the NIST PQC standardization submission; it is the result of a long line of research. Which starts with the scheme proposed by Bai and Galbraith (BAI; GALBRAITH, 2014) that is based on the Lattice Fiat-Shamir construction (LYUBASHEVSKY, 2012). Improvements on this work were proposed in (DAĞDELEN et al., 2014), together with an optimized implementation. The scheme was afterward studied under the name TESLA (ALKIM et al., 2015) and provided with an alternative security reduction from the LWE Problem in the quantum random oracle model.

A variant of TESLA over ideal lattices was then proposed in (AKLEYLEK et al., 2016) and received the name ring-TESLA, that led to subsequent works (GUERON; SCHLIEKER, 2016). Most notably, a version of the scheme ring-TESLA called TESLA♯ (BARRETO et al., 2016) included several implementation improvements and a new Gaussian sampling algorithm.

qTESLA combines the improvements of TESLA♯ with ring-TESLA under new security reductions. It is provably EUF-CMA secure, with a tight security reduction

from the hardness of the decisional R-LWE Problem. This security reduction follows the quantum random oracle model, and it is based on a variant of original TESLA scheme over standard lattices (ALKIM et al., 2015).

One advantage that qTESLA takes from TESLA♯ is the very compact structure consisting of a few, ease-to-implement functions. Implementing qTESLA is relatively easy since the modular polynomial multiplication algorithm is the core part of the scheme. The Number theoretic transform (NTT) used in this implementation as an efficient polynomial multiplication algorithm, was also taken from the TESLA♯ scheme since it is highly optimized. Another advantage of qTESLA is that Gaussian sampling is only required during key generation, the same as for TESLA♯.

It is worth noting that qTESLA is a good candidate to be integrated into hybrid signature schemes easing the transition from classical to post-quantum cryptography. The key sizes are small enough to be used in hybrid signature schemes. Following Bindel et al. (2017) it is suitable to be used in:

- X.509 standard version 3 (COOPER et al., 2008)
- TLSv1.2 (DIERKS; RESCORLA, 2008) for most browsers and libraries tested in (BINDEL et al., 2017)
- Cryptographic Message Syntax (CMS) (HOUSLEY, 2009) that is the main cryptographic component of S/MIME (RAMSDELL; TURNER, 2010)

We summarize below the qTESLA digital signature scheme.

Let π be a permutation that sorts the components of a ring element $\boldsymbol{u}$ in decreasing order of their absolute magnitudes, *i.e.* $|\boldsymbol{u}_{\pi(0)}| \leq \cdots \leq |\boldsymbol{u}_{\pi(n-1)}|$. In what follows, $\max_i(\boldsymbol{u})$ denotes the i-th largest component of $\boldsymbol{u}$ in absolute value, that is, $\max_i(\boldsymbol{u}) = \boldsymbol{u}_{\pi(i)}$. Furthermore, for any integer c and for a given parameter d, $[c]_L$ denotes the unique integer in $(-2^{d-1}, 2^{d-1}] \cap \mathbb{Z}$ such that $c \equiv [c]_L \bmod 2^d$ (*i.e.* the centered d least

significant bits of c) and $[c]_M$ denotes the value $(c - [c]_L)/2^d$ (*i.e.* the corresponding centered most significant bits of c, with $[\boldsymbol{u}]_L$ and $[\boldsymbol{u}]_M$ denoting the application of these operations to all coefficients of $\boldsymbol{u}$. Also, qTESLA's key generation requires two subroutines checkE (Algorithm 11) and checkS (Algorithm 10). The description of both algorithm were taken from support documentation of qTESLA (AKLEYLEK et al., 2017).

Algorithm 10 checkS: simplifies qTESLA security reduction by ensuring that $\|\boldsymbol{sc}\|_\infty \leq L_S$.

INPUT: $\boldsymbol{s} \in \mathcal{R}_q$
OUTPUT: $\{0, 1\}$ ▷ true, false
1: **if** $\sum_{i=1}^{h} \max_i(\boldsymbol{s}) > L_S$ **then**
2: **return** 1
3: **return** 0

Algorithm 11 checkE: ensures correctness qTESLA by checking that $\|\boldsymbol{ec}\|_\infty \leq L_E$.

INPUT: $\boldsymbol{e} \in \mathcal{R}$
OUTPUT: $\{0, 1\}$ ▷ true, false
1: **if** $\sum_{i=1}^{h} \max_i(\boldsymbol{e}) > L_E$ **then**
2: **return** 1
3: **return** 0

Finally, $\mathcal{H}(\cdot)$ is a hash function that maps from $\{0, 1\}^*$ to a ring element $c \in \mathcal{R}_q$.

We describe qTESLA key generation, singing, and verifying below.

Key generation. First, the public polynomial $\boldsymbol{G}$ is generated uniformly at random over $\mathcal{R}_q$. Then, a secret polynomial $\boldsymbol{s}$ is sampled with Gaussian distribution $\mathcal{D}_\sigma$. This polynomial must fulfill the requirement check in checkS. A similar procedure to sample the secret error polynomial $\boldsymbol{e}$ follows. In this case, these polynomials must fulfill the correctness check (by checkE). Finally, the secret key *sk* consists of $\boldsymbol{s}$ and $\boldsymbol{e}$ and the seeds and the public key *pk* consists of the polynomial $\boldsymbol{S} \leftarrow \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e} \in \mathcal{R}_q$.

Signature generation. To sign a message msg, first generate a random polynomial $\boldsymbol{y}$ (with coefficients in $[-B, B] \cap \mathbb{Z}$). Next, compute $\boldsymbol{V} \leftarrow \boldsymbol{y} \cdot \boldsymbol{G} \in \mathcal{R}_q$. Afterward, the hash function $\mathcal{H}_{\mathcal{R}_q}$ takes as an input $[\boldsymbol{V}]_M$ and msg to generate $\boldsymbol{C}$. It is worth note that this $\mathcal{H}_{\mathcal{R}_q}$ maps the input deterministically to a pseudorandom polynomial $\boldsymbol{C}$ which the coefficients are integers in the range $\{-1, 1\}^h$.

In order for the *potential* signature $\boldsymbol{Z} \leftarrow \boldsymbol{y} + \boldsymbol{s} \cdot \boldsymbol{C} \in \mathcal{R}_q$ to be returned by the signing algorithm, it needs to pass a *security* and a *correctness* check. The goal of the security check is to avoid signature leakage of any information about the

secret $\boldsymbol{s}$. It is done by checking that $\max_0(\boldsymbol{Z}) < B - L_S$. If the check fails, the algorithm discards the current pair $(\boldsymbol{Z}, \boldsymbol{C})$ and repeats all the steps beginning with the sampling of $\boldsymbol{y}$. The correctness check has a *dual* purpose: it ensures that the signature does not leak any secret information and also ensures the correctness of the signature scheme. It is done by checking that $\max_0([\boldsymbol{W}]_L) < 2^{d-1} - L_E$ and $\max_0(\boldsymbol{W}) < \lfloor q/2 \rfloor - L_E$. If the check fails, the algorithm discards the current pair $(\boldsymbol{Z}, \boldsymbol{C})$ and repeats all the steps beginning with the sampling of $\boldsymbol{Y}$. Otherwise, the algorithm returns the signature $(\boldsymbol{Z}, \boldsymbol{C})$ on msg.

Verification. The verification algorithm, upon input of a message msg and a signature $(\boldsymbol{Z}, \boldsymbol{C})$ and computes $\boldsymbol{W} \leftarrow \boldsymbol{Z} \cdot \boldsymbol{G} - \boldsymbol{S} \cdot \boldsymbol{C} \in \mathcal{R}_q$. Finally, it uses the hash function $\mathcal{H}_{\mathcal{R}_q}$ to compute $\boldsymbol{C'}$. If the bit string resulting from the previous computation matches the signature bit string $\boldsymbol{C}$, and $\max_0(\boldsymbol{Z}) < B - L_S$, the signature is accepted; otherwise, it is rejected.

We give the pseudo-code for each function in the algorithms 12, 13 and 14 respectively. One can find more details and implementations aspects in (AKLEYLEK et al., 2017).

Algorithm 12 qTESLAKeyGen: qTESLA Key Generation

INPUT: $n, q, \sigma, h, L_S, L_E, \boldsymbol{G}$
OUTPUT: A qTESLA valid key pair ▷ $\boldsymbol{S}, \{\boldsymbol{s}, \boldsymbol{e}\}$
1: **do**
2: $\quad \boldsymbol{s} \xleftarrow{\$} \mathcal{D}_\sigma^n$
3: **while** checkS($\boldsymbol{s}$) $\neq 0$
4: **do**
5: $\quad \boldsymbol{e} \xleftarrow{\$} \mathcal{D}_\sigma^n$
6: **while** checkE($\boldsymbol{e}$) $\neq 0$
7: Compute $\boldsymbol{S} \leftarrow \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e} \in \mathcal{R}_q$
8: **return** The public key $\boldsymbol{S}$, and the secret key $\{\boldsymbol{s}, \boldsymbol{e}\}$

Algorithm 13 qTESLASign: qTESLA Signature Generation

Input: $n, q, B, L_S, d, \boldsymbol{G}, \boldsymbol{s}, \boldsymbol{e}, \boldsymbol{S}, \mathsf{msg} \in \{0,1\}^n$
Output: A qTESLA valid signature ▷ $[\boldsymbol{C}, \boldsymbol{Z}]$
1: Sample $\boldsymbol{y}$ uniformly from $[-B, B]^n \cap \mathbb{Z} \in \mathcal{R}_q$
2: $\boldsymbol{V} \leftarrow \boldsymbol{y} \cdot \boldsymbol{G} \in \mathcal{R}_q$
3: $\boldsymbol{C} \leftarrow \mathcal{H}_{\mathcal{R}_q}([\boldsymbol{V}]_M \,\|\, \mathsf{msg}) \in \mathcal{R}_q$
4: $\boldsymbol{Z} \leftarrow \boldsymbol{y} + \boldsymbol{s} \cdot \boldsymbol{C} \in \mathcal{R}_q$
5: **if** $\max_0(\boldsymbol{Z}) > B - L_S$ **then**
6: **Restart**
7: $\boldsymbol{W} \leftarrow \boldsymbol{V} - \boldsymbol{e} \cdot \boldsymbol{C} \in \mathcal{R}_q$
8: **if** $\max_0([\boldsymbol{W}]_L) > 2^{d-1} - L_E$ **or** $\max_0(\boldsymbol{W}) > \lfloor q/2 \rfloor - L_E$ **then**
9: **Restart**
10: **return** The signature $[\boldsymbol{C}, \boldsymbol{Z}]$ ▷ typically $\boldsymbol{C}$ is represented as a short raw hash value

Algorithm 14 qTESLAVerify: qTESLA Signature Verification

Input: $n, q, B, L_S, d, \boldsymbol{S}, [\boldsymbol{C}, \boldsymbol{Z}], \mathsf{msg} \in \{0,1\}^n$
Output: $\{0,1\}$▷ Rejected, Accepted
1: **if** $\max_0(\boldsymbol{Z}) > B - L_S$ **then**
2: **return** 0 ▷ Reject Signature
3: $\boldsymbol{W} \leftarrow \boldsymbol{Z} \cdot \boldsymbol{G} - \boldsymbol{S} \cdot \boldsymbol{C} \in \mathcal{R}_q$
4: $\boldsymbol{C}' \leftarrow \mathcal{H}([\boldsymbol{W}]_M \,\|\, \mathsf{msg}) \in \mathcal{R}_q$
5: **if** $\boldsymbol{C}' \neq \boldsymbol{C}$ **then**
6: **return** 0 ▷ Reject Signature
7: **else**
8: **return** 1 ▷ Accept Signature

2.4.2 Lyubashevsky-Peikert-Regev Key Encapsulation Scheme

The Lyubashevsky-Peikert-Regev (LPR) scheme (LYUBASHEVSKY; PEIKERT; REGEV, 2010) is a variant of the Gentry-Peikert-Vaikuntanathan (GPV) algorithm (GENTRY; PEIKERT; VAIKUNTANATHAN, 2008), which in turn is based on the seminal Regev cryptosystem (REGEV, 2009). The main difference is that, in the LPR cryptosystem, both, the public key and the encryption nonce, are R-LWE samples, while in GPV and Regev either the nonce or the public key are syndromes of a R-LWE sample. The LPR cryptosystem is more suitable in our context because its keys have in the same format as qTESLA and it has the same basic operations, which allows us to use the same underlying function for both schemes including the field arithmetic and polynomial operations (e.g., NTT). Additionally, LRP enables a more flexible choice

of parameters, which allows us to find secure parameters that fit qTESLA and LRP security requirements at the same time.

Although it requires Gaussian samplings in the encryption algorithm, it does not require the Gaussian sample to have high quality as in signature schemes (DWARAKANATH; GALBRAITH, 2014).

We summarize below the LPR public-key Encryption scheme.

In what follows $\bar{\boldsymbol{G}} \in \mathcal{R}_q$ is a uniformly sampled ring element, which is shared among the users in the present scenario, but elsewhere could be individually chosen as part of the public key. The encryption base $\bar{\boldsymbol{G}}$ corresponds to the signing base $-\boldsymbol{G}$.

The give the description for LPR key generation, Encryption and Decryption in the algorithms 15, 16 and 17 respectively.

Algorithm 15 LPRKeyGen: LPR Key Generation

INPUT: $n, q, \boldsymbol{G}$
OUTPUT: A LPR valid key pair ▷ $\boldsymbol{S}$, $\boldsymbol{s}$
1: $\boldsymbol{s}, \boldsymbol{e} \xleftarrow{\$} \mathcal{D}_\sigma^n$ until $\boldsymbol{s}, \boldsymbol{e} \in \mathcal{R}_q^\times$
2: Compute $\boldsymbol{S} \leftarrow \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e} \in \mathcal{R}_q^\times$
3: **return** The public key $\boldsymbol{S}$, and the secret key $\boldsymbol{s}$ ▷ The $\boldsymbol{e}$ component remains secret but is not further used.

Algorithm 16 LPREncrypt: : LPR Key Encryption

INPUT: $n, q, r, \boldsymbol{G}, \boldsymbol{S}, \mathsf{msg} \in \{0,1\}^\lambda$
OUTPUT: The Ciphertext $[\boldsymbol{C}, \boldsymbol{D}]$
1: Sample $\boldsymbol{u}, \boldsymbol{v} \xleftarrow{\$} \mathcal{D}_r^n$ and $\boldsymbol{w} \xleftarrow{\$} \mathcal{D}_r^\lambda$
2: Encode msg as $\boldsymbol{M} \leftarrow \lfloor q/2 \rfloor \cdot \mathsf{msg} \in \mathcal{R}_q$, truncated to λ out of n coefficients.
3: $\boldsymbol{C} \leftarrow \boldsymbol{u} \cdot \boldsymbol{G} + \boldsymbol{v} \in \mathcal{R}_q$
4: $\boldsymbol{D} \leftarrow (\boldsymbol{u} \cdot \boldsymbol{S})|_\lambda + \boldsymbol{w} + \boldsymbol{M} \in \mathcal{R}_q^\lambda$
5: **return** The Ciphertext $[\boldsymbol{C}, \boldsymbol{D}]$

Algorithm 17 LPRDecrypt: LPR Key Decryption

INPUT: $n, q, \boldsymbol{s}, [\boldsymbol{C}, \boldsymbol{D}]$
OUTPUT: Plaintext
1: Compute $\boldsymbol{M'} \leftarrow \boldsymbol{D} - \boldsymbol{s} \cdot \boldsymbol{C} \in \mathcal{R}_q$.
2: For all $0 \leqslant j < n$, decode $\mathsf{msg}_j \leftarrow \lfloor |\boldsymbol{M'}_j| / (q/2) \rceil$
3: **return** msg

Notice that

$$\begin{aligned} \boldsymbol{M'} &= \boldsymbol{D} - \boldsymbol{s} \cdot \boldsymbol{C} \\ &= \boldsymbol{u} \cdot \boldsymbol{S} + \boldsymbol{w} + \boldsymbol{M} - \boldsymbol{s} \cdot (\boldsymbol{u} \cdot \boldsymbol{G} + \boldsymbol{v}) \\ &= \boldsymbol{u} \cdot (\boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e}) + \boldsymbol{w} + \boldsymbol{M} - \boldsymbol{s} \cdot \boldsymbol{u} \cdot \boldsymbol{G} - \boldsymbol{s} \cdot \boldsymbol{v} \\ &= \boldsymbol{u} \cdot \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{u} \cdot \boldsymbol{e} + \boldsymbol{w} + \boldsymbol{M} - \boldsymbol{u} \cdot \boldsymbol{s} \cdot \boldsymbol{G} - \boldsymbol{s} \cdot \boldsymbol{v} \\ &= \boldsymbol{M} + (\boldsymbol{u} \cdot \boldsymbol{e} - \boldsymbol{s} \cdot \boldsymbol{v} + \boldsymbol{w}). \end{aligned}$$

Thus decryption will be correct as long as

$$\boldsymbol{u} \cdot \boldsymbol{e} - \boldsymbol{s} \cdot \boldsymbol{v} + \boldsymbol{w}$$

is within the error threshold of decoding, i.e:

$$\left|(\boldsymbol{u} \cdot \boldsymbol{e} - \boldsymbol{s} \cdot \boldsymbol{v} + \boldsymbol{w})_j\right| \leqslant \lfloor q/4 \rfloor$$

for all $0 \leqslant j < \lambda$.

If msg is less than n bits long, the $\boldsymbol{D}$ component of the ciphertext can be restricted to its first λ bits, thereby considerably reducing bandwidth occupation and slightly speeding up both encryption and decryption. This is the usual case since when LPR is used as a key encapsulation mechanism (KEM). In a KEM, the input for the encryption algorithm (msg) is an ephemeral symmetric key at the desired security level: e.g., it is only $\lambda = 128$ bits long as compared to $n = 1024$, or $\lambda = 256$ bits long as compared to $n = 2048$.

2.5 Summary

In this Chapter, we have given the preliminary concepts for Elliptic Curves Cryptography and Lattice-based Cryptography. We presented the definition of an elliptic curve and the basic operations defined by the algebraic group defined by them. We

have also shown the basics concepts on Lattice-based cryptography, as well, as the basic operations defined by this primitive.

Such concepts are essential to understand the Butterfly key expansion presented in the Chapter 3, as well as, the contributions of this work presented in Chapters 4 and 5.

3 THE SECURITY CREDENTIAL MANAGEMENT SYSTEM (SCMS)

In this Chapter, we introduce the SCMS' pseudonym certificate issuing process, which provides revocable privacy and prevents other entities from tracking devices. The SCMS is one of the most prominent pseudonym-based VPKI the US (PETIT et al., 2015). In fact, the SCMS is the leading candidate design for protecting V2X communications in the United States (CAMP, 2016).

In SCMS(WHYTE et al., 2013), vehicles are provisioned with two types of certificates:

1. **An enrollment certificate**, which has a long validity time (e.g., 6 years (CAMP, 2017)). This certificate identifies authorized vehicles in the system, enabling them to acquire pseudonym certificates.

2. **Multiple pseudonym certificates**, which have short expiration times (e.g., one week) and do not explicitly identify their owners. In such a manner that, $\varsigma \geqslant 1$ pseudonym certificates are expected to be valid at the same time.

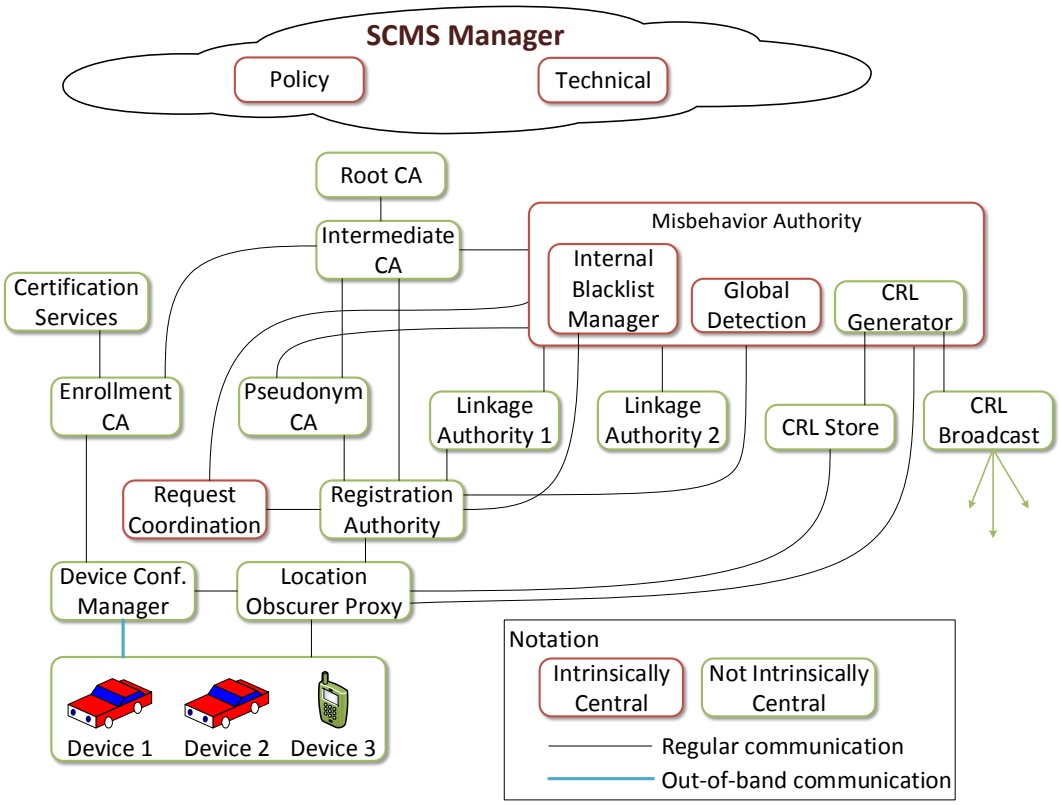
In order to protect its privacy, a specific vehicle may then frequently change the pseudonym certificate employed in its communications. As a result, the vehicle can avoid tracking by nearby vehicles or roadside units. However, it is worth note that in practice, it is useful to limit the value of ς to a small number in order to avoid "Sybil-like" attacks (DOUCEUR, 2002). In this type of attack one vehicle impersonate a

platoon aiming to get some advantage over the system (MOALLA et al., 2012). As a consequence, such a fake platoon could end up receiving preferential treatment from traffic lights programmed to give higher priority to congested roads.

The SCMS was designed to enable the distribution of multiple pseudonym certificates to vehicles in an efficient fashion. Additionally, it provides mechanisms for easily revoking the certificates in case of misbehavior by their owners. Therefore, SCMS relies on the following entities (see Figure 2 for a complete architecture and (WHYTE et al., 2013) for a detailed description of all elements):

- **Pseudonym Certificate Authority (PCA):** Responsible for issuing pseudonym certificates to the devices.
- **Registration Authority (RA):** Responsible for expanding the original keys from the devices into a batch of multiple keys. It receives and validates requests for batches of pseudonym certificates from the devices (identified by their enrollment certificates). The RA individually forwards those requests to the PCA, in such a manner that requests associated with different devices are shuffled together so the PCA cannot link a group of requests to the same device.
- **Linkage Authority (LA):** Responsible for generating pseudorandom bitstrings, which are added to the certificates so they can be efficiently revoked (namely, multiple certificates belonging to the same device can be linked together by adding a small amount of information to certificate revocation lists – CRLs). Originally, the SCMS uses two LAs, even though its architecture supports additional LAs.
- **Misbehavior Authority (MA):** Responsible for identifying misbehavior by devices and, whenever necessary, revokes them by placing their certificates into a CRL.

Figure 2: SCMS overview.



Source:(WHYTE et al., 2013).

Since the focus of this work is on batch pseudonym certificate provisioning rather than revocation, we focus on the RA, the PCA, and the devices. We further elaborate on the *butterfly key expansion* process. For further details on SCMS's revocation and certificate linkage procedure, we refer the reader to the original SCMS proposal (WHYTE et al., 2013).

For sake of ease of reading we summarize in Table 3 the list of symbols used in this Chapter. We use $s_1 \,||\, s_2$ to represent the concatenation of bit strings s_1 and s_2. We denote by $Enc(\mathcal{K}, s)$ the encryption of a bit string s with key $\mathcal{K}$, which can be done using standard block ciphers such as the Advanced Encryption Standard (AES) (NIST, 2001). Similarly, $\mathcal{H}(s)$ denotes the hash of s, using some standard hash function such as SHA-2 (NIST, 2015) or SHA-3 (NIST, 2015). The length of a given string s in

bytes is denoted $|s|$.

To facilitate the mapping between the ECC and lattice-based versions of SCMS (which we present in Section 5.2), we use the same symbols for both constructions. However, we use **boldface** fonts for the lattice-based public and private keys aiming to avoid confusion with its ECC counterpart.

Table 3: General notation and symbols

Symbol	Meaning
G	The generator of an elliptic curve group Bold $\boldsymbol{G}$: Lattice public basis
`sig`	A digital signature
`cert`	A digital certificate
$\mathsf{U}, \mathbf{U}, \mathcal{U}$	Public keys (stylized $\mathcal{U}$: reserved for PCA) (Bold $\mathbf{U}$: Lattice key)
u, u	Private keys corresponding to U and $\mathcal{U}$
S, s	Public and private caterpillar signature keys
E, e	Public and private caterpillar encryption keys
$\hat{\mathsf{S}}, \hat{\mathsf{s}}$	Public and private cocoon signature keys
$\hat{\mathsf{E}}, \hat{\mathsf{e}}$	Public and private cocoon encryption keys
X, x	Public and private unified caterpillar keys
$\hat{\mathsf{X}}, \hat{\mathsf{x}}$	Public and private unified cocoon keys
b	Number of keys generated in a batch
f, f_s, f_e, g	Pseudorandom functions (Subscript $_{\mathcal{D}}$: Gaussian Pseudorandom)
$Enc(\mathcal{K}, s)$	Encryption of bitstring s with key $\mathcal{K}$
$Dec(\mathcal{K}, s)$	Decryption of bitstring s with key $\mathcal{K}$
$Sign(\mathcal{K}, s)$	Signature of bitstring s, using key $\mathcal{K}$
$Ver(\mathcal{K}, s)$	Verification of signature on s, using key $\mathcal{K}$
$\mathcal{H}(s)$	Hash of bitstring s
$s_1 \,\|\|\, s_2$	Concatenation of bitstrings s_1 and s_2

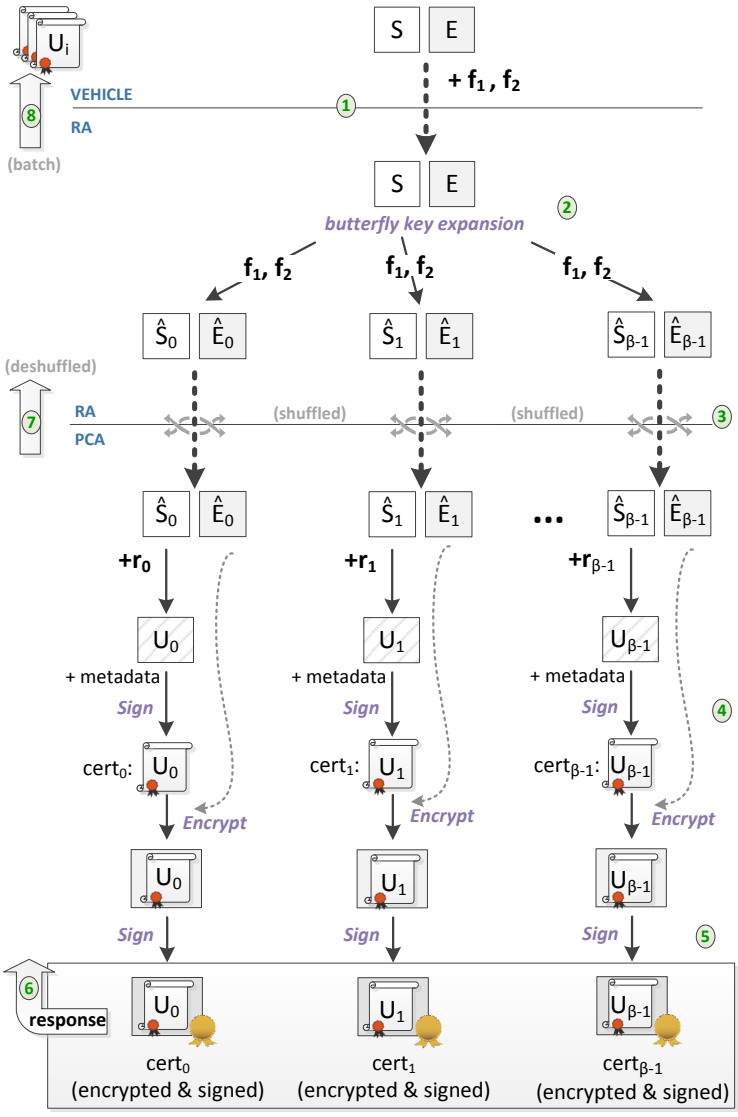
3.1 Butterfly Key Expansion

The *butterfly key* expansion process is an elegant approach of enabling a proxy server to turn a device-provided public key into multiple public keys. This process is done while allowing only the requesting device to compute the corresponding private keys. The public keys can then be placed into pseudonym certificates, signed by a

trusted authority, sent back to the proxy server, and finally forwarded to the original requester.

The *butterfly key expansion* consist of the following steps, as illustrated in Figure 3. The numbers in circles indicate the sequence of steps involved in this process and are detailed afterward.

Figure 3: SCMS's butterfly key expansion and certificate generation



Source:(SIMPLICIO et al., 2018).

(1) The vehicle generates two *caterpillar* private/public key pairs, one for signature and another for the batch encryption:

$$(\mathsf{s}, \quad \mathsf{S} = \mathsf{s} \cdot G)$$

$$(\mathsf{e}, \quad \mathsf{E} = \mathsf{e} \cdot G).$$

The public *caterpillar* keys S and E are then sent to the Registration Authority (RA) together with two suitable pseudorandom functions (PRF) f_s and f_e (or, more precisely, to their corresponding seeds).

(2) The RA uses S for generating a batch of b public *cocoon* signature keys

$$\hat{\mathsf{S}}_i = \mathsf{S} + f_s(i) \cdot G,$$

where $0 \leqslant i < b$ for an arbitrary value of b.

Similarly, the RA uses E for generating a batch of b public cocoon encryption keys

$$\hat{\mathsf{E}}_i = \mathsf{E} + f_e(i) \cdot G.$$

The exact manner by which f_s and f_e are instantiated in (WHYTE et al., 2013) differs from the description given in (CAMP, 2016), but since the differences are not pertinent to our discussion, we here omit their internal details.

(3) Pairs of cocoon keys $(\hat{\mathsf{S}}_i, \hat{\mathsf{E}}_i)$ from different vehicles are then shuffled together by the RA and sent to the Pseudonym Certificate Authority (PCA), which generates the corresponding pseudonym certificates. The RA needs to shuffle pairs of cocoon keys in such way that the PCA cannot tell which cocoon key pairs were generated from the same vehicle requisition. This shuffling process protects the vehicle's privacy from the PCA; then the shuffling process must be cryptographically secure, done by a cryptographically secure random number generator.

(4) After receiving the cocoon keys from the RA the PCA can create explicit or im-

plicit certificates (CERTICOM, 2013). In the explicit model, the PCA computes the vehicle's public signature key as

$$\mathsf{S}_i = \hat{\mathsf{S}}_i + \mathsf{r}_i \cdot G,$$

for a random value r_i. It then inserts S_i into a certificate $\texttt{cert}_i$ containing the required metadata `meta` (e.g., a validity period), and digitally signs $\texttt{cert}_i$ with its own private key u. The $\hat{\mathsf{E}}_i$ key is then used to encrypt the signed certificate together with the value of r_i, generating the PCA's response package. The PCA encrypts the signed certificate using the public key $\hat{\mathsf{E}}_i$ for which only the vehicle knows the private key ($\hat{\mathsf{e}}_i$). As a result, the RA cannot decrypt the package and learn its content, which is the final vehicle's certificate. This process protects the vehicle's privacy from the RA.

(5) The PCA's response is then digitally signed. This avoids MitM attacks from the RA.

(6) The PCA's send the response to the RA.

(7) The RA relays it (in batch) to the requesting vehicle.

(8) As a result of this process, only the corresponding vehicle can decrypt the PCA's responses to learn S_i and compute the corresponding private signature key

$$\mathsf{s}_i = \mathsf{s} + \mathsf{r}_i + f_s(i).$$

To ensure this is the correct key, the vehicle should also perform a final verification $\mathsf{s}_i \cdot G \stackrel{?}{=} \mathsf{S}_i$.

For implicitly certified keys, this process is slightly different in the Steps (4) and (8): the PCA starts by computing a credential

$$V_i = \hat{\mathsf{S}}_i + \mathsf{r}_i \cdot G$$

again for a random r_i, and then creates the implicit certificate $\mathtt{cert}_i = (V_i, \mathtt{meta})$. The PCA then signs this certificate to obtain

$$\mathtt{sig}_i = h_i \cdot \mathsf{r}_i + u,$$

where $h_i = \mathcal{H}(\mathtt{cert}_i)$, and sends back to the vehicle (via the RA) the pair $(\mathtt{cert}_i, \mathtt{sig}_i)$, encrypted with $\hat{\mathsf{E}}_i$ and signed with the PCA's private key. The vehicle, after decrypting the PCA's response and checking its signature, computes $h_i = \mathcal{H}(\mathtt{cert}_i)$ and sets its own private signature key to

$$\mathsf{s}_i = h_i \cdot (\mathsf{s} + f_s(i)) + \mathtt{sig}_i,$$

whereas the corresponding public signature key takes the form

$$\mathsf{S}_i = \mathsf{s}_i \cdot G.$$

The validity of the public key S_i can then be implicitly verified by ascertaining that

$$\mathsf{S}_i = h_i \cdot V_i + \mathcal{U},$$

where $\mathcal{U}$ is the PCA's public signature key.

We note that, for both certificate models, the PCA signs the encrypted response using its own private signature key for preventing an "honest-but-curious" RA from engaging in a Man-in-the-Middle (MitM) attack. A MitM attack happens when a malicious entity C intercepts the communication between two honest entities A and B. As a Result, the entity C gain some advantage over the entities A and B, such as reading private messages, impersonating one of the parties, exchanging original messages by fake messages. This kind of attack requires that malicious entity not be detected by the other parties, otherwise, the honest parties can suspend the connection and avoid the attack. Namely, without this signature, a MitM attack by the RA could be performed as follows:

1. Instead of $\hat{\mathsf{E}}_i$, the RA sends to the PCA a fake cocoon encryption key

$$\hat{\mathsf{E}}_i^* = z \cdot G,$$

   for an arbitrary value of z

2. The RA decrypts the PCA's response using z, learning the value of $\texttt{cert}_i$

3. The RA re-encrypts the certificate with the correct $\hat{\mathsf{E}}_i$, sending the result to the vehicle, which proceeds with the protocol as usual; and

4. Whenever a vehicle presents a pseudonym-based $\texttt{cert}_i$ to its counterparts so that they can validate its public signature key S_i, the RA can link that $\texttt{cert}_i$ to the original request, thus identifying the corresponding vehicle.

As long as the vehicle verifies the PCA's signature on the received response, however, such MitM attempt would fail because the RA would not be able to provide a valid signature for the re-encrypted certificate generated in the attack's step 3.

Also, independently of the type of certificate adopted, the users' privacy is protected in this process as long as the RA and PCA do not collude. After all, the shuffling of public cocoon keys performed by the RA prevents the PCA from learning whether or not a group of keys in the batch belongs to the same device. Unlinkability of public keys towards the RA, in turn, is obtained because the latter does not learn the value of $\texttt{cert}_i$ in the PCA's response since it is encrypted.

One can find a more detailed description in the Table 4, in this Table, we show the protocol in terms of its elliptic curves operations.

3.2 Summary

In this Chapter, we presented the original SCMS architecture, which enables the construction of a secure and privacy-preserving V2X ecosystem. To accomplish this

Table 4: Issuing pseudonym certificates: the original SCMS

	Vehicle	→	RA	→	PCA	RA→	Vehicle
SCMS (expl.)	s, $\mathsf{S}=\mathsf{s}\cdot G$	S, f_s	$\hat{\mathsf{S}}_i=\mathsf{S}+f_s(i)\cdot G$ $\hat{\mathsf{E}}_i=\mathsf{E}+f_e(i)\cdot G$ $0\leqslant i<b$	$\hat{\mathsf{S}}_i$,	$\mathsf{S}_i=\hat{\mathsf{S}}_i+\mathsf{r}_i\cdot G$ $\mathtt{sig}_i=Sign(u,\{\mathsf{S}_i,\mathtt{meta}\})$ $\mathtt{cert}_i=\{\mathsf{S}_i,\mathtt{meta},\mathtt{sig}_i\}$ $\mathtt{pkg}=Enc(\hat{\mathsf{E}}_i,\{\mathtt{cert}_i,\mathsf{r}_i\})$ $res=\{\mathtt{pkg},Sign(u,\mathtt{pkg})\}$	res	$\hat{\mathsf{e}}_i=\mathsf{e}+f_e(i)$ $Ver(\mathcal{U},res)$ $\{\mathtt{cert}_i,\mathsf{r}_i\}=Dec(\hat{\mathsf{e}}_i,\mathtt{pkg})$ $Ver(\mathcal{U},\mathtt{cert}_i)$ $\mathsf{s}_i=\mathsf{s}+f_s(i)+\mathsf{r}_i$ $\mathsf{s}_i\cdot G\stackrel{?}{=}\mathsf{S}_i$
SCMS (impl.)	e, $\mathsf{E}=\mathsf{e}\cdot G$	E, f_e		$\hat{\mathsf{E}}_i$	$V_i=\hat{\mathsf{S}}_i+\mathsf{r}_i\cdot G$ $\mathtt{cert}_i=\{V_i,\mathtt{meta}\}$ $\mathtt{sig}_i=\mathcal{H}(\mathtt{cert}_i)\cdot\mathsf{r}_i+u$ $\mathtt{pkg}=Enc(\hat{\mathsf{E}}_i,\{\mathtt{cert}_i,\mathtt{sig}_i\})$ $res=\{\mathtt{pkg},Sign(u,\mathtt{pkg})\}$		$\hat{\mathsf{e}}_i=\mathsf{e}+f_e(i)$ $Ver(\mathcal{U},res)$ $\{\mathtt{cert}_i,\mathtt{sig}_i\}=Dec(\hat{\mathsf{e}}_i,\mathtt{pkg})$ $h_i=\mathcal{H}(\mathtt{cert}_i)$ $\mathsf{s}_i=h_i\cdot(\mathsf{s}+f_s(i))+\mathtt{sig}_i$ $\mathsf{S}_i=\mathsf{s}_i\cdot G\stackrel{?}{=}h_i\cdot V_i+\mathcal{U}$

task, the SCMS provisions vehicles with an arbitrarily large batch of pseudonym certificates, using an efficient process called *Butterfly key* expansion, which as also described in this Chapter. It is worth noting that the original SCMS supports two versions of the butterfly key expansion. The first one uses traditional certificates, in which the signature butterfly key is explicitly shown in the certificate itself. In contrast, the second version relies on the implicit certificate model. When compared to the explicit certification model, the main advantage of implicit certificates is that they lead to a lower bandwidth occupation, due to the smaller size of the key reconstruction material.

Finally, the *Butterfly key* expansion process ensures that only the vehicle knows the private key that allows signatures to be generated with the received pseudonym certificates. It also creates an environment in which neither PCA or RA can link different pseudonym certificates together or identify its owner, except in the case of misbehavior. Vehicles can then protect their privacy by frequently changing the respective pseudonym certificates, thus avoiding tracking and ensuring privacy by design.

Whichever the certification model is adopted, one source of inefficiency in the original SCMS is that vehicles must send two public keys to RAs, as well as verify

two signatures generated by PCAs for each received certificate (one for the encrypted package and one for the pseudonym certificate itself). Aiming to address this issue, a simpler and more efficient version of the butterfly key expansion process is proposed in Chapter 4.

4 UNIFIED BUTTERFLY KEY EXPANSION

Albeit quite efficient, in particular from the vehicles' perspective, SCMS' pseudonym certificate provisioning protocol can be optimized. In particular, the butterfly key expansion procedure is executed twice by the RA for each pseudonym certificate: once for the generation of the public signature keys and another for encryption keys. As a result, the device itself needs to send to the RA two caterpillar keys (S and E), as well as the corresponding PRFs (f_s and f_e), for the computation of the corresponding cocoon keys ($\hat{\mathsf{S}}_i$ and $\hat{\mathsf{E}}_i$, where $0 \leqslant i < b$). Furthermore, since $\hat{\mathsf{S}}_i$ and $\hat{\mathsf{E}}_i$ are seen as independent keys by the PCA when issuing a certificate, the PCA needs not only to encrypt the certificate but also sign the resulting encrypted package to prevent the RA from manipulating $\hat{\mathsf{E}}_i$. This additional signature leads to overheads in multiple places: on the PCA, for its computation and transmission; on the RA, for its reception and re-transmission; and on the end devices, for its reception and verification, besides the verification of the certificate's signature itself.

Aiming to avoid the need of such extra signature while preserving SCMS's original certificates' format and also improving the overall efficiency of the certificate issuance procedure, we propose a unified butterfly key (UBK) expansion process. In the proposed solution, described in what follows and summarized in Table 5, the vehicle's request takes a single *unified* key instead of two separated keys. This approach leads to better efficiency without loss of security or functionality.

For the new, improved protocol the vehicle proceeds as follows: The vehicle first

generates a single caterpillar private/public key pair ($\mathsf{x}, \mathsf{X} = \mathsf{x} \cdot G$), and sends X together with a PRF f to the RA. The RA then generates b public cocoon keys

$$\hat{\mathsf{X}}_i = \mathsf{X} + f(i) \cdot G$$

for several devices, shuffles them, and sends the resulting batch to the PCA.

The subsequent procedure followed by the PCA when processing $\hat{\mathsf{X}}_i$ depends on whether explicit or implicit certificates are employed, but the steps followed are analogous to those described in Chapter 3. Namely, for implicit certificates, the PCA uses $\hat{\mathsf{X}}_i$ to generate the vehicle's credential

$$V_i = \hat{\mathsf{X}}_i + \mathsf{r}_i \cdot G,$$

builds the certificate $\mathtt{cert}_i = (V_i, \mathtt{meta})$, and computes the corresponding signature

$$\mathtt{sig}_i = h_i \cdot \mathsf{r}_i + u,$$

where $h_i = \mathcal{H}(\mathtt{cert}_i)$. In the explicit model, in turn, the device's public signature key is computed directly from $\hat{\mathsf{X}}_i$, by making

$$\mathsf{S}_i = \hat{\mathsf{X}}_i + \mathsf{r}_i \cdot G;$$

this key is then inserted into the certificate $\mathtt{cert}_i$, which is signed by the PCA using u.

Whether implicit or explicit certificates are adopted, $\hat{\mathsf{X}}_i$ is employed by the PCA as the encryption key for its response. In other words, the PCA uses $\hat{\mathsf{X}}_i$ to encrypt $\mathtt{cert}_i$ and also any additional data that it needs to provide to the vehicle (e.g., r_i for explicit certificates, and $\mathtt{sig}_i$ for implicit ones). This encryption process must be done using a suitable encryption algorithm, such as ECIES (IEEE, 2004) (as recommended in (CAMP, 2016)). This encrypted package is then sent to the RA, which relays it to the vehicle.

The vehicle can then compute the corresponding decryption key

$$\hat{\mathsf{x}}_i = \mathsf{x} + f(i)$$

and retrieve $\texttt{cert}_i$. This decryption key works because the encryption is performed using

$$\hat{\mathsf{X}}_i = \mathsf{x} \cdot G + f(i) \cdot G$$

as the public key, whereas the corresponding private key is known only to the vehicle. For implicit certificates, S_i is then computed and verified, as usual, using $\texttt{cert}_i$ and the PCA's public signature key to check that

$$\mathsf{S}_i = \mathsf{s}_i \cdot G$$

satisfies

$$\mathsf{S}_i = h_i \cdot V_i + \mathcal{U},$$

where

$$\mathsf{s} = \mathcal{H}(\texttt{cert}_i) \cdot (\mathsf{x} + f(i)) + \texttt{sig}_i.$$

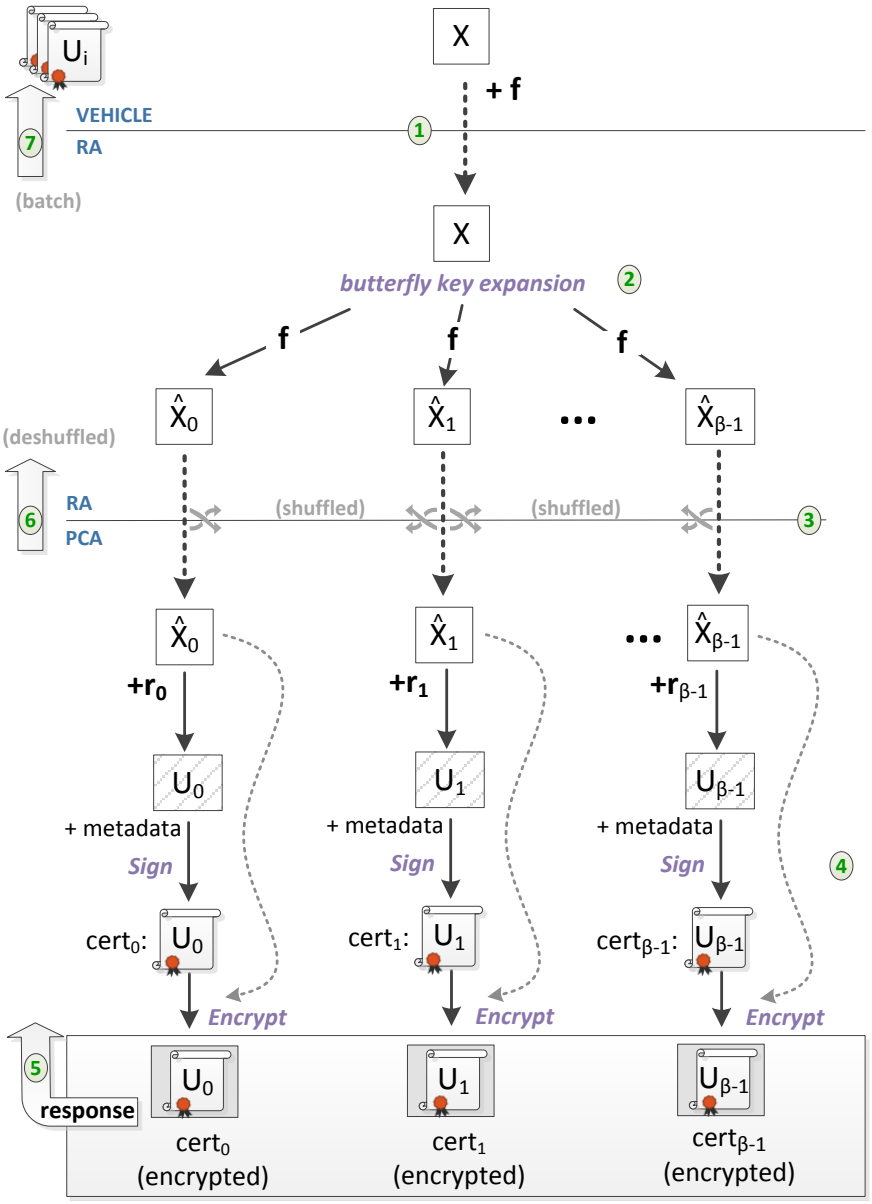
For explicit certificates, the vehicle:

1. Verifies the PCA's signature on $\texttt{cert}_i$, which encloses S_i
2. Computes $\mathsf{s}_i = \mathsf{r}_i + \hat{\mathsf{x}}_i$ using the value of r_i received in the encrypted package, ascertaining that $\mathsf{s}_i \cdot G = \mathsf{S}_i$

Whereas this final verification on the received S_i was only advisable in SCMS, for verifying the correctness of the PCA's response, it is mandatory in the proposed approach for avoiding MitM attacks by the RA (as further discussed in Section 4.1.3).

One can find a more detailed description in the Table 5, in this Table, we show our protocol in terms of its elliptic curves operations.

Figure 4: A more efficient, unified butterfly key (UBK) expansion. Numbers in circles indicate the sequence of steps involved in the process.



Source:(SIMPLICIO et al., 2018).

Table 5: UBK in the implicit and explicit models.

	Vehicle	→	RA	→	PCA	RA→	Vehicle
UBK (expl.)	x, $\mathsf{X}=\mathsf{x}\cdot G$	X, f	$\hat{\mathsf{X}}_i=\mathsf{X}+f(i)\cdot G$	$\hat{\mathsf{X}}_i$	$\mathsf{U}_i=\hat{\mathsf{X}}_i+\mathsf{r}_i\cdot G$ $\mathtt{sig}_i=Sign(u,\{\mathsf{U}_i,\mathtt{meta}\})$ $\mathtt{cert}_i=\{\mathsf{U}_i,\mathtt{meta},\mathtt{sig}_i\}$ $\mathtt{pkg}=Enc(\hat{\mathsf{X}}_i,\{\mathtt{cert}_i,\mathsf{r}_i\})$	$\mathtt{pkg}$	$\hat{\mathsf{x}}_i=\mathsf{x}+f(i)$ $\{\mathtt{cert}_i,\mathsf{r}_i\}=Dec(\hat{\mathsf{x}}_i,\mathtt{pkg})$ $Ver(\mathcal{U},\mathtt{cert}_i)$ $\mathsf{u}_i=\hat{\mathsf{x}}_i+\mathsf{r}_i$ $\mathsf{u}_i\cdot G\stackrel{?}{=}\mathsf{U}_i$
UBK (impl.)			$0\leqslant i<b$		$V_i=\hat{\mathsf{X}}_i+\mathsf{r}_i\cdot G$ $\mathtt{cert}_i=\{V_i,\mathtt{meta}\}$ $\mathtt{sig}_i=\mathcal{H}(\mathtt{cert}_i)\cdot\mathsf{r}_i+u$ $\mathtt{pkg}=Enc(\hat{\mathsf{X}}_i,\{\mathtt{cert}_i,\mathtt{sig}_i\})$		$\hat{\mathsf{x}}_i=\mathsf{x}+f(i)$ $\{\mathtt{cert}_i,\mathtt{sig}_i\}=Dec(\hat{\mathsf{x}}_i,\mathtt{pkg})$ $h_i=\mathcal{H}(\mathtt{cert}_i)$ $\mathsf{u}_i=h_i\cdot(\mathsf{x}+f(i))+\mathtt{sig}_i$ $\mathsf{U}_i=\mathsf{s}_i\cdot G\stackrel{?}{=}h_i\cdot V_i+\mathcal{U}$

4.1 Security Discussion

Even though SCMS does not provide formal security proofs, we present here security arguments focused on the following underlying security goals:

- The confidentiality of the vehicle's private key s; the confidentiality of the PCA's response (or, more precisely, of the pseudonym certificate thereby enclosed), in particular toward the RA
- The integrity of the pseudonym certificates in the PCA's response
- The unlinkability of the pseudonym certificates, as long as PCA and RA do not collude.

The overall security of the proposed scheme builds upon the same principles as the original SCMS butterfly key expansion. Namely, there is no modification on the process that defines how the PCA and RA handle the caterpillar and cocoon signature keys. Therefore, the security arguments of SCMS regarding the confidentiality of each u_i, which rely on the fact that x remains protected by the elliptic curve discrete logarithm problem (ECDLP, given in Definition 5) during the whole execution of the protocol, remain valid. Hence, neither RA nor PCA can recover the signature or de-

cryption private keys derived from it, even if they collude. Because, only the vehicle and knows the original x which is protected by the ECDLP in the equation $\mathsf{X} = \mathsf{x} \cdot G$. As a consequence, the RA only knows the public value X, and the PCAonly know the public value $\hat{\mathsf{X}}$. So if even them collude, they have only access to public keys protected by the ECDLP and cannot derive signature or decryption private keys from it.

Additionally, certificate unlinkability is also preserved as long as RA and PCA do not collude: the shuffling done by the RA hides from the PCA any relationship between certificate requests for the same vehicle; meanwhile, the PCA's encrypted response prevents anyone but the owner of the decryption key from learning the content of $\texttt{cert}_i$.

Definition 5. *Elliptic Curve Discrete Logarithm Problem (ECDLP) (LAUTER; STANGE, 2008). Let E be an elliptic curve over a finite field $\mathbb{Z}_q$. Suppose there are points $P, Q \in E(\mathbb{Z}_q)$ given such that $Q \in \langle P \rangle$. Determine k such that $Q = k \cdot P$.*

On the other hand, the unified key approach introduces two changes to SCMS:

1. It modifies how the encryption key is computed
2. It eliminates the PCA's signature on the encrypted package

The first modification could affect the confidentiality of the communication, thus allowing the RA to learn $\texttt{cert}_i$. Meanwhile, since the final signature made by the PCA on its response is aimed at ensuring the system's security against MitM attacks by the RA, the second modification could result in vulnerabilities on that aspect, affecting the confidentiality and/or integrity of issued certificates. However, in what follows, we show that the unified key approach still protects the pseudonym certificates' contents and prevents MitM attacks, assuming the hardness of the ECDLP. More precisely, we show that the problem of decrypting the PCA's response encrypted with X is reduced to an instance of ECDLP. The same computational hardness applies to MitM attacks,

for which we show that the PCA's response is such that any manipulation by the RA is detectable by the device when validating the public key S_i, either explicitly or implicitly.

Ideally, a PKIs must achieve security with respect to the strongest possible adversary model, the same assumption is valid for VPKIs. Since PKIs (and VPKIs) run over a probably vulnerable network, in the worst case the adversary can control the network. We, therefore, assume that the adversary can eavesdrop, modify, and insert messages. For SCMSthis is the case of the RA. The RA is the system entity that receives the vehicle's request and relays it to the PCA(after shuffling them). As a result, it has access to every request from every vehicle and to all the PCA encrypted responses before forwarding them to the vehicles. Given that, the RA could act as a natural MitM for the system and it is the strongest adversary in the system.

Therefore, we assume that an adversary that can compromise the RA is the strongest adversary in the system. Our following security arguments cover mainly attacks made by a rogue (compromised) RA. Such adversary can eavesdrop, modify, and insert requests (from vehicles) and encrypted responses (from the PCA).

Additionally, we consider that the system's entities may be "dishonest if allowed": they may engage in active attacks (without colluding), subverting the protocols if this would bring them some advantage (e.g., the ability to track vehicles, recover vehicles private key); however, we assume such misbehavior occurs only if it can go undetected.

4.1.1 Confidentiality of Pseudonym Certificates

In the SCMS, the goal of encrypting the response package with the public encryption key $\hat{E}$ is to prevent RA from learning its contents. This feature is accomplished simply by using $\hat{e}$ for which the corresponding private key $\hat{e}$ remains unknown by the RA. The unified key strategy, here proposed, is built upon the observation that both the encryption $\hat{e}$ and signature $\hat{s}$ private keys need to remain protected in SCMS. We

observe that this feature can still be achieved if we combine the keys into a single piece of information. We discuss the security of using $\hat{\mathsf{X}}_i$ directly as an encryption key below.

Suppose that the RA follows an honest-but-curious security model, sending the correct

$$\hat{\mathsf{X}} = \mathsf{X} + f(i) \cdot G$$

to the PCA. In that case, the RA is unable to recover the contents of the PCA's encrypted response pkg in polynomial time unless it can solve an instance of the ECDLP in polynomial time.

We argue that, if the encryption is performed with a secure algorithm, there should be no polynomial-time algorithm that allows decryption without knowledge of $\hat{\mathsf{x}}$, nor a polynomial-time algorithm that allows the recovery of this key from pkg. Hence, violating the confidentiality of the scheme requires the recovery of $\hat{\mathsf{x}}$ from either X or $\hat{\mathsf{X}}$. After all, besides pkg itself, these are the only pieces of information possessed by the RA that carry some relationship with the decryption key $\hat{\mathsf{x}}$. However, since

$$\hat{\mathsf{X}} = \mathsf{X} + f(i) \cdot G,$$

where $f(i)$ is known by the RA, this task is equivalent to finding x from X, i.e., to solving the ECDLP for X.

4.1.2 Security Against MitM Attacks by RAs (Implicit Model)

The security result obtained in Section 4.1.1 assumes that the RA follows the UBK protocol, providing the correct $\hat{\mathsf{X}}$ to the PCA. However, the RA might prefer to replace this key with

$$\hat{\mathsf{X}}_i^* = z \cdot G,$$

for an arbitrary value of z. In this case, the RA would violate the client's confidentiality in the process, because the PCA would end up encrypting pkg with $\hat{\mathsf{X}}_i^*$ and the

result would be trivially decrypted by the RA using the corresponding private key z. Therefore, we need also to consider the security of this Man-in-the-Middle scenario, which is complementary to the "honest-but-curious" scenario previously assumed. We impose no constraint on the (mis)behavior of the RA, letting it freely choose $\hat{X}_i^*$ as long as:

1. The choice leads to some advantage to the RA, in particular, the ability to violate the confidentiality or integrity of `pkg`;
2. The vehicles cannot detect the RA's misbehavior, so they believe it is safe to use the corresponding certificates.

From this scenario, we formulate the following argument: Suppose that the RA replaces $\hat{X}_i$ by an arbitrary $\hat{X}_i^*$ in the request for implicit certificates sent to the PCA. Assuming the hardness of the ECDLP and the random oracle model, the RA cannot violate the integrity or confidentiality of the PCA's response `pkg` without the requesting vehicle's knowledge.

We first notice that the `pkg`'s content integrity is protected despite the lack of the PCA signature over it. Indeed, even if the RA is somehow able to violate the confidentiality of `pkg`, it would only be able to obtain the (signed) implicit certificate $\texttt{cert}_i$. However, $\texttt{cert}_i$ is not treated as confidential in the implicit certification model (CERTICOM, 2013, Section 3.4), and yet such model ensures the integrity of $\texttt{cert}_i$ in the random oracle model assuming the hardness of the ECDLP (BROWN; GALLANT; VANSTONE, 2002). Therefore, the implicit certification itself already ensures that any modification of $\texttt{cert}_i$, either directly (i.e., after decrypting `pkg`) or indirectly (i.e., by modifying only the ciphertext), would be detectable by vehicles.

Proving the confidentiality of the unified key expansion, however, requires some more effort because we cannot rely so directly on the security properties of implicit certificates. Once again, we follow the reductionist approach, showing that violating

the confidentiality of $\texttt{pkg}$ requires the resolution of an instance of the ECDLP.

Suppose that the malicious RA replaces the correct value of $\hat{\mathsf{X}}_i$ by

$$\hat{\mathsf{X}}_i^* = z \cdot G,$$

for an arbitrary value of z. This assumption comes without loss of generality since, in principle, we do not impose any restriction on the actual value of z chosen by the RA. Upon reception of the RA's request, the PCA ends up encrypting the implicit certificate $\texttt{cert}_i$ with $\hat{\mathsf{X}}_i^*$, since it is unable to detect such misbehavior. As a result, the RA can decrypt the PCA's response using z as the decryption key, thus violating the confidentiality of the system. This attack would allow the RA to learn the vehicle's implicit certificate $\texttt{cert}_i^* = (V_i^*, \texttt{meta})$, where

$$V_i^* = \hat{\mathsf{X}}_i^* + \mathsf{r}_i \cdot G,$$

as well as its corresponding signature

$$\texttt{sig}_i^* = \mathcal{H}(\texttt{cert}_i^*) \cdot \mathsf{r}_i + u,$$

where $h_i^* = \mathcal{H}(\texttt{cert}_i^*)$.

However, this misbehavior by the RA can be detected by the vehicle because for any

$$z \neq \mathsf{x} + f(i),$$

the resulting $\texttt{sig}_i^*$ would not be a valid signature for the actual $\hat{\mathsf{X}}_i$ expected by the vehicle. More precisely, after the vehicle computes

$$\mathsf{S}_i = \mathsf{s}_i \cdot G$$

for

$$\mathsf{s}_i = h_i^* \cdot (\mathsf{x} + f(i)) + \texttt{sig}_i^*,$$

the implicit verification

$$\mathsf{S}_i \stackrel{?}{=} h_i^* \cdot V_i^* + \mathcal{U}$$

fails, unless $z = \mathsf{x} + f(i)$:

$$\begin{aligned}
\mathsf{S}_i &\stackrel{?}{=} h_i^* \cdot V_i^* + \mathcal{U} \\
\mathsf{s}_i \cdot G &\stackrel{?}{=} h_i^* \cdot (\hat{\mathsf{X}}_i^* + \mathsf{r}_i \cdot G) + u \cdot G \\
(h_i^* \cdot (\mathsf{x} + f(i)) + \mathtt{sig}_i^*) \cdot G &\stackrel{?}{=} (h_i^* \cdot (z + \mathsf{r}_i) + u) \cdot G \\
h_i^* \cdot (\mathsf{x} + f(i)) + h_i^* \cdot \mathsf{r}_i + u &\stackrel{?}{=} h_i^* \cdot (z + \mathsf{r}_i) + u \\
h_i^* \cdot (\mathsf{x} + f(i)) &\stackrel{?}{=} h_i^* \cdot z \\
\mathsf{x} + f(i) &\stackrel{?}{=} z \quad \triangleright \text{ Assuming } h_i^* \neq 0
\end{aligned}$$

Hence, to bypass the vehicle's verification, the RA cannot just choose any z: it is obliged to make

$$z = \mathsf{x} + f(i).$$

Even though $f(i)$ is known by the RA, finding the value of x that allows the computation of z in this scenario is equivalent to solving the ECDLP for X.

4.1.3 Security Against MitM Attacks by RAs (explicit model)

The security arguments for explicit certificates are similar to those presented in Section 4.1.2 for the implicit model, as summarized as follows.

Suppose that the RA replaces $\hat{\mathsf{X}}_i$ by an arbitrary $\hat{\mathsf{X}}_i^*$ in the request for explicit certificates sent to the PCA. Assuming the hardness of the ECDLP, the RA cannot violate the integrity or confidentiality of the PCA's response `pkg` without the requesting vehicle's knowledge.

Once again, it is easy to show that the explicit certificate $\mathtt{cert}_i$ enclosed in the PCA's encrypted response, `pkg`, cannot be modified while avoiding detection by vehicles. After all, the $\mathtt{cert}_i$ is itself digitally signed by the PCA, so any modification

would invalidate the signature assuming that the PCA employs a secure algorithm for its computation. Therefore, even if the confidentiality of `pkg` is somehow violated by the RA, that might allow the (unsigned) value of r_i to be modified, but not the modification of the (signed) $\texttt{cert}_i$. Indirectly, however, the non-malleability of $\texttt{cert}_i$ also ensures that a possible modification of r_i would be detectable by the vehicle. The reason is that the value of S_i obtained from $\texttt{cert}_i$ is verified by the vehicle when it computes

$$\mathsf{s}_i = \mathsf{r}_i + \mathsf{x} + f(i)$$

and then checks if

$$\mathsf{s}_i \cdot G \stackrel{?}{=} \mathsf{S}_i.$$

Since x and $f(i)$ are known by the vehicle (i.e., cannot be manipulated by the RA), and S_i is fixed in the certificate, turning r_i into $\mathsf{r}_i^* \neq \mathsf{r}_i$ would lead to

$$\mathsf{s}_i^* = \mathsf{r}_i^* + \mathsf{x} + f(i) \neq \mathsf{s}_i$$

and hence to

$$\mathsf{s}_i^* \cdot G \neq \mathsf{S}_i.$$

Therefore, none of the `pkg`'s contents can be modified without detection by the vehicle.

The final verification performed by the vehicle also ensures the confidentiality of the UBK expansion, assuming the hardness of the ECDLP to which this problem can be reduced. To prove this, we once again suppose without loss of generality that the malicious RA replaces $\hat{\mathsf{X}}_i$ by:

$$\hat{\mathsf{X}}_i^* = z \cdot G,$$

for an arbitrarily chosen value of z. In this case, the RA uses z to decrypt the PCA's response and then learns:

1. The device's final public key $\mathsf{S}_i^* = \mathsf{r}_i \cdot G + \hat{\mathsf{X}}_i^*$ enclosed in the certificate;

2. The value of r_i itself.

To avoid detection, the RA would then have to re-encrypt the PCA's response in such a manner that the vehicle does not notice that the RA did not use the $\hat{\mathsf{X}}_i$ in the computation of the received S_i^*. Accomplishing this requires replacing the original r_i by some r_i^* that passes the verification process performed at the vehicle, i.e., that satisfies the equation

$$(\mathsf{r}_i^* + \mathsf{x} + f(i)) \cdot G \stackrel{?}{=} \mathsf{S}_i^*.$$

Otherwise, the vehicle that performs this final verification would identify the received S_i^* as invalid, frustrating the attack. Unfortunately for the RA, however, this means that r_i^* must be set to

$$(\mathsf{r}_i + z) - (\mathsf{x} + f(i)),$$

meaning that finding such r_i^* is equivalent to solving the ECDLP for the point $(\mathsf{S}_i^* - \hat{\mathsf{X}}_i)$. Equivalently, since the RA knows $f(i)$, z can be freely chosen by it, and r_i is learned due to the attack, this problem can be reduced to finding x given the value of X provided by the vehicle. Nevertheless, this is still an ECDLP instance, which concludes our argument.

4.1.4 Implementation-related Security Aspects

As an additional remark, the original SCMS design proposes the adoption of two caterpillar keys most likely because it is considered a good practice to avoid using the same key pair for encryption and signature. The main reason for this recommendation is that possible vulnerabilities (e.g., implementation errors) found in one process may leak the key for the other (CORON et al., 2002). Hence, if an attacker can somehow interact with a vehicle in such a manner that (1) the vehicle works as an oracle for one process, and then (2) recover the private key thereby employed, then (3) that would also give away the private key for the other process.

At first sight, it may seem that the strategy here described violates this general rule by creating a key $\hat{\mathsf{X}}_i$ that is used both for encryption (by the PCA) and for generating

a digital signature (by the vehicles). However, this is *not* the case in the proposed scheme. The reason is that vehicles never use the private key $\hat{x}_i$ (corresponding to $\hat{X}_i$) for signing any piece of data. Instead, vehicles use

$$s_i = \hat{x}_i + r_i$$

as signature keys in the explicit model, and

$$h_i \cdot \hat{x}_i + \mathtt{sig}_i$$

in the implicit model, where r_i and $\mathtt{sig}_i$ are secret values known only by the vehicle and the PCA. Additionally, $\hat{x}_i$ is a secret value known only by the vehicle. As long as $r_i \neq 0$ (for explicit certificates) and $\mathtt{sig}_i \neq 0$ (for implicit ones), any predictable correlation between the encryption and the signature processes is eliminated from the perspective of all entities (as expected from randomly generated keys), except for the PCA itself. Interestingly, this approach follows the same line of thought behind the original butterfly key expansion process that is the basis for SCMS: different signature cocoon keys are generated from the same secret information (the caterpillar key), but this correlation is known only by the vehicle and a system entity (in this case, the RA). Therefore, the proposed modification can is a natural development of the original SCMS protocol.

Additionally, as an exercise, it is useful to consider which kind of implementation flaw would be necessary to jeopardize the resulting system's security. We start by noticing that, even if the signature key u_i is somehow compromised, recovering $\hat{x}_i$ as a result of this flaw would only be feasible by the PCA, since it would still be the only entity (besides the vehicle) with knowledge of the r_i or $\mathtt{sig}_i$ associated to the compromised U_i. However, the PCA would not gain anything by doing so, because it already knows the plaintext protected with $\hat{x}_i$: after all, the PCA is the one who encrypted that plaintext in the first place.

Hence, the only implementation issue that might lead to a useful attack against the UBK process refers to the compromise of the encryption key $\hat{\mathsf{x}}_i$. Such an attack would require capturing the PCA's response carrying r_i and $\mathtt{sig}_i$, so u_i can be recovered, and used for signing messages. Once again, however, this is only feasible by an RA or PCA, but not by external entities. The reason is that the PCA-RA and RA-vehicle communications are protected using secure (e.g., TLS-based) channels (CAMP, 2016), so RA and PCA are the only entities (besides the vehicle) with access to the PCA's response. The RA and the PCA, on the other hand, would not gain much (if anything) by engaging in such attacks, since they could themselves create valid certificates and sign the corresponding messages without any interaction with the final clients. Nevertheless, this scenario does not follow the "honest-but-curious" scenario previously assumed, but a dishonest (or collusion) model which is not covered by the original SCMS nor by this work.

As a final remark, notice that vehicles do not act as decryption oracles in the target scenario: each decryption key $\hat{\mathsf{x}}_i$ is employed only *once*, for the decryption of a single certificate, and a response is never returned (otherwise, the vehicles would put its own privacy at risk). Therefore, (1) only the PCA would actually be able to query vehicles (i.e., send requests containing valid certificates) during attacks aimed at compromising the encryption key $\hat{\mathsf{x}}_i$, and (2) the implementation flaw would have to be severe enough to (2a) include a response channel from vehicles and to (2b) allow the PCA to recover the decryption key from a single query. Consequently, we can conclude that the attack surface introduced by the UBK strategy even in the case of a catastrophic implementation flaw is minimal, or null if we consider SCMS's "honest-but-curious" security model.

4.2 UBK Practical Analisys

Besides preserving SCMS' security properties, this unified butterfly key expansion leads to a reduced overhead when compared to the original process.

For vehicles, the request sent to the RA includes a single cocoon public key and a single PRF rather than two, the processing and bandwidth costs involved in this process drop by half. The batches received are also smaller, because each encrypted package containing a certificate is not signed (only the certificate itself is). Finally, the processing costs for validating batches are smaller than in SCMS, since the verification of the PCA's signature on the encrypted package is eliminated.

The RA, in turn, only performs the butterfly key expansion for signature keys, leading to half the processing overhead. If we Ignore ancillary metadata, the bandwidth usage is similarly reduced when forwarding the request to the PCA, which involves a single cocoon key and a single PRF rather than two of each. Finally, the response by the PCA is also smaller due to the absence of a signature on the encrypted package.

At the PCA, The processing savings come from the fact that each (implicit or explicit) certificate issued takes a single signature instead of two. Inbound and outbound bandwidth is also saved since the RA's requests are smaller (they do not include $\hat{\mathsf{E}}_i$) and so are the PCA's responses (one less signature is sent).

To give some concrete numbers, Table 6 compares the estimated costs of the proposed procedure with the original SCMS as described in (CAMP, 2016), assuming the algorithms thereby recommended: ECDSA for signature generation/verification and ECIES for asymmetric encryption/decryption. Both algorithms are configured to provide a 128-bit security level.

The bandwidth costs are measured in bytes, ignoring eventual metadata not strictly related to the butterfly key expansion process (e.g., the time period to which the cer-

tificate should be associated, and so on). The processing costs are measured in cycles, using the RELIC cryptography library version 0.4.1 (ARANHA; GOUVêA, 2018), running on an Intel i5 4570 processor.

As shown in Table 6, the bandwidth and processing gains of the proposed UBK process can reach up to 50%, whereas in the worst case it is at least as efficient as SCMS' original approach. Interestingly, those gains are slightly more significant in the implicit certification model, which is the one suggested for standardization (CAMP, 2016). The more significant gains are obtained at the vehicles and RAs when processing the requests to be sent to the PCA. In comparison, the verification of implicit certificates requires a (random point) multiplication by the credential V_i; since V_i is unique per pseudonym certificate, there is no advantage in using pre-computation methods for accelerating this operation.

In practice, whereas such benefits at the vehicles' side are clearly important given their resource-constrained nature, we argue that they are also relevant at the servers' side due to the large scale of the vehicular networking scenario. More precisely, to put these savings in perspective, we can consider the case of the PCA: more than 16 million vehicles are sold per year only in US (STATISTA, 2018)). Since the number of certificates provisioned per vehicle is expected to range from a few thousand (WHYTE et al., 2013) to more than 30 thousand (KUMAR; PETIT; WHYTE, 2017), avoiding one signature per certificate translates to 16 – 480 billion signatures that do not need to be computed or transmitted per year by PCAs in the US. Those savings alone are, thus, orders of magnitude larger than the total load of the world's largest PKI, which issues under 10 million certificates every year, run by the US Department of Defense (WHYTE et al., 2013).

As a final remark, notice that the certificate verification process at the vehicles is more efficient in the explicit model than in the implicit model when $\mathcal{U}$ is considered a fixed point. The reason for this is that, with pre-computation, the ECDSA verifications

performed in the explicit model do not involve any random-point multiplication. In comparison, the verification of implicit certificates requires a (random point) multiplication by the credential V_i; since V_i is unique per pseudonym certificate, there is no advantage in using pre-computation methods for accelerating this operation. Despite these higher processing costs at this point, the adoption of implicit certificates does make sense in the context of vehicular communications for at least two reasons (both of which are highlighted in (CERTICOM, 2013, page 17).) The reasons are as follows: (1) it potentially leads to smaller certificates and, hence, to lower bandwidth usage in V2X communications, and (2) during operation, the process of verifying a message's signature can be combined with public key verification for better computational efficiency.

Table 6: Comparison of processing (in cycles, shown in a gray background) and communication (in bytes) costs between the original SCMS and the proposed solution when issuing b explicit and implicit certificates, including request and response.

	Vehicle	→	RA	→	PCA	→	RA	→	Vehicle
SCMS expl.	508×10^3	96	$b\cdot(499\times10^3)$	$b\cdot(64)$	$b\cdot(3.27\times10^6)$	$b\cdot(\|\texttt{cert}\|+80)$	0	$b\cdot(\|\texttt{cert}\|+80)$	$b\cdot(5.30\times10^6)$
UBK expl.	254×10^3	48	$b\cdot(250\times10^3)$	$b\cdot(32)$	$b\cdot(2.86\times10^6)$	$b\cdot(\|\texttt{cert}\|+48)$	0	$b\cdot(\|\texttt{cert}\|+48)$	$b\cdot(3.75\times10^6)$
UBK/SCMS	0.5	0.5	0.5	0.5	0.88	[0.75, 1[‡	–	[0.75, 1[‡	0.71
SCMS impl.	508×10^3	96	$b\cdot(499\times10^3)$	$b\cdot(64)$	$b\cdot(2.86\times10^6)$	$b\cdot(\|\texttt{cert}\|+48)$	0	$b\cdot(\|\texttt{cert}\|+48)$	$b\cdot(5.74\times10^6)$
UBK impl.	254×10^3	48	$b\cdot(250\times10^3)$	$b\cdot(32)$	$b\cdot(2.46\times10^6)$	$b\cdot(\|\texttt{cert}\|+16)$	0	$b\cdot(\|\texttt{cert}\|+16)$	$b\cdot(4.19\times10^6)$
UBK/SCMS	0.5	0.5	0.5	0.5	0.86	[0.67, 1[‡	–	[0.67, 1[‡	0.72

‡ Ignoring encryption overhead and assuming $|\texttt{cert}| \geqslant 48$, which is close to the minimum for a certificate (32 bytes for representing S_i or V_i, plus 16-bytes for metadata such as the expiration date and linkage values)

4.3 Summary

In this chapter, we have presented an improvement to the original SCMS butterfly key expansion, specifically the unified version of the butterfly key expansion.

The UBK expansion improves the original key expansion process by combining the encryption and signature key pairs in a verifiable manner. Consequently, We prevent MitM attacks by RAs even though the PCA does not sign the encrypted package containing the pseudonym certificates.

In summary, the benefits of the UBK optimization are that:

1. It reduces in half the number of keys sent from the vehicle to the RA, as well as from the RA to the PCA (a single key pair replaces two key pairs);

2. It removes one signature from each pseudonym certificate generated by the PCA and, thus, avoids the corresponding costs for their creation, transmission and verification.

The overall result is that UBK reduces the processing and bandwidth utilization for certificate provisioning from 10% up to 50% for all entities involved in the protocol, as we showed in Section 4.2. We obtain this gain while preserving the independence between the encryption key employed by the PCA and the signature key enclosed in pseudonym certificates. Thus, UBK complies with the general guideline of never using the same public-private key pair for both encrypting and signing data.

Additionally, we made an extensive discussion on the security aspects of the original butterfly key expansion and the UBK. Our security discussions covered the following aspects:

- Confidentiality of pseudonym certificates.
- Security against MitM attacks by the RA in the implicit and explicit models.
- Implementation-related attacks.

It is worth noting that our security analysis is informal and follows security arguments instead of formal proofs. This approach is the same as the original SCMS, so we argue that our improved version is as secure as the original SCMS.

Finally, we use the UBK as the basis for our Post-quantum pseudonym certificate provisioning for V2X, which we presented in Chapter 5.

5 THE POST QUANTUM BUTTERFLY KEY EXPANSION

The adaptation of the ECC-based UBK to a quantum-resistant setting requires that underlying schemes to support both encryption and signatures under key pairs that are related to each other. More precisely, the signature key pair needs to be derived from the encryption key pair. Indeed, the public key $\mathsf{S}_i = \mathsf{s}_i \cdot G$ that is used by the vehicle to sign messages is derived (using the randomization factor r_i) from $\hat{\mathsf{X}}_i$, which in turn is used by the PCA to encrypt that vehicle's certificate.

We have chosen lattice-based cryptosystems for this purpose; then, we must take into account some particularities of the target scenario. Notably, the cryptosystems must support the notion of *blind transference of* LWE *samples*, which parallels the possibility of blind transferring elliptic curve points discussed in Section 5.1. This feature depends crucially on the existence of a R-LWE-based encryption (or more precisely, key encapsulation) scheme and a digital signature scheme that:

- Support *additively homomorphic keys*;
- Can use the same structure for the key pair employed in their individual operations;
- Are similarly secure for the same set of distributions and parameters.

The most restrictive of these properties is the first one. It means that the sum of

two private keys $(\mathbf{s}, \mathbf{e})$ and $(\mathbf{s}', \mathbf{e}')$ is still an algebraically admissible private key

$$(\mathbf{s}'', \mathbf{e}'') = (\mathbf{s} + \mathbf{s}', \mathbf{e} + \mathbf{e}'),$$

and (2) and the sum of the corresponding public keys $\mathbf{S}$ and $\mathbf{S}'$ is not only still an algebraically admissible public key

$$\mathbf{S}'' = \mathbf{S} + \mathbf{S}',$$

but actually the same key that naturally corresponds to the sum of public keys. Which is the case, for instance, when the keys have the form

$$\mathbf{S} = \mathbf{s} \cdot \boldsymbol{G} + \mathbf{e}$$

and

$$\mathbf{S}' = \mathbf{s}' \cdot \boldsymbol{G} + \mathbf{e}'$$

whereby

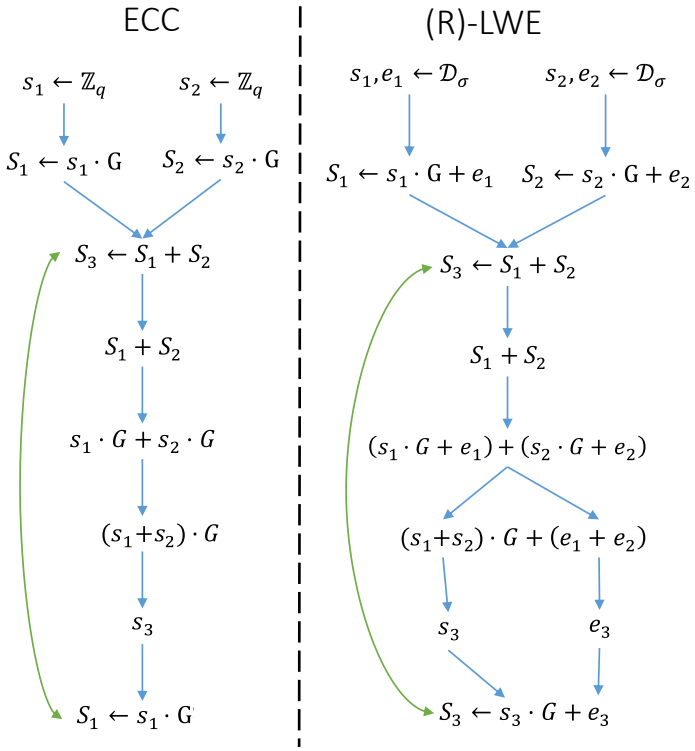
$$\begin{aligned} \mathbf{S}'' &= \mathbf{S} + \mathbf{S}' \\ &= \mathbf{s} \cdot \boldsymbol{G} + \mathbf{e} + \mathbf{s}' \cdot \boldsymbol{G} + \mathbf{e}' \\ &= (\mathbf{s} + \mathbf{s}') \cdot \boldsymbol{G} + (\mathbf{e} + \mathbf{e}') \\ &= \mathbf{s}'' \cdot \boldsymbol{G} + \mathbf{e}''. \end{aligned}$$

Taking such restrictions into account, a Lattice-based analog of the ECC-based UBK is possible with (minor variants of) the qTESLA digital signature scheme and the LPR key encapsulation scheme described in Section 5.2 . It is worth to note that any digital signature scheme and key encapsulation scheme purely based on R-LWE or LWE would work with our post-quantum butterfly key expansion. The construction is wide enough for supporting other schemes, and we give *qTESLA* and *LPR* as examples to instantiate it.

Figure 5 summarizes and shows how the Key Additive Homomorphism from ECC

also exists on Lattice-based cryptography with the R-LWE Problem.

Figure 5: Summary of the Key Additive Homomorphism



Source: the authors

As we show in Figure 5, ECC presents an additive homomorphism in the key pairs. Namely, the addition of the public keys S_1 $(s_1 \cdot G)$ and S_2 $(s_2 \cdot G)$ is also a valid ECC public key. In a more deeply analysis we can see that, S_3 is also the public key related to the addition of the private keys $s_1 + s_2$, which is the addition of the private keys related to S_1 and S_2 respectively. Such a feature is essential for the UBK; fortunately, R-LWE-based cryptography has the same feature. As for ECC, in R-LWE the addition of the public keys S_1 $(s_1 \cdot G + e_1)$ and S_2 $(s_2 \cdot G + e_2)$ is also a valid R-LWE public key. Similarly to ECC in R-LWE S_3 is also the public key related to the addition of the private keys $s_1 + s_2$ and $e_1 + e_2$.

5.1 Blind Transference of LWE Samples

Intuitively, the blind transference of LWE samples in qUBK relies on the (pseudo)random sampling of (short and secret) ring elements by the different entities that participate in the protocol, namely vehicle, RA and PCA. Whenever such short elements are added to a public key, they create other valid keys that are indistinguishable from each other under the R-LWE assumption. This indistinguishability property is quite similar to what happens when known ECC points are added to secret random points, as in the original UBK.

More formally, consider the following definition:

Definition 6. *A* σ_0-sample *is a ring element on* $\mathcal{R}$ *sampled from the (zero-centered) Gaussian distribution with standard deviation* σ.

Let $\boldsymbol{G}$ and $\boldsymbol{H}$ be uniformly sampled from $\mathcal{R}_q$. And let $\boldsymbol{s}$ and $\boldsymbol{e}$ be $\mathcal{D}_\sigma^n$-samples. The R-LWE assumption for a sample of form

$$\boldsymbol{L} \leftarrow \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e}$$

is that distinguishing between $\boldsymbol{L}$ and $\boldsymbol{H}$ is hard. Therefore, we can write the following definition on the indistinguishability:

Definition 7. *Let* $\boldsymbol{s}$ *and* $\boldsymbol{e}$ *be* σ*-samples. A ring element of form*

$$\boldsymbol{L} \leftarrow \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e}$$

will be called σ-indistinguishable *(from a uniform random sample) under the R-LWE assumption.*

Here we assume that the R-LWE assumption holds at the desired security level for all $\sigma \in [\sigma_0, \sqrt{k}\sigma_0]$ where $k \in \{1, 2, 3\}$. If so:

- The vehicle's ring elements, of form

$$\mathbf{X} \leftarrow \mathbf{s} \cdot \boldsymbol{G} + \mathbf{e}$$

for σ_0-samples $\mathbf{s}$ and $\mathbf{e}$, are σ_0-indistinguishable from the point of view of the RA, the PCA, and other vehicles.

- The RA's ring elements, of form

$$\hat{\mathbf{X}}_i \leftarrow \mathbf{X} + f_{\mathcal{D}}(i) \cdot \boldsymbol{G} + g_{\mathcal{D}}(i)$$

for $f_{\mathcal{D}}(i)$, $g_{\mathcal{D}}(i)$ simulating σ_0-sampling, are $\sqrt{2}\sigma_0$-indistinguishable from the point of view of the PCA and other vehicles, since they do not know $f_{\mathcal{D}}(i)$ and $g_{\mathcal{D}}(i)$.

This is because the RA is essentially masking the σ_0-indistinguishable element $\mathbf{X}$ with another σ_0-indistinguishable element, yielding a ring element identical to that obtained from ring elements

$$\begin{aligned} \hat{\mathbf{s}}_i &\leftarrow \mathbf{s} + f_{\mathcal{D}}(i) \\ \hat{\mathbf{e}}_i &\leftarrow \mathbf{e} + g_{\mathcal{D}}(i). \end{aligned}$$

which are themselves identical to $\sqrt{2}\sigma_0$-samples by virtue of being the sum of two σ_0-samples each.

- The ring elements that are signed by the PCA, and have the form

$$\mathbf{S}_i \leftarrow \hat{\mathbf{X}}_i + (\mathbf{s}'_i \cdot \boldsymbol{G} + \mathbf{e}'_i)$$

for σ_0-samples $\mathbf{s}'_i$ and $\mathbf{e}'_i$, are $\sqrt{3}\sigma_0$-indistinguishable from the point of view of the RA and other vehicles.

This is because the PCA is essentially masking the $\sqrt{2}\sigma_0$-indistinguishable element $\hat{\mathbf{X}}_i$ with a σ_0-indistinguishable element, yielding a ring element identical

to that obtained from (the vehicle's new secret) ring elements

$$\begin{aligned} \mathbf{s}_i &\leftarrow \hat{\mathbf{s}}_i + \mathbf{s}'_i \\ \mathbf{e}_i &\leftarrow \hat{\mathbf{e}}_i + \mathbf{s}'_i. \end{aligned}$$

which are themselves identical to $\sqrt{3}\sigma_0$-samples by virtue of being the sum of a $\sqrt{2}\sigma_0$-sample and a σ_0-sample each.

5.2 The Protocol

In what follows, we use bold capital letters ($\boldsymbol{S}$) to represent public uniformly random ring elements and small bold letters ($\boldsymbol{s}$) to represent secret short ring elements.

Let $s{\cdot}\boldsymbol{G} + \varepsilon = \boldsymbol{\mathcal{U}}$, the Table 7 compares the ECC-based (pre-quantum) and the proposed lattice-based (post-quantum) variants of the explicit butterfly protocol.

Table 7: ECC and Lattice-based protocol comparison

	Vehicle	→	RA	→	PCA	-RA›	Vehicle
explicit	$\mathsf{x} \xleftarrow{\$} \mathbb{Z}_q$ $\mathsf{X} \leftarrow \mathsf{x}{\cdot}G$	X, f	$\hat{\mathsf{X}}_i \leftarrow \mathsf{X} + f(i){\cdot}G$ $(0 \leqslant i < b)$	$\hat{\mathsf{X}}_i$	$\mathsf{r}_i \xleftarrow{\$} \mathbb{Z}_q$ $\mathsf{S}_i \leftarrow \hat{\mathsf{X}}_i + \mathsf{r}_i{\cdot}G$ $\mathtt{sig}_i \leftarrow Sign(u, \{\mathsf{S}_i, \mathtt{meta}\})$ $\mathtt{cert}_i \leftarrow \{\mathsf{S}_i, \mathtt{meta}, \mathtt{sig}_i\}$ $\mathtt{pkg} \leftarrow Enc(\hat{\mathsf{X}}_i, \{\mathtt{cert}_i, \mathsf{r}_i\})$	pkg	$\hat{\mathsf{x}}_i \leftarrow \mathsf{x} + f(i)$ $\{\mathtt{cert}_i, \mathsf{r}_i\} \leftarrow Dec(\hat{\mathsf{x}}_i, \mathtt{pkg})$ $Ver(\boldsymbol{\mathcal{U}}, \mathtt{cert}_i)$ $\mathsf{s}_i \leftarrow \hat{\mathsf{x}}_i + \mathsf{r}_i$ NB: $\mathsf{s}_i{\cdot}G = \mathsf{S}_i$
explicit R-LWE	$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \mathcal{D}^n_\sigma$ $\mathbf{X} \leftarrow \mathbf{s}{\cdot}\boldsymbol{G} + \mathbf{e}$ $\mathbf{x} = \{\mathbf{s}, \mathbf{e}\}$	$\mathbf{X}$, $f_{\mathcal{D}}, g_{\mathcal{D}}$	$\hat{\mathbf{X}}_i \leftarrow \mathbf{X} + f_{\mathcal{D}}(i){\cdot}\boldsymbol{G} + g_{\mathcal{D}}(i)$ $(0 \leqslant i < b)$	$\hat{\mathbf{X}}_i$	$\mathbf{s}'_i, \mathbf{e}'_i \xleftarrow{\$} \mathcal{D}^n_\sigma[seed_i]$ $\mathbf{S}_i \leftarrow \hat{\mathbf{X}}_i + (\mathbf{s}'_i{\cdot}\boldsymbol{G} + \mathbf{e}'_i)$ $\mathtt{sig}_i \leftarrow Sign(u, \{\mathbf{S}_i, \mathtt{meta}\})$ $\mathtt{clip}_i \leftarrow (seed_i, \mathtt{meta}, \mathtt{sig}_i)$ $\mathtt{pkg} \leftarrow Enc(\hat{\mathbf{X}}_i, \mathtt{clip}_i)$	pkg	$\hat{\mathbf{s}}_i \leftarrow \mathbf{s} + f_{\mathcal{D}}(i), \hat{\mathbf{e}}_i \leftarrow \mathbf{e} + g_{\mathcal{D}}(i)$ $\hat{\mathbf{x}}_i = \{\hat{\mathbf{s}}_i, \hat{\mathbf{e}}_i\}$ $\mathtt{clip}_i \leftarrow Dec(\hat{\mathbf{x}}_i, \mathtt{pkg})$ $\mathbf{s}'_i, \mathbf{e}'_i \xleftarrow{\$} \mathcal{D}^n_\sigma[seed_i]$ $\mathbf{S}_i \leftarrow \hat{\mathbf{X}}_i + (\mathbf{s}'_i{\cdot}\boldsymbol{G} + \mathbf{e}'_i)$ $Ver(\boldsymbol{\mathcal{U}}, \{\mathbf{S}_i, \mathtt{meta}\})\mathtt{sig}_i$ $\mathtt{cert}_i \leftarrow (\mathbf{S}_i, \mathtt{meta}, \mathtt{sig}_i)$ $\mathbf{s}_i \leftarrow \hat{\mathbf{s}}_i + \mathbf{s}'_i, \mathbf{e}_i \leftarrow \hat{\mathbf{e}}_i + \mathbf{e}'_i$ $\mathbf{s}_i = \{\mathbf{s}_i, \mathbf{e}_i\}$ NB: $\mathbf{s}_i{\cdot}\boldsymbol{G} + \mathbf{e}_i = \mathbf{S}_i$

5.2.1 Signature Scheme

For doing the PCA signature, we have chosen the qTESLA signature Schema. The qTESLA is especially a good fit for a post-quantum version of UBK because it requires high-quality Gaussian sampling only for key generation. As a result, vehicles do not

need to do such an operation while signing messages in the field. Such a feature is essential given the constrained nature of vehicle embedded systems which would not be able to produce the Gaussian sampler in real-time.

Signing only requires functions $f(i)$ and $g(i)$ to deterministically emulate sampling from D_σ^n, with i as the seed. Accordingly, the PCA must sample $\mathbf{s}_i'$ and $\mathbf{e}_i'$ from D_σ^n. Encryption will impose additional constraints on these functions.

From the R-LWE assumption,

$$\mathbf{S} = \mathbf{s} \cdot \boldsymbol{G} + \mathbf{e}$$

is indistinguishable from uniformly random for $\mathbf{s}$ and $\mathbf{e}$ sampled with distribution parameter σ, while the

$$\hat{\mathbf{S}}_i = \hat{\mathbf{s}}_i \cdot \boldsymbol{G} + \hat{\mathbf{e}}_i$$

are indistinguishable from uniformly random for $\hat{\mathbf{s}}_i$ and $\hat{\mathbf{e}}_i$ sampled with distribution parameter $\sqrt{2}\sigma$ by virtue of these secret components being each the sum of 2 identically parameterized Gaussian variables, namely

$$\begin{aligned} \hat{\mathbf{s}}_i &= \mathbf{s} + f_{\mathcal{D}}(i) \\ \hat{\mathbf{e}}_i &= \mathbf{e} + g_{\mathcal{D}}(i). \end{aligned}$$

Additionally the

$$\mathbf{S}_i = \mathbf{s}_i \cdot \boldsymbol{G} + \mathbf{e}_i$$

is indistinguishable from uniformly random for s_i and $\mathbf{e}_i$ sampled with distribution parameter $\sqrt{3}\sigma$ by virtue of these secret components being each the sum of 3 such variables, namely

$$\begin{aligned} \mathbf{s}_i &= \mathbf{s} + f_{\mathcal{D}}(i) + \mathbf{s}_i' \\ \mathbf{e}_i &= \mathbf{e} + g_{\mathcal{D}}(i) + \mathbf{e}_i'. \end{aligned}$$

The actual scheme parameters must be chosen to take this into account (they must remain secure at the desired level or above for all of these distribution parameters). Furthermore, the parameters must ensure that all signature operations are efficient with distribution parameter $\sqrt{3}\sigma$ (and related quantities, e.g., the qTESLA parameters L_S and L_E) since the final certificate is equivalent to keys prepared according to this setting.

5.2.2 Encryption Scheme

Encryption requires functions $f_{\mathcal{D}}(i)$ and $g_{\mathcal{D}}(i)$ to deterministically emulate sampling from D^n_σ, with i as the seed, until $f_{\mathcal{D}}(i), g_{\mathcal{D}}(i) \in \mathcal{R}_q^\times$, that is, they must be both invertible. Accordingly, the PCA must sample $\mathbf{s}'_i, \mathbf{e}'_i$ from D^n_σ until $\mathbf{s}'_i, \mathbf{e}'_i \in \mathcal{R}_q^\times$. Like the secret $\mathbf{e}$ component, the $\hat{\mathbf{e}}_i$ and $\mathbf{e}_i$ components need not actually be computed, nor $\mathbf{e}'_i$ kept after the certificate generation.

Let $\bar{\boldsymbol{G}} := -\boldsymbol{G}$, to bridge the qTESLA and LPR notations. From the RLWE assumption,

$$\mathbf{S} = \mathbf{e} - \mathbf{s} \cdot \bar{\boldsymbol{G}}$$

is indistinguishable from uniformly random for $\mathbf{s}$ and $\mathbf{e}$ sampled with distribution parameter σ, while the

$$\hat{\mathbf{S}}_i = \hat{\mathbf{e}}_i - \hat{\mathbf{s}}_i \cdot \bar{\boldsymbol{G}}$$

are indistinguishable from uniformly random for $\hat{\mathbf{s}}_i$ (and $\hat{\mathbf{e}}_i$) sampled with distribution parameter $\sqrt{2}\sigma$ by virtue of these secret components being each the sum of 2 identically parameterized Gaussian variables, namely

$$\begin{aligned} \hat{\mathbf{s}}_i &= \mathbf{s} + f_{\mathcal{D}}(i) \\ \hat{\mathbf{e}}_i &= \mathbf{s} + g_{\mathcal{D}}(i). \end{aligned}$$

Encryption occurs in the scheme only under the key pair $(\hat{\mathbf{s}}_i, \hat{\mathbf{S}}_i)$ (and implicitly

$\hat{\mathbf{e}}_i$), so in principle, the actual scheme parameters could be chosen to take only this into account, but the resulting signature key pair already forces a more stringent condition. If the resulting key pair is used not only for signatures but for encryption as well, then the parameters must be double-checked (although the requirements for encryption tend to be less stringent than those for signatures, e.g., the distributions need not be as precise).

5.3 Handling Signature and Decryption Failures

When analyzing the building blocks that compose qUBK, note that qTESLA requires the private key samples to satisfy $\mathsf{checkS}(\mathbf{s})$ and $\mathsf{checkE}(\mathbf{e})$, which means that these conditions must hold true for the vehicle's private key $\mathsf{s}_i = \{\mathbf{s}_i, \mathbf{e}_i\}$. However, the PCA cannot perform this verification since it does not know $\mathbf{s}_i$ nor $\mathbf{e}_i$, which are only known by the vehicles themselves. As a result, the vehicle is forced to reject a received key if it fails to pass either of those checks. Otherwise, accepting such keys might lead to failure when verifying genuine signatures.

Fortunately, it is possible to choose parameters such that the probability of key rejection is reasonably low. Hence, by provisioning vehicles with a number of certificates that is slightly larger than the minimum necessary for their operation, eventual key rejections should not be an actual concern. Interestingly, this also covers the possibility of decryption failure for LPR encryption. This happens because (as discussed in Section 5.2.2) the chance of decryption failure can be negligible even though one must use qTESLA parameters and keys.

Section 5.6 gives some specially tailored parameters for keeping key, signature, and decryption rejection rates under control, for different security levels.

5.4 Post-Quantum Implicit Certificates

If parameters comparable to those of the underlying LPR and qTESLA schemes are adopted, the proposed protocol is *not* amenable to being expressed in an implicit form that is at once secure and efficient.

The reason is that a signing equation like

$$\mathtt{sig}_i \leftarrow \mathcal{H}(\mathtt{cert}_i) \cdot \mathsf{r}_i + u$$

must always involve at least one long element (typically either $\mathcal{H}(\mathtt{cert}_i)$, or r_i, or both) to protect the private key u, and hence yields a long result $\mathtt{sig}_i$ as well, rendering the implicit key with the form

$$\mathsf{s}_i \leftarrow h_i \hat{\mathsf{s}}_i + \mathtt{sig}_i.$$

which are invariably long and thus unsuitable as a private credential key in a LWE setting.

Even plain signature schemes where the signatures are designed to be fairly short vectors (like the BLISS family (DUCAS et al., 2013)) end up yielding ring elements that are much (in the BLISS, quadratically) longer than typical private keys. It is not presently clear how to obtain parameters that could support the further use of such elements themselves as private keys. At the very least, they would have to be quadratically larger (and hence quadratically more inefficient) than plain parameters.

Additionally, it might be possible to adopt the Gentry-Peikert-Vankuntanathan (GPV) trapdoor preimage sampling technique (GENTRY; PEIKERT; VAIKUNTANATHAN, 2008) to generate certificates whose signatures are short lattice vectors. Trapdoor preimage sampling theoretically paves the way to many other cryptographic primitives like blind signatures and identity-based encryption (Gentry, Peikert and Vaikuntanathan (2008) already suggested an IBE scheme using this approach), but it is not presently clear what the parameters would look like to attain suitable security

levels.

In fact, there exists works toward such advanced schemes seem to indicate severe parameter deterioration. For instance, the only lattice-based blind signature scheme known so far (RÜCKERT, 2010) incurs bandwidth/storage and processing time requirements that are several orders of magnitude larger than qTESLA or BLISS, while only attaining legacy-level security (below 2^{100}).

We, therefore, left further exploration of the (in)feasibility of implicit schemes for future research.

5.5 Security Discussion

As discussed in Section 4.1 the security goals of SMCS are to ensure:

- The confidentiality of the vehicle's private key s_i;
- The confidentiality of the pseudonym certificate generated by the PCA toward other entities;
- The integrity of the pseudonym certificates in the PCA's response; and
- The unlinkability of the pseudonym certificates with respect to any entity (e.g., vehicles, RSU, and so on)

Note that the stated above properties are expected to hold as long as PCA and RA do not collude. We provide a more detailed security analysis of the UBK in (SIMPLICIO et al., 2018).

The security and privacy properties of SCMS rely on the fact that a public key on an ECC scheme (namely an elliptic curve point $P = s \cdot G$, where s is the private key) is indistinguishable from a completely random point. Additionally, from this ECC property, SCMS derives its privacy properties. On the other hand, signature unforgeability

and ciphertext indistinguishability underlie on the corresponding security of each protocol, so we refer them to the original publications (NIST, 2013; BERNSTEIN et al., 2012; IEEE, 2004).

We argue that qSCMS certificate provisioning process, using the qUBK protocol, provides the same security features as the ECC-based SCMS. Our argument comes from the fact that, as for ECC, R-LWE public keys are also indistinguishable from random. Particularly, according to the R-LWE assumption, a public key on a R-LWE scheme (i.e., a ring element $\boldsymbol{P} = \boldsymbol{s} \cdot \boldsymbol{G} + \boldsymbol{e}$, where $\boldsymbol{s}$ and $\boldsymbol{e}$ are the private key) is indistinguishable from a completely random ring element. On the other hand, the signature unforgeability and ciphertext indistinguishability also underlie on security properties of the individual protocol components, namely qTESLA signature, and LPR encryption schemes (AKLEYLEK et al., 2017; LYUBASHEVSKY; PEIKERT; REGEV, 2010).

Therefore, the base of qSCMS' security properties are the same ones presented in the original UBK (SIMPLICIO et al., 2018). Once the R-LWE assumption is true, the property of being indistinguishable from random guarantees to qSCMS the same privacy properties as the original SCMS.

5.6 qUBK Practical Analisys

The practical parameters were obtained by a modification of the NIST PQC standard candidate qTESLA (AKLEYLEK et al., 2017) Sage script, which in turn is based on the LWE-Estimator script available at (ALBRECHT, 2017) and (ALBRECHT et al., 2017b). The analysis of all lattice-based NIST candidates is available at (ALBRECHT et al., 2017a).

Table 8 shows the value for each parameter divided in three security level parameter sets.

The key, signature, certificate, and encrypted package sizes are shown in Table 9.

Table 8: Parameter sets

Parameter	2^{128} sec. level	2^{192} sec. level	2^{256} sec. level
q	16091137	25366529	55308289
k	10	11	11
σ	14.71	14.71	14.71
h_{max}	36	50	72
L_E	1324	1839	2648
L_S	1324	1839	2648
d	22	23	24
B	$2^{21} - 1$	$2^{22} - 1$	$2^{23} - 1$

The sizes are given in bytes and are divided into three security level parameter sets.

A public certificate $\texttt{cert}_i$ contains:

- A qTESLA public key $\mathbf{S}_i$ of size $|\mathbf{S}_i| = n\lceil \lg q \rceil$ bits.

  Note that, a general qTESLA signature would include a κ-bit seed for the pseudo-random choice of the base $\boldsymbol{G}$, but this is omitted here since the same $\boldsymbol{G}$ must be used system wide to preserve the vehicle's anonymity);

- Meta-data of unspecified size $|\texttt{meta}| = \mu$ bits;

- A qTESLA signature $\texttt{sig}_i$ of size $|\texttt{sig}_i| = \kappa + n(\lceil \lg(B - L_S) + 1 \rceil)$ bits (more precisely, a hash value of size κ bits and a ring element in $\mathcal{R}_q$ with 0-centered coefficients not exceeding $B - L_S$ in absolute value).

Hence the certificate size in bits is:

$$\begin{aligned} |\texttt{cert}_i| &= |\mathsf{S}_i| + |\texttt{meta}| + |\texttt{sig}_i| \\ &= n\lceil \lg q \rceil + \mu + \kappa + n(\lceil \lg(B - L_S) + 1 \rceil) \end{aligned}$$

An encrypted clipped certificate pkg contains:

- A LPR encryption capsule $[\boldsymbol{C}, \boldsymbol{D}]$ of size $|[\boldsymbol{C}, \boldsymbol{D}]| = |\boldsymbol{C}| + |\boldsymbol{D}| = (n + \lambda)\lceil \lg q \rceil$ bits (where $\boldsymbol{C}$ is the encryption nonce and $\boldsymbol{D}$ is the encapsulation of a λ-bit symmetric key);

- A seed $seed_i$ of size $|seed_i| = \kappa$ bits for the blind LWEtransfer;

- Meta-data of size $|\texttt{meta}| = \mu$ bits;
- A qTESLA signature $\texttt{sig}_i$ of size $|\texttt{sig}_i| = \kappa + n(\lceil \lg(B - L_S) + 1 \rceil)$ bits;
- A MAC tag τ_i of size κ bits as part of the authenticated symmetric encryption of the clipped certificate.

Hence the Encrypted package size in bits is:

$$\begin{aligned} |\texttt{pkg}| &= |[\boldsymbol{C}, \boldsymbol{D}]| + |seed_i| + |\texttt{meta}| + |\sigma_i| + |\tau_i| \\ &= (n + \lambda)\lceil \lg q \rceil + 3\kappa + n(\lceil \lg(B - L_S) + 1 \rceil) + \mu \end{aligned}$$

Table 9: Key and Signature Sizes in bytes

	2^{128} sec. level	2^{192} sec. level	2^{256} sec. level
Public Key Size	3104	6432	6688
Secret Key Size	2080	4128	4640
Signature size	2848	5920	6176
Certificate size	5920	12320	12832
Encrypted package	5216	10649	11232

Table 10 Gives the processing costs are measured in cycles, using the qTESLA implementation as the basis, running on an Intel i5 4570 processor.

Table 10: Processing costs of qSCMS.

	Vehicle	RA	PCA	Vehicle
qSCMS expl.	11.6×10^6	$b\cdot(15.3\times10^6)$	$b\cdot(27.5\times10^6)$	$b\cdot(27.3\times10^6)$

Interestingly, the 192-bit level parameters appear to offer little bandwidth/storage/processing improvements over the 256-bit parameters. Such behavior is not really surprising since there is no suitable intermediate lattice dimension between $n = 1024$ and $n = 2048$, since n must has the form $n = 2^k$ for the usual ring $\mathcal{R}_q$. This suggests limiting parameters to the 128-bit and 256-bit levels. More options may be available with other rings (e.g., modular lattices, which appear to support $n = 1536$

without an undue decrease in the security level), but since they are less scrutinized, we cannot presently recommend such choices.

5.7 Summary

In this Chapter, we have present our post-quantum protocol for pseudonym certificate provisioning to be used in V2X communications.

Our solution aims to tackle the lack of quantum secure solutions for provisioning certificates in the V2X scenario. Currently, most well-accepted solutions rely on classical cryptographic primitives, like ECC. Thus they cannot withstand the possible threat of quantum computers, which are expected to become a reality in the future (CHEN et al., 2016).

We have adapted the UBK expansion proposed in Chapter 4, which in turn is an optimization of the SCMS butterfly key expansion presented in Chapter 3. As a result, we were able to build a Lattice-based protocol version of the UBK; in particular, we could get a solution based on the R-LWEsecurity assumption.

We also provide a practical analysis of our Post-quantum solution in Section 5.6. Our analysis includes practical parameter sets, which lead to competitive key and signature sizes, as well as bandwidth usage. By combining qUBK with SCMS's original linkage process, we here propose a post-quantum SCMS (qSCMS) architecture that, to the best of our knowledge, is the first post-quantum VPKI in the literature.

6 CONCLUSIONS

Data authentication and user privacy are essential for preventing abuse in intelligent transportation systems, either by drivers or by the system itself. Achieve those features are, however, a challenging task, in particular, because any acceptable solution needs to cope with constraints on the vehicle's side such as limited connectivity and processing power. Fortunately, SCMS's pseudonym certificates provisioning and revocation processes can address such requirements while also taking into account performance and scalability issues.

Despite these advances, we have shown here that there are still optimization opportunities in the SCMS architecture. Specifically, we describe a novel Unified Butterfly Key expansion, in which we replace two vehicle-provided key pairs by a single one. Besides eliminating the need for including such extra key in the vehicle's requests, this approach also removes one signature from each pseudonym certificate generated in response (and, hence, the corresponding costs for their creation, transmission, and verification). As a result, when compared to SCMS's pseudonym certificate provisioning protocol, we can obtain processing and bandwidth savings (both downstream and upstream) that reach up to 50%. Such savings are especially relevant when considering that the number of certificates provisioned per vehicle is expected to range from a few thousand (WHYTE et al., 2013) to tens of thousands (KUMAR; PETIT; WHYTE, 2017). Also, whereas these gains are more noticeable at the vehicles' side, which are exactly the most resource-constrained entities in the system, they are also relevant from the PCA's and RA's perspective given a large number of certificates that need to

be handled by the system. The proposed schemes support either implicit or explicit certificates, while still preserving the system's security, flexibility, and scalability in both approaches.

It is worth noting that the previously cited solution for provisioning certificates relies on ECC, which is a classical cryptographic primitive, which means that it is unable to withstand the threat possibly posed by quantum computers, which are expected to become a reality in the future (CHEN et al., 2016). Aiming to tackle this issue, we have presented a post-quantum protocol for certificate provisioning for V2X.

Our protocol is an adaption of the UBK process, presented in chapter 4, which in turn is a simplification and optimization of the well-known butterfly key expansion process (from the original SCMS architecture) (CAMP, 2016). The resulting post-quantum UBK (qUBK) uses lattices as the underlying primitive, in particular, schemes based on the R-LWEsecurity assumption. We also provide practical parameter sets for different security levels, which lead to competitive key and signature sizes, as well as bandwidth usage, when compared to other lattice-based protocols. By combining qUBK to the original SCMS's linkage process, we here propose a post-quantum SCMS (qSCMS) architecture that, to the best of our knowledge, is the first post-quantum VPKI in the literature.

7 LIST OF PUBLICATIONS AND CONTRIBUTIONS

Dissertation main Publications and contributions

- Marcos A. Simplicio, Eduardo L. Cominetti, Harsh Kupwade Patil, Jefferson E. Ricardini and Marcos V. M. Silva, *The Unified Butterfly Effect: Efficient Security Credential Management System for Vehicular Communications*, 2018 IEEE Vehicular Networking Conference (VNC), Taipei, Taiwan, pp. 1-8. doi: 10.1109/VNC.2018.8628369,2018.

- Paulo S. L. M. Barreto, Jefferson E. Ricardini, Marcos A. Simplicio Jr., Harsh Kupwade Patil. *qSCMS: Post-quantum certificate provisioning process for V2X*. Cryptology ePrint Archive, Report 2018/1247 <https://eprint.iacr.org/2018/1247>, 2018. Pre-print version. To be submitted soon.

Dissertation side Publications and contributions

- Jefferson E. Ricardini, Paulo S. L. M. Barreto. *A New Matrix Algebra for LWE Encryption*. IEEE Latin America Transactions, v. 13, n. 9, p. 3038-3043, 2015.

- Paulo S. L. M. Barreto, Patrick Longa, Michael Naehrig, Jefferson E. Ricardini, Gustavo Zanon. *Sharper Ring-LWE Signatures*. Cryptology ePrint Archive Report 2016/1026 <https://eprint.iacr.org/2016/1026>, 2016. Pre-print version.

- M. A. Simplicio Junior and E. Lopes Cominetti and H. Kupwade Patil and J. Ricardini and L. Ferraz and M. V. Silva. *Privacy-Preserving Method for Temporarily Linking/Revoking Pseudonym Certificates in VANETs*. 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And

Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018. p.1322-1329, ISSN 2324-9013.

- Marcos A. Simplicio Jr., Eduardo L. Cominetti, Harsh Kupwade Patil, Jefferson E. Ricardini , Leonardo T. D. Ferraz , Marcos V. M. Silva. *Privacy-preserving linkage/revocation of VANET certificates without LAs*. Cryptology ePrint Archive, Report 2018/788 <https://eprint.iacr.org/2018/788>, 2018. pre-print version. To be submitted soon.

- Marcos A. Simplicio, Eduardo Lopes Cominetti, Harsh Kupwade Patil, Jefferson E. Ricardini, Marcos Vinicius M. Silva. *ACPC: Efficient revocation of pseudonym certificates using activation codes*. Ad Hoc Networks. 2018. ISSN 1570-8705.

- Paulo S. L. M. Barreto, Marcos A. Simplicio Jr., Jefferson E. Ricardini, Harsh Kupwade Patil. *Schnorr-based implicit certification: improving the security and efficiency of V2X communications*. Cryptology ePrint Archive, Report 2019/157 <https://eprint.iacr.org/2019/157>, 2019. Pre-print version. To be submitted soon.

- Gustavo Banegas, Paulo S. L. M. Barreto, Brice O. Boidje, Pierre-Louis Cayrel, Gilbert N. Dione, Kris Gaj, Cheikh T. Gueye, Richard Haeussler, Jean B. Klamti, Ousmane N'diaye, Duc T. Nguyen, Edoardo Persichetti, Jefferson E. Ricardini. *DAGS: Key encapsulation using dyadic GS codes*. Journal of Mathematical Cryptology, 12.4 (2018): 221-239, 2018

- Gustavo Banegas, Paulo S. L. M. Barreto, Brice O, Boidje, Pierre-Louis Cayrel, Gilbert N. Dione, Kris Gaj, Cheikh T. Gueye, Richard Haeussler, Jean B. Klamti, Ousmane N'diaye, Duc T. Nguyen, Edoardo Persichetti, Jefferson E. Ricardini. *DAGS: Reloaded Revisiting Dyadic Key Encapsulation*. Cryptology ePrint Archive, Report 2018/1203 <https://eprint.iacr.org/2018/1203>, 2018. pre print

version.

- Erdem Alkim, Paulo S. L. M. Barreto, Nina Bindel, Patrick Longa, Jefferson E. Ricardini. *The Lattice-Based Digital Signature Scheme qTESLA*. Cryptology ePrint Archive, Report 2019/085 <https://eprint.iacr.org/2019/085>, 2019. Preprint version. Submitted to TCHES 2019.

REFERENCES

ABDALLA, M.; BELLARE, M.; ROGAWAY, P. *DHAES: An encryption scheme based on the Diffie-Hellman problem*. 1998.

ABDALLA, M.; BELLARE, M.; ROGAWAY, P. The oracle diffie-hellman assumptions and an analysis of dhies. In NACCACHE, D. (Ed.). *Topics in Cryptology — CT-RSA 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 143–158. ISBN 978-3-540-45353-6.

AJTAI, M. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM, 1996. (STOC '96), p. 99–108. ISBN 0-89791-785-5. Available from Internet: <http://doi.acm.org/10.1145/237814.237838>.

AKLEYLEK, S. et al. *Lattice-based digital signature scheme qTESLA*. 2017. QTESLA official website. <https://qtesla.org/>.

AKLEYLEK, S. et al. An efficient lattice-based signature scheme with provably secure instantiation. In *Progress in Cryptology - AFRICACRYPT 2016 - 8th International Conference on Cryptology in Africa, Fes, Morocco,2016*. Fes, Morocco: Springer, 2016. p. 44–60.

ALBRECHT, M. R. *lwe-estimator*. 2017. Available from Internet: <https://bitbucket.org/malb/lwe-estimator>.

ALBRECHT, M. R. et al. *Estimate all the {LWE, NTRU} schemes!* 2017. Available from Internet: <https://estimate-all-the-lwe-ntru-schemes.github.io/paper.pdf?v=aug18>.

ALBRECHT, M. R. et al. *Estimate all the {LWE, NTRU} schemes! - code*. 2017. Available from Internet: <https://bitbucket.org/malb/lwe-estimator>.

ALKIM, E. et al. *TESLA: Tightly-Secure Efficient Signatures from Standard Lattices*. 2015. Cryptology ePrint Archive, Report 2015/755. Available from Internet: <http://eprint.iacr.org/2015/755>.

ARANHA, D.; GOUVêA, C. *RELIC is an Efficient LIbrary for Cryptography*. 2018. <https://github.com/relic-toolkit/relic>.

BABAI, L. On Lovász's lattice reduction and the nearest lattice point problem. *Combinatorica*, Springer-Verlag, vol. 6, no. 1, p. 1–13, 1986. ISSN 0209-9683.

BAI, S.; GALBRAITH, S. D. An improved compression technique for signatures based on learning with errors. In BENALOH, J. (Ed.). *RSA Conference – Cryptographer's Track – CT-RSA 2014*. San Francisco, CA, USA: Springer, 2014. (Lecture Notes in Computer Science, vol. 8366), p. 28–47.

BANSARKHANI, R. E.; MISOCZKI, R. *G-Merkle: A Hash-Based Group Signature Scheme From Standard Assumptions*. 2018. Cryptology ePrint Archive, Report 2018/274. <https://eprint.iacr.org/2018/274>.

BARRETO, P. S. L. M. et al. A panorama of post-quantum cryptography. In KOÇ, Ç. K. (Ed.). *Open Problems in Mathematics and Computational Science*. Cham: Springer International Publishing, 2014. p. 387–439.

BARRETO, P. S. L. M. et al. *Sharper Ring-LWE Signatures*. 2016. Cryptology ePrint Archive, Report 2016/1026. Available from Internet: <https://eprint.iacr.org/2016/1026>.

BATCHER, K. E. Sorting networks and their application. In *AFIPS Spring Joint Computer Conference*. Atlantic City (NJ): Thomson Book Company, Washington D.C., 1968. (AFIPS Conference Proceedings, vol. 32), p. 307–314.

BELLARE, M.; ROGAWAY, P. Minimizing the use of random oracles in authenticated encryption schemes. In *ICICS*. Beijing, China: Springer, 1997.

BERNSTEIN, D. Cost analysis of hash collisions : will quantum computers make SHARCS obsolete? In *SHARCS'09 Workshop Record – Proceedings 4th Workshop on Special-purpose Hardware for Attacking Cryptograhic Systems)*. Lausanne, Switserland: Springer, 2009. p. 105–116.

BERNSTEIN, D. et al. *Classic McEliece: conservative code-based cryptography*. 2017. NIST post-quantum standardization submission. <https://classic.mceliece.org>.

BERNSTEIN, D. et al. *NTRU Prime*. 2017. NIST Post-Quantum Cryptography Standardization (STANDARDS; (NIST), 2017). <https://ntruprime.cr.yp.to>. Accessed on 2019-01-07.

BERNSTEIN, D. et al. High-speed high-security signatures. *Journal of Cryptographic Engineering*, vol. 2, no. 2, p. 77–89, Sep 2012. ISSN 2190-8516.

BERNSTEIN, D. J. et al. Twisted edwards curves. In VAUDENAY, S. (Ed.). *Progress in Cryptology – AFRICACRYPT 2008*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 389–405. ISBN 978-3-540-68164-9.

BERNSTEIN, D. J.; BUCHMANN, J.; DAHMEN, E. *Post-Quantum Cryptography*. Heidelberg, Deutschland: Springer, 2008.

BERNSTEIN, D. J. et al. High-speed high-security signatures. *Journal of Cryptographic Engineering*, vol. 2, no. 2, p. 77–89, Sep 2012. ISSN 2190-8516. Available from Internet: <https://doi.org/10.1007/s13389-012-0027-1>.

BERNSTEIN, D. J.; LANGE, T. *Post-quantum cryptography—dealing with the fallout of physics success*. 2017. Cryptology ePrint Archive, Report 2017/314. <https://eprint.iacr.org/2017/314>.

BINDEL, N. et al. Transitioning to a quantum-resistant public key infrastructure. In *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings*. Utrecht, the Netherlands: Springer, 2017. p. 384–405.

BOGDANOV, A. et al. Present: An ultra-lightweight block cipher. In PAILLIER, P.; VERBAUWHEDE, I. (Ed.). *Cryptographic Hardware and Embedded Systems - CHES 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 450–466.

BROWN, D.; GALLANT, R.; VANSTONE, S. Provably secure implicit certificate schemes. In *Financial Cryptography*. Berlin, Heidelberg: Springer, 2002. p. 156–165. ISBN 978-3-540-46088-6.

CAMP. *Security Credential Management System Proof-of-Concept Implementation – EE Requirements and Specifications Supporting SCMS Software Release 1.1*. USA: Intelligent Transportation Systems (ITS) – US Department of Transportation, 2016.

CAMP. *PoC Certificate Expiration Timelines*. 2017. CAMP Wiki. <https://wiki.campllc.org/display/SCP/PoC+Certificate+Expiration+Timelines>.

CERTICOM. *SEC 4 v1.0: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV)*. Canada, 2013. Available from Internet: <http://www.secg.org/sec4-1.0.pdf>.

CHAUM, D. Blind signatures for untraceable payments. In SPRINGER. *Advances in cryptology (CRYPTO'82)*. Santa Barbara, USA, 1983. p. 199–203.

CHAUM, D.; HEYST, E. van. Group signatures. In DAVIES, D. W. (Ed.). *Advances in Cryptology — EUROCRYPT '91*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991. p. 257–265. ISBN 978-3-540-46416-7.

CHEN, L. et al. *Report on Post-Quantum Cryptography (NIST IR 8105 Draft)*. Gaithersburg (MD), USA, 2016. Available from Internet: <http://csrc.nist.gov/publications/drafts/nistir-8105/nistir_8105_draft.pdf>.

COOPER, D. et al. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. IETF, 2008. RFC 5280 (Proposed Standard). (Request for Comments, 5280). Available from Internet: <http://www.ietf.org/rfc/rfc5280.txt>.

CORON, J.-S. et al. Universal padding schemes for RSA. In *Advances in Cryptology (CRYPTO'02)*. London, UK, UK: Springer, 2002. p. 226–241. ISBN 3-540-44050-X. Available from Internet: <http://dl.acm.org/citation.cfm?id=646767.704297>.

DAĞDELEN, Ö. et al. High-speed signatures from standard lattices. In ARANHA, D. F.; MENEZES, A. (Ed.). *International Conference on Cryptology and Information Security in Latin America – Latincrypt 2014*. Florianópolis, Brazil: Springer, 2014. (Lecture Notes in Computer Science, vol. 8895), p. 84–103.

DIERKS, T.; RESCORLA, E. *The Transport Layer Security (TLS) Protocol Version 1.2*. IETF, 2008. RFC 5246 (Proposed Standard). (Request for Comments, 5246). Available from Internet: <http://www.ietf.org/rfc/rfc5246.txt>.

DING, J.; SCHMIDT, D. Rainbow, a new multivariable polynomial signature scheme. In *International Conference on Applied Cryptography and Network Security – ACNS 2005*. New York, NY, USA: Springer, 2005. (Lecture Notes in Computer Science, vol. 3531), p. 164–175. ISBN 3-540-26223-7, 978-3-540-26223-7.

DODS, C.; SMART, N.; STAM, M. Hash based digital signature schemes. In *Cryptography and Coding*. Cirencester, UK: Springer, 2005. (Lecture Notes in Computer Science, vol. 3796), p. 96–115.

DOUCEUR, J. The Sybil attack. In *Proc. of 1st International Workshop on Peer-to-Peer Systems (IPTPS)*. Springer, 2002. Available from Internet: <https://www.microsoft.com/en-us/research/publication/the-sybil-attack/>.

DUCAS, L. et al. Lattice signatures and bimodal Gaussians. In CANETTI, R.; GARAY, J. A. (Ed.). *Advances in Cryptology – CRYPTO 2013*. Santa Barbara (CA), USA: Springer, 2013. (Lecture Notes in Computer Science, vol. 8042), p. 40–56.

DWARAKANATH, N. C.; GALBRAITH, S. D. Sampling from discrete gaussians for lattice-based cryptography on a constrained device. *Applicable Algebra in Engineering, Communication and Computing*, vol. 25, no. 3, p. 159–180, Jun 2014.

EDWARDS, H. A normal form for elliptic curves. *Bulletin of the American Mathematical Society*, vol. 44, no. 3, p. 393–422, 2007.

FIGUEIREDO, L. et al. Towards the development of intelligent transportation systems. In *Proceedings of the IEEE Intelligent Transportation Systems (ITSC'2001)*. CA, USA: IEEE, 2001. p. 1206–1211.

FÖRSTER, D.; KARGL, F.; LÖHR, H. PUCA: A pseudonym scheme with user-controlled anonymity for vehicular ad-hoc networks (VANET). In *VNC*. Germany: IEEE, 2014. p. 25–32.

FUJII, H.; ARANHA, D. F. Curve25519 for the cortex-m4 and beyond. In . Havana, Cuba: Springer New York, 2017. Available from Internet: <http://www.cs.haifa.ac.il/~orrd/LC17/paper39.pdf>.

GALBRAITH, S. D.; PETIT, C.; SILVA, J. Identification protocols and signature schemes based on supersingular isogeny problems. In TAKAGI, T.; PEYRIN, T. (Ed.). *Advances in Cryptology – ASIACRYPT 2017*. Cham: Springer International Publishing, 2017. p. 3–33. ISBN 978-3-319-70694-8.

GENTRY, C.; PEIKERT, C.; VAIKUNTANATHAN, V. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th annual ACM symposium on Theory of computing – STOC '08*. New York, NY,USA: ACM, 2008. (Lecture Notes in Computer Science), p. 197–206. ISBN 978-1-60558-047-0. Available from Internet: <http://doi.acm.org/10.1145/1374376.1374407>.

GOLDREICH, O.; GOLDWASSER, S.; HALEVI, S. Public-key cryptosystems from lattice reduction problems. In *Advances in Cryptology – CRYPTO '97*. CA, USA: Springer, 1997. (Lecture Notes in Computer Science, vol. 1294), p. 112–131.

GUERON, S.; SCHLIEKER, F. Optimized implementation of ring-TESLA. Available online. 2016. Available from Internet: <https://github.com/fschlieker/ring-TESLA>.

HARDING, J. et al. *Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application*. Washington, DC, USA, 2014.

HOUSLEY, R. *Cryptographic Message Syntax (CMS)*. IETF, 2009. RFC 5652 (INTERNET STANDARD). (Request for Comments, 5652). Available from Internet: <http://www.ietf.org/rfc/rfc5652.txt>.

HUSEMÖLLER, D. Elementary properties of the chord-tangent group law on a cubic curve. In *Elliptic Curves*. New York, USA: Springer, 2004. vol. 111, p. 23–43. Available from Internet: <http://dx.doi.org/10.1007/0-387-21577-8_2>.

IEEE. *IEEE Standard Specifications for Public-Key Cryptography – Amendment 1: Additional Techniques*. New York, 2004.

JAO, D.; FEO, L. D. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In YANG, B.-Y. (Ed.). *Post-Quantum Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 19–34.

JAO, D.; SOUKHAREV, V. Isogeny-based quantum-resistant undeniable signatures. In MOSCA, M. (Ed.). *Post-Quantum Cryptography*. Cham: Springer International Publishing, 2014. p. 160–179. ISBN 978-3-319-11659-4.

KHODAEI, M.; PAPADIMITRATOS, P. The key to intelligent transportation: Identity and credential management in vehicular communication systems. *IEEE Vehicular Technology Magazine*, vol. 10, no. 4, p. 63–69, Dec 2015. ISSN 1556-6072.

KIPNIS, A.; PATARIN, J.; GOUBIN, L. Unbalanced oil and vinegar signature schemes. In STERN, J. (Ed.). *Advances in Cryptology – EUROCRYPT '99*. Prague, Czech Republic: Springer, 1999, (Lecture Notes in Computer Science, vol. 1592). p. 206–222.

KNUTH, D. E. *The Art of Computer Programming*. 2ns. ed. United States: Addison-Wesley, 1997. vol. 3: Sorting and Searching. 111 p.

KOBLITZ, N. Elliptic Curve Cryptosystems. *Mathematics of Computation*, vol. 48, no. 177, p. 203–209, 1987. Available from Internet: <http://www.jstor.org/stable/2007884>.

KUMAR, V.; PETIT, J.; WHYTE, W. Binary hash tree based certificate access management for connected vehicles. In *Proc. of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. Boston, Massachusetts: ACM, 2017. (WiSec'17), p. 145–155. ISBN 978-1-4503-5084-6.

LAMPORT, L. Constructing digital signatures from a one way function. In *SRI International*. Califórnia, EUA: CSL-98, 1979.

LAUTER, K. E.; STANGE, K. E. The elliptic curve discrete logarithm problem and equivalent hard problems for elliptic divisibility sequences. In SPRINGER. *Selected Areas in Cryptography (SAC'08)*. New Brunswick, Canada, 2008. p. 309–327.

LENSTRA, A.; LENSTRA H.W., J.; LOVÁSZ, L. Factoring polynomials with rational coefficients. *Mathematische Annalen*, Springer-Verlag, vol. 261, no. 4, p. 515–534, 1982. ISSN 0025-5831. Available from Internet: <http://dx.doi.org/10.1007/BF01457454>.

LENSTRA, H. W. Integer programming in a fixed number of variables. *Math. Oper. Res.*, vol. 8, p. 538–548, 1983.

LYUBASHEVSKY, V. Lattice signatures without trapdoors. In POINTCHEVAL, D.; JOHANSSON, T. (Ed.). *Advances in Cryptology – Eurocrypt 2012*. Cambridge, UK: Springer, 2012. (Lecture Notes in Computer Science, vol. 7237), p. 738–755.

LYUBASHEVSKY, V.; PEIKERT, C.; REGEV, O. On ideal lattices and learning with errors over rings. In GILBERT, H. (Ed.). French Riviera: Springer, 2010. vol. 6110/2010, no. 015848, p. 1–23.

MCELIECE, R. *A Public-Key Cryptosystem Based On Algebraic Coding Theory*. 1978. 114–116 p. The Deep Space Network Progress Report, DSN PR 42–44. Available from Internet: <http://ipnpr.jpl.nasa.gov/progressreport2/42-44/44N.pdf.>

MERKLE, R. C. *Secrecy, Authentication, and Public Key Systems*. United States: Stanford Ph.D. thesis, 1979.

MICCIANCIO, D.; PEIKERT, C. Trapdoors for lattices: Simpler, tighter, faster, smaller. In POINTCHEVAL, D.; JOHANSSON, T. (Ed.). *Advances in Cryptology – Eurocrypt 2012*. Cambridge, UK: Springer Berlin Heidelberg, 2012, (Lecture Notes in Computer Science, vol. 7237). p. 700–718.

MILLER, V. S. Use of elliptic curves in cryptography. In *Lecture notes in computer sciences; 218 on Advances in cryptology – Crypto'85*. Santa Barbara, California, United States: Springer-Verlag New York, Inc., 1986. p. 417–426. ISBN 0-387-16463-4. Available from Internet: <http://portal.acm.org/citation.cfm?id=18262.25413>.

MOALLA, R. et al. Risk analysis study of ITS communication architecture. In *3rd International Conference on The Network of the Future*. Gammarth, Tunisia: IEEE Computer Society, 2012. p. 2036–2040.

NGUYEN, P.; STERN, J. The two faces of lattices in cryptology. *Cryptography and Lattices*, Springer, p. 146–180, 2001.

NIEDERREITER, H. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, vol. 15, no. 2, p. 159–166, 1986.

NIST. *Federal Information Processing Standard (FIPS 197) – Advanced Encryption Standard (AES)*. National Institute of Standards and Technology, U.S. Department of Commerce. Gaithersburg, MD, USA, 2001. Available from Internet: <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

NIST. *FIPS 186-4 – Digital Signature Standard (DSS)*. Gaithersburg, USA, 2013. Available from Internet: <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>.

NIST. *Federal Information Processing Standard (FIPS 180-4) – Secure Hash Standard (SHS)*. National Institute of Standards and Technology, U.S. Department of Commerce (NIST). Gaithersburg, MD, USA, 2015. DOI:10.6028/NIST.FIPS.180-4.

NIST. *Federal Information Processing Standard (FIPS 202) – SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. National Institute of Standards and Technology, U.S. Department of Commerce. Gaithersburg, MD, USA, 2015. DOI:10.6028/NIST.FIPS.202.

PAPADIMITRATOS, P. et al. Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE Communications Magazine*, vol. 47, no. 11, p. 84–95, November 2009. ISSN 0163-6804.

PATARIN, J.; GOUBIN, L. Trapdoor one-way permutations and multivariate polynomials. In *ICICS'97*. Beijing, China: Springer, 1997. (Lecture Notes in Computer Science, vol. 1334), p. 356–368.

PEIKERT, C. An efficient and parallel Gaussian sampler for lattices. In *Advances in Cryptology – CRYPTO 2010*. Santa Barbara (CA): Springer, 2010. (Lecture Notes in Computer Science, vol. 6223), p. 80–97.

PETIT, J. et al. Pseudonym schemes in vehicular networks: A survey. *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, p. 228–255, 2015. ISSN 1553-877X.

RAMSDELL, B.; TURNER, S. *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification*. IETF, 2010. RFC 5751 (Proposed Standard). (Request for Comments, 5751). Available from Internet: <http://www.ietf.org/rfc/rfc5751.txt>.

REGEV, O. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, ACM, New York, NY, USA, vol. 56, no. 6, p. 34:1–34:40, Sep. 2009. ISSN 0004-5411. Available from Internet: <http://doi.acm.org/10.1145/1568318.1568324>.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, vol. 21, p. 120–126, 1978.

RÜCKERT, M. Lattice-based blind signatures. In *Advances in Cryptology – Asiacrypt 2010*. Singapore: Springer, 2010. (Lecture Notes in Computer Science, vol. 6477), p. 413–430.

SCHAUB, F.; MA, Z.; KARGL, F. Privacy requirements in vehicular communication systems. In *Proc. of the International Conference on Computational Science and Engineering*. Vancouver, Canada: IEEE, 2009. vol. 3, p. 139–145.

SCHNORR, C. P. Efficient identification and signatures for smart cards. In BRASSARD, G. (Ed.). *Advances in Cryptology — CRYPTO' 89 Proceedings*. New York, NY: Springer New York, 1990. p. 239–252. ISBN 978-0-387-34805-6.

SCHNORR, C. P. Efficient signature generation by smart cards. *Journal of Cryptology*, vol. 4, no. 3, p. 161–174, Jan 1991. ISSN 1432-1378. Available from Internet: <https://doi.org/10.1007/BF00196725>.

SHOR, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, vol. 26, p. 1484–1509, 1997.

SILVERMAN, J. H. *The Arithmetic of Elliptic Curves*. Berlin, Germany: Springer, 1986. (Graduate Texts in Mathematics, 106).

SIMPLICIO, M. A. et al. The unified butterfly effect: Efficient security credential management system for vehicular communications. In *2018 IEEE Vehicular Networking Conference (VNC)*. Taipei, Taiwan: IEEE, 2018. p. 1–8. ISSN 2157-9865. See also <https://eprint.iacr.org/2018/089>.

SMART, N. P. The exact security of ecies in the generic group model. In HONARY, B. (Ed.). *Cryptography and Coding*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 73–84. ISBN 978-3-540-45325-3.

STANDARDS, T. N. I. of; (NIST), T. *Post-Quantum Cryptography Standardization*. 2017. <https://csrc.nist.gov/projects/post-quantum-cryptography>. Accessed on 2019-01-07.

STATISTA. *U.S. car sales from 1951 to 2017 (in units)*. 2018. <www.statista.com/statistics/199974/us-car-sales-since-1951/>.

The National Institute of Standards and Technology (NIST). *Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process*. December, 2016. Available from Internet: <https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf>.

VERHEUL, E. *Activate Later Certificates for V2X – Combining ITS efficiency with privacy*. 2016. Cryptology ePrint Archive, Report 2016/1158. Available from Internet: <http://eprint.iacr.org/2016/1158>.

WHYTE, W. et al. A security credential management system for V2V communications. In *IEEE Vehicular Networking Conference*. Boston, USA: IEEE, 2013. p. 1–8. ISSN 2157-9857.