Drive (Quantum) Safe! – Towards Post-Quantum Security for V2V Communications *

Nina Bindel[1], Sarah McCarthy[2], Geoff Twardokus[3], and Hanif Rahbari[3]

[1]Technische Universität Darmstadt
[2]Institute for Quantum Computing and University of Waterloo
[3]Rochester Institute of Technology

April 20, 2022

Abstract

We tackle a challenging problem at the intersection of two emerging technologies: Post-quantum cryptography (PQC) and vehicle-to-vehicle (V2V) communications. Connected vehicles use V2V technology to exchange safety messages that allow them to increase proximity awareness, improving roadway safety and efficiency. The integrity and authenticity of these messages is critical to prevent an adversary from abusing V2V technology to cause a collision, traffic jam, or other unsafe and/or disruptive situations. The IEEE 1609.2 standard (2016) specifies authentication mechanisms for V2V communications that rely on the elliptic curve digital signature algorithm (ECDSA) and are therefore not secure against quantum attackers. In this paper, we are the first to devise and evaluate PQC for authenticating messages in IEEE 1609.2. By analyzing the properties of the NIST PQC standardization finalists, as well as XMSS (RFC 8391), we propose three practical, ECDSA-PQ hybrid designs for use during the transition from classical to PQ-secure cryptography.

*N.B. was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery grant RGPIN-2016-05146, NSERC Discovery Accelerator Supplement grant RGPIN-2016-05146, and Contract 2L 165-180499/001/sv, "PQC Analysis", funded by Public Works and Government Services Canada.

S.M was supported by Canada's NSERC Alliance Program and Public Works Government Services Canada (PWGSC).

This work was supported by the University of Waterloo Institute for Quantum Computing; IQC is supported in part by the Government of Canada through Innovation, Science and Economic Development Canada (ISED) and the Province of Ontario.

Contents

1 Introduction **3**

2 Background **4**
2.1 V2V Communication for Proximity Awareness 4
2.2 V2V Standards 4
2.2.1 V2V security 4
2.2.2 V2V communication technologies 5
2.3 Post-Quantum Signature Algorithms 5

3 Definitions and Threat Model **6**

4 Pure PQ Design **7**
4.1 Proposed Pure PQ Design 8
4.2 Viability of PQ Signature Schemes 9

5 Classical-PQ Hybrid Designs **9**
5.1 Certificate Fragmentation 10
5.2 True Hybrid Design 10
5.3 Backwards-compatible Hybrid Design 11
5.4 Partially PQ Design 12
5.5 Discussion 14

6 Related Work **15**

7 Conclusion and Future Work **16**

A Definitions of Cryptographic Primitives **19**

1 Introduction

Vehicle-to-Everything (V2X) services are among the key enablers of emerging connected technologies such as autonomous and remote driving. With the potential to reduce roadway collisions by 94% [1], Vehicle-to-Vehicle (V2V) communication is a central part of V2X. V2V allows vehicles to exchange safety-related messages containing information such as velocity and direction of travel. Thousands of vehicles on the road today are already equipped with V2V modules [2] and its adoption rate is expected to grow as 3GPP and IEEE envision more use cases of V2X (e.g., sensor data sharing and 3D mapping) [3, 4].

Due to consumer expectations for the longevity of vehicles in combination with the generally low response rate to recalls [5], new vehicles must remain secure for at least 12–15 years after they hit the roads [6]. By then, large quantum computers threatening the security of current ("classical") cryptography might exist. According to a recent study [7], quantum experts expect quantum computers large enough to break classical cryptography may be built within the next 15–30 years with a likelihood of 50%. Given that the security of V2V is currently based on classical (elliptic curve) cryptography [8], safety-critical communications of tens of millions of vehicles will be at a critical risk of a quantum threat whereby malicious V2V messages could put lives at risk. This quantum threat is particularly dangerous because the hardware security modules of vehicles on the road cannot be "purged", updated over-the-air, or successfully recalled if their security is compromised by future quantum attacks; in this sense, they lack *crypto agility* [9]. In fact, the current V2V security standard is developed *explicitly* based on assumptions that become violated once quantum computers of sufficient scale are developed [10]. Retroactively adopting and using Post-Quantum (PQ) cryptography (to fend off quantum attacks) is therefore extremely challenging, even more so because there is not yet a standard PQ algorithm that can be considered for future amendments to V2V standards.

It is therefore critical that future V2V standards employ backwards-compatible, crypto-agile PQ-secure designs that enhance their security against quantum attacks whilst maintaining the current classical security guarantees for interoperability with existing vehicles (and in case current PQ candidates turn out to be insecure). In fact, most standardization agencies, including the National Institute of Standards and Technology (NIST), recommend transitioning to PQ security using hybrid algorithms [11]; that is, combining classical and PQ algorithms in one design [12]. Hybrid PQ-classical designs have been recently explored in [13] and [14] for more delay-tolerant and less restrictive Transport Layer Security (TLS) and Secure Shell (SSH) network protocols. Developing efficient hybrid designs is intrinsically challenging due to the large key/signature sizes of current PQ schemes.

Different from TLS/SSH, secure V2V communication for safety applications must further satisfy a stringent latency requirement. On the transmitter side, safety messages must be sent both quickly and frequently (in terms of milliseconds), restricting both signing time and frame size. On the receiver side, incoming messages must be processed and verified within a few milliseconds of arrival, restricting signature verification time. These requirements do not allow the messages, including signatures or certificates, to exceed a strict size limit.

Therefore, designing a hybrid security solution for V2V communication is particularly challenging since it should not only (1) support general PQ solutions that can adapt to future standards and (2) be backwards-compatible to support the classical mechanisms of legacy vehicles that will continue to be incapable of supporting Post-Quantum Cryptography (PQC) for at least the next several years, through the relatively long "transition period" to purely PQC for V2V, but also (3) be delay-sensitive when generating and verifying safety-critical messages, and more specifically (4) comply with strict V2V protocol constraints (e.g., on frame size and periodicity) for both Cellular Vehicle-to-Everything (C-V2X) and IEEE 802.11-based V2V technologies. Incorporating PQC–with its excessively large key sizes or lengthy digital signatures–is a nontrivial and challenging problem that was not anticipated when V2V security standards were initially developed to be flexible.

The IEEE 1609.2 [8] standard, with its subsequent amendments [15–17], standardizes how current vehicles can securely transmit V2V messages irrespective of the underlying technology. It defines (among other things) digital signature schemes based on elliptic curve cryptography, as well as the structure of V2V certificates. In this paper, we focus on the authentication mechanisms defined in this standard, with the primary objective of replacing (or augmenting) them with general PQ alternatives. We take the first steps toward filling the gap between the current, quantum-vulnerable cryptography used in IEEE 1609.2 and future, PQ-secure V2V communication (PQ-V2V) by proposing practical, PQ alternatives for message authentication in V2V during

the transition period.

Specifically, we propose hybrid designs that weave together classical and PQ signatures in a way that guarantees PQ security as long as classical signatures cannot be forged in *less than a week*, even by quantum attackers. Our careful analysis of the quantum threat to V2V yields more efficient and secure hybrid designs than blindly substituting classical with PQ or existing hybrid schemes. While PQC has been examined for intra-vehicle wired communications (e.g., between electronic control units [18]), to the best of our knowledge, we are the first to undertake the challenge of devising and evaluating a PQ design for authenticating safety messages exchanged in a connected V2X environment with V2V wireless communications secured by 1609.2.

Contributions— The key finding of this paper is that PQC, despite its apparent incompatibility with safety-related V2V communication (due to large signatures and keys), can in fact be securely integrated with IEEE 1609.2 and technologies for a transition period. However, this requires the PQ instantiation and its parameters to be chosen carefully.

Our main contribution is that of *Hybrid Designs for PQ-V2V Authentication.* We analyze the suitability of seven PQ signature algorithms (six NIST candidates and the promising eXtended Merkle Signature Scheme (XMSS) [19]) for use in V2V. We show that simply replacing ECDSA with PQC is not viable due to V2V constraints. To overcome this, we devise and instantiate three practical, 1609.2-compatible ECDSA-PQ hybrid designs—*true*, *backwards-compatible*, and *partially PQ*—for use in the PQ transition. While our true hybrid design gives the strongest security guarantee, it is also the least flexible, motivating our backwards-compatible design; however, aiming for greater efficiency, we propose our partially PQ hybrid design.

We would also like to note that our findings in this work highlight important considerations about the practicality of several PQ algorithms, especially those that may be considered in the standardization and design of PQ technologies. These findings are especially relevant to NIST's intention to standardize PQ algorithms based on practicality and suitability analyses of Round 3 candidates for real-life applications (e.g., V2V communication), and we show that some of those candidates are unsuitable for use in V2V.

Paper Organization After preliminaries in Section 2, we describe our threat model and formally define V2V protocols in Section 3. Section 4 discusses the challenges of naively adopting a purely PQ design. We present our hybrid designs and discuss their merits and drawbacks in Section 5. We conclude with related works and future directions in Sections 6 and 7, respectively.

2 Background

2.1 V2V Communication for Proximity Awareness

To achieve the safety benefits of V2V, every vehicle under direct V2V communication mode periodically broadcasts a digitally-signed basic safety message (BSM) a minimum of once every $100\,ms$ (i.e., $10\,\text{Hz}$). Each BSM contains motion and position information to allow other vehicles to coordinate their movements to avoid collisions, especially when vehicle sensors or drivers are unable to see or detect other vehicles.

Each BSM is signed and packed into a Secure Protocol Data Unit (SPDU) along with security information needed for verification, see Figure 1. The SPDU is broadcast within a frame using one of two major technologies: Dedicated Short Range Communications (DSRC) [20], based on the IEEE 802.11 protocol tailored for the decentralized, high-mobility V2V environment, or C-V2X [21, 22], a similarly decentralized protocol based on LTE or 5G cellular technology. *Each technology defines a tight upper bound on the frame size.*

2.2 V2V Standards

2.2.1 V2V security

Security requirements and services for both DSRC and C-V2X are defined in IEEE 1609.2-2016 [8] and IEEE 1609.2.1-2020 [17]. Among other elements, 1609.2 specifies asymmetric cryptographic mechanisms and algorithms to securely exchange BSMs, while 1609.2.1 specifies certificate management requirements for vehicles. Of particular concern to us, it is specified that Elliptic Curve Digital Signature Algorithm (ECDSA) is used to generate all signatures with either the *NIST P-256* (Federal Information Processing Standards

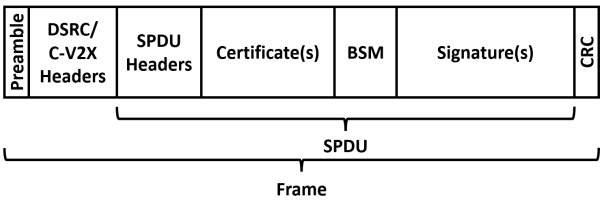

Figure 1: Relationship between frame, SPDU, and BSM.

(FIPS) 186-4 [23]) or *brainpoolP256r1* [24] elliptic curves. Either option gives 128 bits of security and results in the key and signature sizes shown in Table 1.

Beyond signatures, IEEE 1609.2 allows certificates, explicit or implicit, as pseudonym certificates to protect the integrity of the public (verification) key included in broadcast SPDUs; pseudonym certificates refresh periodically every five minutes[1], and they are valid for at most one week [10]. In *explicit* certificates, the verification key is explicitly given and signed by the issuer (also called the Certificate Authority (CA)). In contrast, *implicit* certificates include a reconstruction value used to reconstruct the verification key using the issuer's public key (provided by a root certificate—a self-signed, trusted CA certificate). Implicit pseudonym certificates are generated using the Elliptic Curve Qu-Vanstone (ECQV) scheme [8,25]. They allow for smaller messages, as the reconstruction value is smaller than the combined verification key and issuer signature used by explicit certificates. In the current 1609.2 ECDSA-based standard, the size of one SPDU is at most 226 bytes using implicit or 248 bytes using explicit certificates. This is well below the maximally allowed frame size (discussed later). Although implicit certificates are smaller, a practical design for PQ implicit certificates has not yet been devised. Therefore, in our PQ designs, explicit certificates must be used instead.

In addition to certificates, 1609.2 also defines a peer-to-peer certificate distribution (P2PCD) protocol to allow sharing knowledge of certificates with other vehicles upon request, reducing the time needed to verify a pseudonym certificate. However, analyzing any possible delay incurred by P2PCD is beyond the scope of this work. We also note that under current industry standards (e.g., [26]), a vehicle will only include its full (explicit) pseudonym certificate in every fifth SPDU; in the other transmissions (80%), only a hash value of the certificate is included. This defines what we call a *five-message cycle* and is important to establish as an important factor in our designs.

2.2.2 V2V communication technologies

C-V2X (based on 3GPP specifications for LTE and 5G) and DSRC are two physical- and MAC-layer communication protocols for V2V; within the context of BSMs, however, *neither protocol provides any security requirements*. Instead, both DSRC and C-V2X adopt the IEEE 1609.2 standard for security. As we are concerned primarily with security at the SPDU level (which is above the MAC layer) rather than a communications level, our PQ designs are transparent to both protocols as long as it satisfies their constraints, specifically, maximum allowed frame size. That maximum is $2,304$ and $5,861$ bytes in DSRC and C-V2X, respectively. By designing hybrid PQ solutions to fit the requirements of IEEE 1609.2, we are able to abstract from the particulars of any communication technology. We consider DSRC in the rest of this paper for evaluation purposes as it requires the tighter constraint. We calculate the sizes of SPDUs and frames as follows. Let $|\mathsf{pk}|$ and $|\mathsf{sig}|$ denote the public key and signature sizes of a given signature scheme, respectively (see Table 1). We compute the explicit certificate size as $36 + |\mathsf{pk}| + |\mathsf{sig}|$, SPDU size as $|\text{SPDU}| = 24 + |\mathsf{BSM}| + |\text{explicit cert}| + |\mathsf{sig}|$, and then the total MAC layer frame size as $|F| = 36 + |\text{SPDU}|$.

2.3 Post-Quantum Signature Algorithms

We consider all six candidate PQ signature algorithms from the third round of NIST's PQ standardization process, as well as the recent *XMSS* [19] scheme for use in our PQ-V2V designs. While both XMSS and and Leighton-Micali Hash-Based Signature Scheme (LMS) [27] are approved in the *FIPS PUB 186-5 Draft* [28], we use the XMSS reference implementation from RFC 8391 [19] as it leads to slightly smaller keys and signatures [29], while the performance benefit of LMS is not sufficient to make it more viable than XMSS.

[1]IEEE 1609.2 does not describe how often pseudonyms are refreshed, but five minutes is a common reference point [10].

Table 1: The maximum cryptographic element sizes (in bytes) for considered signature schemes and their viability for each proposed design; $|F|$ gives the total frame size (using explicit/implicit ECDSA certificates).

Algorithm	$\|\mathsf{pk}\|$	$\|\mathsf{sig}\|$	$\|F\|$	True&BC	Partial PQ	Pure PQ
ECDSA	64	64	292/226	-	-	-
XMSS	68	2,692	5,552/-	No ($\|\mathsf{sig}\|$)	**Yes**	No ($\|\mathsf{sig}\|$)
Dilithium-II	1,312	2,420	6,252/-	No ($\|\mathsf{sig}\|$)	**Yes**	No ($\|\mathsf{sig}\|$)
Falcon-512	897	690	2,329/-	**Yes**	**Yes**	(**Yes**)
Rainbow-I	161,600	66	161,892/-	No ($\|\mathsf{pk}\|$)	No ($\|\mathsf{pk}\|$)	No ($\|\mathsf{pk}\|$)
GeMSS-128	352,188	32	352,412/-	No ($\|\mathsf{pk}\|$)	No ($\|\mathsf{pk}\|$)	No ($\|\mathsf{pk}\|$)
Picnic-L1-FS	33	34,036	68,196/-	No ($\|\mathsf{sig}\|$)	No ($\|\mathsf{sig}\|$)	No ($\|\mathsf{sig}\|$)
Sphincs$^+$-128s	32	7,856	15,844/-	No ($\|\mathsf{sig}\|$)	**Yes**	No ($\|\mathsf{sig}\|$)

For each algorithm, we chose an instantiation yielding a minimum of 128 bits of security from the literature. We selected the following instantiations for the three NIST finalists: *Falcon-512* [30], *Dilithium-II* [31], and *Rainbow-I* [32] as well as the following instantiations of the alternate candidates: *Picnic-L1-FS* [33], GrEat Multivariate Short Signature (GeMSS) instantiation *GeMSS-128* [34], and *Sphincs*$^+$*-128s* [35]. Further, we chose *XMSS-SHA2_16_256* as our instantiation of the stateful XMSS scheme using Secure Hash Algorithm 2 (SHA2), as the signature must be unforgeable for at least $3,000$ signatures (translating to using the key for about five minutes and generating 10 sign/sec). Table 1 provides the sizes of the public keys and signatures of each algorithm and a summary of whether they are suitable for each of our designs. Where an algorithm is deemed unsuitable for use in V2V due to the size of its public key or signature, this is indicated by, e.g., "No ($|\mathsf{sig}|$)" for when an algorithm's signature length excluded it from consideration for certain designs.

3 Definitions and Threat Model

Threat model—IEEE 1609.2 [36, Introduction] defines its security goal as "to protect messages from attacks such as eavesdropping, spoofing, alteration, and replay." Hence, we assume the attacker's goal is to make receivers accept fraudulent BSMs by launching the attacks mentioned above to cause traffic delays or collisions, among other disruptive events. We assume the following capabilities of the attacker:

- observe, drop, delay, replay legitimately generated and broadcast SPDUs,
- alter SPDUs, e.g., changing the BSM, changing/dropping/adding/swapping the/a pseudonym certificate,
- trigger BSM transmissions that are then legitimately signed and broadcast by the targeted sender, and
- is not allowed to acquire more than one pseudonym certificate for pseudonyms from CA.

However, we assume that all computations (including storage of secret values) are secure, i.e., no side-channel or fault attacks can occur. Moreover, we assume that the certificate registration and verification is correct and secure. In particular, we assume that CAs are honest, the root certificate cannot be forged, only legitimate pseudonyms can be registered, and invalid certificates are detected. It is important to emphasize that we assume that pseudonym certificates can be verified *immediately*. Furthermore, we assume that *all honestly generated* SPDUs are received by the verifier in the *same order* that they have been sent by the signer as long as the receiver is in the transmission range. Moreover, we assume that communication errors during the transmission are handled by lower layers.

Next we define V2V protocols formally based on *certified signature schemes*—a combination of signature schemes and certificates formalized in [37]. We recall their original definition in Appendix A, and give our adapted definition for V2V protocols that enable instantiations with our hybrid designs.

Definition 1 (V2V protocol)**.** *Let* $\mathcal{S}_i = (\mathsf{KGen}^{(i)}, \mathsf{Sign}^{(i)}, \mathsf{Vrfy}^{(i)})$ *be digital signature schemes (defined in Appendix A) for all* $i \in \mathbb{Z}/5\mathbb{Z}$[2] *Moreover, let* $\mathcal{M}$ *be a message space and* $\mathcal{U}$ *be the set of pseudonyms. A V2V protocol* $\mathcal{P} = (\mathsf{KGen}_C, \mathsf{CGen}(\mathsf{CGen}_C, \mathsf{CGen}_U), \mathsf{SPDUGen}, \mathsf{SPDUVerify})$ *is defined via the following polynomial-time algorithms*

KGen$_C$ *returns a public-secret key pair* $(\mathsf{pk}_C, \mathsf{sk}_C)$.

CGen(CGen$_C$, CGen$_U$) *is an interactive protocol between the user* $U \in \mathcal{U}$ *and the CA* C *following Definition 3 in Appendix A, generating a certificate* C_U *over* (U, pk_U)*. We refrain from describing the registration process in detail as it is not finalized for V2V protocols yet [38]. We write* $\mathsf{CGen}(\mathsf{sk}_C, U, \mathsf{pk}_C)$ *instead of* $\mathsf{CGen}(\mathsf{CGen}_C(\mathsf{sk}_C), \mathsf{CGen}_U(U, \mathsf{pk}_C))$ *for brevity.*

SPDUGen *corresponds to signing a BSM and generating the SPDU. Upon input an identity* $U \in \mathcal{U}$*, a secret key* sk_U*, a certificate* C_U*, the CA's public key* pk_C*, a message* $\mathsf{BSM} \in \mathcal{M}$*, and* $i \in \mathbb{Z}/5\mathbb{Z}$*, it returns an SPDU depending on* i*, including a signature* $\mathsf{sig} \leftarrow \mathsf{Sign}^{(i)}(\mathsf{sk}_U, \mathsf{BSM})$*. Depending on* $\mathcal{S}_i$*,* sig *might be the empty string.*

SPDUVerify *returns 0 or 1, upon input* U*,* pk_U*,* C_U*,* pk_C*, a state* st[3]*, and* spdu*. It outputs* 1 *if* spdu *(including* BSM *and* sig*) is valid, and* 0 *otherwise.*

The spdu *is valid if* $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{spdu}, \mathsf{st}) = 1$*,* $\mathsf{Vrfy}^{(i)}(\mathsf{pk}_U, \mathsf{sig}, \mathsf{BSM}) = 1$*, and* spdu *is of the correct form depending on* i *(which in turns depends on* st*). The function* CVrfy *will not be formally defined here, but it suffices to assume it checks all aspects of certificate validation. Depending on the protocol instantiation this might include verifying the issuer's signature, the certificate chain, the certificate's expiration date, etc. Depending on the internal verification, the state* st *is updated accordingly within* CVrfy*.* $\mathsf{Vrfy}^{(i)}$ *follows that of Definition 3 and the correct form of* spdu *is dependent on the design and position* $i \in \mathbb{Z}/5\mathbb{Z}$ *within the message cycle, as described in Section 5.*

We require $\mathcal{P}$ *to be correct, i.e., for all* $\mathsf{BSM} \in \mathcal{M}$ *and* $U \in \mathcal{U}$*, it should hold that if* $((U, \mathsf{pk}_U, \mathsf{C}_U), (U, \mathsf{pk}_U, \mathsf{sk}_U, \mathsf{C}_U)) \leftarrow \mathsf{CGen}(\mathsf{sk}_C, U, \mathsf{pk}_C)$ *and* $(\mathsf{pk}_C, \mathsf{sk}_C) \leftarrow \mathsf{KGen}_C()$*, then* $\mathsf{SPDUVerify}(U, \mathsf{pk}_U, \mathsf{C}_U, \mathsf{pk}_C, \mathsf{st}, \mathsf{SPDUGen}(U, \mathsf{sk}_U, \mathsf{C}_U, \mathsf{pk}_C, \mathsf{BSM}, i)) = 1$.

In case of the Pure ECDSA design, $\mathcal{S}_1 = ... = \mathcal{S}_5$ as ECDSA is the only signature scheme used. For $i = 1 \mod 5$, spdu_i consists of an BSM, an ECDSA signature, and the pseudonym certificate. All other SPDUs consist of an BSM, an ECDSA signature, and the hash of pseudonym certificate. After receiving spdu_1, the verifier stores the hash of the certificate and the public key in st.

Notation. For the description of V2V designs in the next sections, we fix the following notation. Each design considers one run of the protocol, which re-occurs every five minutes. Moreover, we define $\mathcal{S}_\mathsf{c} = (\mathsf{KGen}^\mathsf{c}, \mathsf{Sign}^\mathsf{c}, \mathsf{Vrfy}^\mathsf{c})$ to be ECDSA and $\mathcal{S}_\mathsf{pq} = (\mathsf{KGen}^\mathsf{pq}, \mathsf{Sign}^\mathsf{pq}, \mathsf{Vrfy}^\mathsf{pq})$ be a PQ scheme. Moreover, for an explicit certificate C, we write cbody for the corresponding certificate body, i.e. the data which is signed by the CA. As per all the described designs, if a signature verification fails, it will be discarded. This is indicated by *Abort* in our pseudo-codes, and follows the reasoning in [39]. Finally, we assume $i \in \mathbb{Z}/5\mathbb{Z}$ and write $[n, m]$ for the integer interval $\{n, ...m\}$.

4 Pure PQ Design

We start by describing a purely PQ V2V design. Due to the large size of public keys and certificates (see Table 2.3 for sizes if we naively substitute ECDSA with each PQ algorithm), the PQ certificate needs to be transmitted before signing BSMs is possible. This leads to a period of unauthenticated BSMs. While this is allowed in IEEE 1609.2 [36, Sect. 4.2.2.2.2], it is less secure than the pure ECDSA-based design, motivating our classical-PQ hybrids presented in Section 5.

[2] We define V2V protocols via 5-SPDU cycles, see Section 2.2.1. It can be easily generalized at the expense of additional notation.

[3] Note the difference in input compared to Vrfy in Definition 3.

1 :	**User** U		**Receiver** R
2 :	**for** $i = 1, ..., \alpha - 1$ **:**		
3 :	$\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, C_i)$	$\xrightarrow{\mathsf{spdu}_i}$	$(\mathsf{BSM}_i, \mathsf{C}_i) \leftarrow \mathsf{spdu}_i$
4 :			$\mathsf{Process}(\mathsf{BSM}_i)$ #w/o verification
5 :	$\mathsf{spdu}_\alpha \leftarrow (\mathsf{BSM}_\alpha, C_\alpha)$	$\xrightarrow{\mathsf{spdu}_\alpha}$	$(\mathsf{BSM}_\alpha, C_\alpha) \leftarrow \mathsf{spdu}_\alpha$
6 :			$\mathsf{Process}(\mathsf{BSM}_\alpha)$ #w/o verification
7 :			$\mathsf{C}_U \leftarrow \mathsf{CCons}(\mathsf{C}_1, ..., \mathsf{C}_\alpha)$
8 :			if $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{C}_U) = 1$:
9 :			$\mathsf{Process}(\mathsf{BSM}_i)$
10 :	**for** $i = \alpha + 1, ..., 5$ **:**		Abort
11 :	$\mathsf{sig}_i \leftarrow \mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_U, \mathsf{BSM}_i)$		
12 :	$\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}_i, h)$	$\xrightarrow{\mathsf{spdu}_i}$	$(\mathsf{BSM}_i, \mathsf{sig}_i, h) \leftarrow \mathsf{spdu}_i$
13 :			if $h == \mathsf{H}(\mathsf{C}_U)$:
14 :			if $\mathsf{Vrfy}^{\mathsf{pq}}(\mathsf{pk}_U, \mathsf{sig}_i, \mathsf{BSM}_i) = 1$:
15 :			$\mathsf{Process}(\mathsf{BSM}_i)$
16 :			Abort

Figure 2: Pseudo-code description of the Pure PQ Design to be repeated every five BSMs; we omit an explicit description of st.

4.1 Proposed Pure PQ Design

To be able to give a general description of a purely PQ design and due to the potential large sizes of PQ certificates, we define a function $\mathsf{CFrag}_\alpha : \mathsf{C} \rightarrow \{\mathsf{C}_1, ..., \mathsf{C}_\alpha\}$ which fragments a given certificate into α equally sized parts[4]. The choice for α is made depending on the specific PQ algorithm used. The inverse $\mathsf{CCons}_\alpha : \{\mathsf{C}_1, ..., \mathsf{C}_\alpha\} \rightarrow \mathsf{C}$ reconstructs a certificate from given fragments. Moreover, let $\{\mathsf{C}_i\}_{i=1,...,\alpha} \leftarrow \mathsf{CFrag}_\alpha(\mathsf{C}_U)$, and $h \leftarrow \mathsf{H}(\mathsf{C}_U)$ for the remainder of this section.

Let $\mathcal{P}$ be the *pure PQ* protocol (following Definition 1) using the signature scheme $\mathcal{S}_{\mathsf{pq}}$ and described as follows. We give a pseudo-code description of $\mathcal{P}$ in Figure 2.

KGen_C returns $(\mathsf{sk}_C, \mathsf{pk}_C) \leftarrow \mathsf{KGen}^{\mathsf{pq}}()$.

CGen generates $\mathsf{C}_U = \mathsf{C}_U^{\mathsf{pq}}$ over (U, pk_U) with $(\mathsf{sk}_U, \mathsf{pk}_U) \in \mathrm{Im}(\mathsf{KGen}_{\mathsf{pq}})$ including $\mathsf{sig} \leftarrow \mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_C, \mathsf{cbody}_U)$.

SPDUGen computes $\mathsf{sig}_i \leftarrow \mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_U, \mathsf{BSM}_i)$, and returns

$$\mathsf{spdu}_i = \begin{cases} (\mathsf{BSM}_i, \mathsf{C}_i), & \text{for } i \in \{1, ..., \alpha\}, \\ (\mathsf{BSM}_i, \mathsf{sig}_i, h), & \text{for } i \in \{\alpha + 1, ..., 5\}. \end{cases}$$

SPDUVerify is defined as follows. For $i \in \{1, ..., \alpha - 1\}$: if spdu_i of correct form, process BSM_i (w/o any verification), update st with C_i, and return 1. For $i = \alpha$: $\mathsf{C}_U \leftarrow \mathsf{CCons}(\mathsf{C}_1, ..., \mathsf{C}_\alpha)$. If spdu_i of correct form and $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{C}_U, \mathsf{st}) = 1$, process BSM_i, update st with $(\mathsf{H}(\mathsf{C}_U), \mathsf{pk}_U)$, and return 1. For $i \in \{\alpha + 1, ..., 5\}$: if $\mathsf{Vrfy}^{\mathsf{pq}}(\mathsf{pk}_U, \mathsf{sig}_i, \mathsf{BSM}_i) = 1 \wedge h = \mathsf{H}(\mathsf{C}_U) \wedge \mathsf{spdu}_i$ of correct form, then process BSM_i and return 1.

An alternative design would be to send the BSM, the PQ signature and certificate segmented into (at least) two frames every time a BSM is sent. However, we consider this to be an impractical approach because doing so would massively decrease the number of total vehicles that the V2V system could support (i.e., every message would be extremely large and take a long time to transmit, so far fewer vehicles could transmit within the required $100\,ms$ interval).

[4]Additionally, the fragmentation could be optimized over the size of the certificate and the first signature, to enable sending the first signature earlier. However, in all our instantiations, it does not make a difference.

4.2 Viability of PQ Signature Schemes

Analysing the viability of PQ signatures as replacement for ECDSA (during certificate and BSM signing) results in two key findings: 1) none of the considered PQ algorithms can be an exact replacement of ECDSA, and 2) using Falcon as the best option leads to unauthenticated BSMs as explained next.

Falcon. Compared with the other PQ schemes, Falcon gives the smallest total frame size, which still exceeds the maximum allowable size of $2,304$ bytes. Hence, not even Falcon satisfies the size restrictions of IEEE 1609.2 and V2V communication protocols. To have BSMs signed as soon as possible, the optimum choice for α in this instance is $\alpha = 1$. This means that for a purely Falcon-based design, we need to send an unauthenticated frame of $F_1 = 1,552$ bytes in order to transmit the certificate. Only the second frame onwards will be authenticated; F_2 onwards has a size of $1,552$, as can been seen in Table 2. The parentheses in Table 1 indicates that Falcon only "works" once the certificate has been transmitted, i.e., from the second SPDU onwards.

Dilithium, Picnic, Sphincs$^+$, and XMSS. Signatures generated by these four schemes are too large to fit into a single frame (see Table 1). We considered fragmenting large signatures similarly to how we have fragmented large certificates. More concretely, suppose we sign BSM_1 with Dilithium, and fragment this signature into two parts. Then we send the first fragment in spdu_1 (together with BSM_1), and the second one in spdu_2 (along with BSM_2). This means 1) the signature over BSM_1 could only be verified after receiving spdu_2, and 2) BSM_2 could not be signed. Hence, less than half of the BSMs could be signed. Another option would be to resend BSM_1 in spdu_2, which would straightaway reduce the channel capacity. Therefore we do not propose this as an instantiation for a pure PQ design. We will see, however, that at least Dilithium, Sphincs$^+$, and XMSS are viable for the Partially PQ design in Section 5.4.

Rainbow and GeMSS. In contrast to Dilithium, a Rainbow-based frame consisting of the BSM and a Rainbow signature would only be 178 bytes large, being smaller than ECDSA-based frames. However, only signature schemes whose certificates can be fragmented into five or less parts can be considered for this design. Hence, Rainbow and GeMSS' extremely large public keys (see Table 1) exclude them as viable instantiations for the purely PQ design.

5 Classical-PQ Hybrid Designs

Designing hybrid schemes is not trivial, and they often make use of concatenation or a nested approach as described in [12], leading to different security–efficiency trade-offs. In this section, we describe our designs that give different degrees of PQ security that satisfy the constraints of IEEE 1609.2 and authenticate *every* BSM at least with ECDSA signstures.

True Hybrid: This design gives the strongest of our security guarantees, but it is not backwards-compatible. It combines ECDSA- and PQ signatures and certificates; more concretely, certificates are a concatenation of an ECDSA/ECQV certificate and a PQ certificate while BSMs are authenticated by concatenations of ECDSA and PQ signatures over the same BSM. To satisfy size constraints, the first few SPDUs sign the BSM using ECDSA and include pieces (*segments*, or *fragments*) of the sender's hybrid certificate. After a few messages, all fragments are received, allowing subsequent messages to be effectively protected with ECDSA-PQ hybrid signatures.

Backwards-compatible Hybrid: This design takes after the *True Hybrid* design but offers backwards-compatibility at the cost of only guaranteeing PQ security for honest users who are able to process both ECDSA and PQ signatures. Legacy vehicles with hardware that cannot support PQ algorithms can still verify the messages using only ECDSA, making this design particularly useful for the transition period as it allows existing vehicles to support our scheme through a simple software update rather than requiring hardware retrofitting with PQ-capable cryptographic processors.

Partially PQ Hybrid: This design has the greatest efficiency and smallest messages, while at the same time offering backwards-compatibility. The intuition behind this design is that although signatures over BSMs are solely generated using ECDSA, the integrity of the ECDSA key used to generate those signatures is guaranteed by both ECDSA and PQ signatures. We consider this an acceptable level

of security because quantum computers are not expected to be capable of forging ECDSA signatures within the short validity period of pseudonym certificates.

5.1 Certificate Fragmentation

Similarly to Section 4.1, we define CFrag_α and CCons_α, but add optimization requirements. To this end, define $\mathsf{C}_U = (\mathsf{C}_U^{\mathsf{c}} || \mathsf{C}_U^{\mathsf{pq}})$ be the hybrid certificate defined differently in every design, with $\mathsf{C}_U^{\mathsf{c}}$ being the classical and $\mathsf{C}_U^{\mathsf{pq}}$ the PQ certificate. The number of fragments α is optimized based on the design and the PQ algorithm used such that 1) all BSMs can be signed at least with ECDSA (i.e., $\mathsf{C}_1 = (\mathsf{C}_U^{\mathsf{c}} || \mathsf{C}_{\mathsf{frac}})$ with $\mathsf{C}_{\mathsf{frac}}$ being some fraction of $\mathsf{C}_U^{\mathsf{pq}}$), 2) α is minimal to ensure PQ security for as many BSMs as possible, 3) all frames are at most $2,304$ bytes, 4) the size of all frames used to transmit C_U is equal, to decrease the likelihood of frame loss due to large frames. For all our designs, let $\{\mathsf{C}_i\}_{i=1,\dots,\alpha} \leftarrow \mathsf{CFrag}_\alpha(\mathsf{C}_U)$, $h \leftarrow \mathsf{H}(\mathsf{C}_U)$, and $h^{\mathsf{c}} \leftarrow \mathsf{H}(\mathsf{C}_U^{\mathsf{c}})$.

5.2 True Hybrid Design

Our first hybrid design uses classical and PQ signatures during the SPDU generation. This ensures that if *at least one* of the signature algorithms is unforgeable, the i-unforgeability of the V2V protocol is guaranteed. Naturally, this additional security guarantee comes at the cost of having to send two certificates and two signatures, increasing frame size.

Description of the True Hybrid Let $\mathcal{P}$ be the true hybrid protocol defined next using the two signature schemes $\mathcal{S}_{\mathsf{c}}$ and $\mathcal{S}_{\mathsf{pq}}$. A pseudo-code description is given in Figure 3.

$\mathsf{KGen}_{\mathcal{C}}$ returns a key pair $(\mathsf{pk}_C, \mathsf{sk}_C)$ with $\mathsf{pk}_C = (\mathsf{pk}_C^{\mathsf{c}} || \mathsf{pk}_C^{\mathsf{pq}})$, $\mathsf{sk}_C = (\mathsf{sk}_C^{\mathsf{c}} || \mathsf{sk}_C^{\mathsf{pq}})$, $(\mathsf{sk}_C^{\mathsf{c}}, \mathsf{pk}_C^{\mathsf{c}}) \leftarrow \mathsf{KGen}^{\mathsf{c}}()$, and $(\mathsf{sk}_C^{\mathsf{pq}}, \mathsf{pk}_C^{\mathsf{pq}}) \leftarrow \mathsf{KGen}^{\mathsf{pq}}()$.

CGen generates $\mathsf{C}_U = (\mathsf{C}_U^{\mathsf{c}} || \mathsf{C}_U^{\mathsf{pq}})$ with $\mathsf{C}_U^{\mathsf{c}}$ over $(U, \mathsf{pk}_U^{\mathsf{c}})$ with $(\mathsf{sk}_U^{\mathsf{pq}}, \mathsf{pk}_U^{\mathsf{pq}}) \in \mathrm{Im}(\mathsf{KGen}_{\mathsf{pq}})$ and $\mathsf{C}_S^{\mathsf{pq}}$ over $(U, \mathsf{pk}_U^{\mathsf{pq}})$ with $(\mathsf{sk}_U^{\mathsf{pq}}, \mathsf{pk}_U^{\mathsf{pq}}) \in \mathrm{Im}(\mathsf{KGen}_{\mathsf{pq}})$.

SPDUGen computes $\mathsf{sig}_i^{\mathsf{pq}} \leftarrow \mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_U^{\mathsf{pq}}, \mathsf{BSM}_i)$, $\mathsf{sig}_i^{\mathsf{c}} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_U^{\mathsf{c}}, \mathsf{BSM}_i)$, and $\mathsf{sig}_i \leftarrow (\mathsf{sig}_i^{\mathsf{pq}} || \mathsf{sig}_i^{\mathsf{c}})$, and returns

$$\mathsf{spdu}_i = \begin{cases} (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{C}_i), & \text{for } i = 1, \\ (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h^{\mathsf{c}}, \mathsf{C}_i), & \text{for } i \in [2, \alpha - 1], \\ (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h, \mathsf{C}_i), & \text{for } i = \alpha, \\ (\mathsf{BSM}_i, \mathsf{sig}_i, h), & \text{for } i \in [\alpha + 1, 5]. \end{cases}$$

SPDUVerify is defined as follows. For $i = 1$: if $\mathsf{CVrfy}(\mathsf{pk}_C^{\mathsf{c}}, \mathsf{C}_U^{\mathsf{c}}, \mathsf{st}) = 1 \wedge \mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1 \wedge \mathsf{spdu}_i$ of correct form, then process BSM_i, update st with $\mathsf{H}(\mathsf{C}_U^{\mathsf{c}})$, C_i, and $\mathsf{pk}_U^{\mathsf{c}}$, and return 1. For $i \in [2, \alpha - 1]$: if $h^{\mathsf{c}} = \mathsf{H}(\mathsf{C}_U^{\mathsf{c}}) \wedge \mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1 \wedge \mathsf{spdu}_i$ of correct form, then process BSM_i, update st with C_i, and return 1. For $i = \alpha$: $\mathsf{C}_U \leftarrow \mathsf{CCons}(\mathsf{C}_1, ..., \mathsf{C}_\alpha)$. If $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{C}_U, \mathsf{st}) = 1 \wedge h = \mathsf{H}(\mathsf{C}_U) \wedge \mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1 \wedge \mathsf{spdu}_i$ of correct form, then process BSM_i, update st with $\mathsf{H}(\mathsf{C}_U)$ and pk_U, and return 1. For $i \in [\alpha+1, 5]$: if $h = \mathsf{H}(\mathsf{C}_U) \wedge \mathsf{Vrfy}^{\mathsf{pq}}(\mathsf{pk}_U^{\mathsf{pq}}, \mathsf{sig}_i^{\mathsf{pq}}, \mathsf{BSM}_i) = 1 \wedge \mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1 \wedge \mathsf{spdu}_i$ of correct form, then process BSM_i and return 1, with $(\mathsf{sig}_i^{\mathsf{c}} || \mathsf{sig}_i^{\mathsf{pq}}) \leftarrow \mathsf{sig}_i$. Else, return 0.

PQ Instantiations Only signature schemes whose explicit certificate can be sent in five or less fragments α can be used in the *True Hybrid* design. After careful analysis of all schemes given in Table 1, Falcon is the only viable scheme. We chose to instantiate $\alpha = 1$. To be more concrete, using explicit ECDSA certificates the frame F_1 is $1,835$ bytes, as it contains the BSM, the ECDSA signature, and the entire certificate C_U (see Table 2, column "F_1"). Frames 2,3,4, and 5, each contain a BSM, $\mathsf{H}(\mathsf{C}_U)$, and ECDSA and Falcon signatures, totalling to 834 bytes each (see Table 2, columns "F_2" - "F_5").

Although the Dilithium and XMSS certificates can be split similarly, the size of Dilithium and XMSS signatures alone exceed the allowed frame size of $2,304$ bytes, it would not be possible to transmit a signed message. Hence, we do not instantiate our design using them. We also needed to rule out Picnic for its large

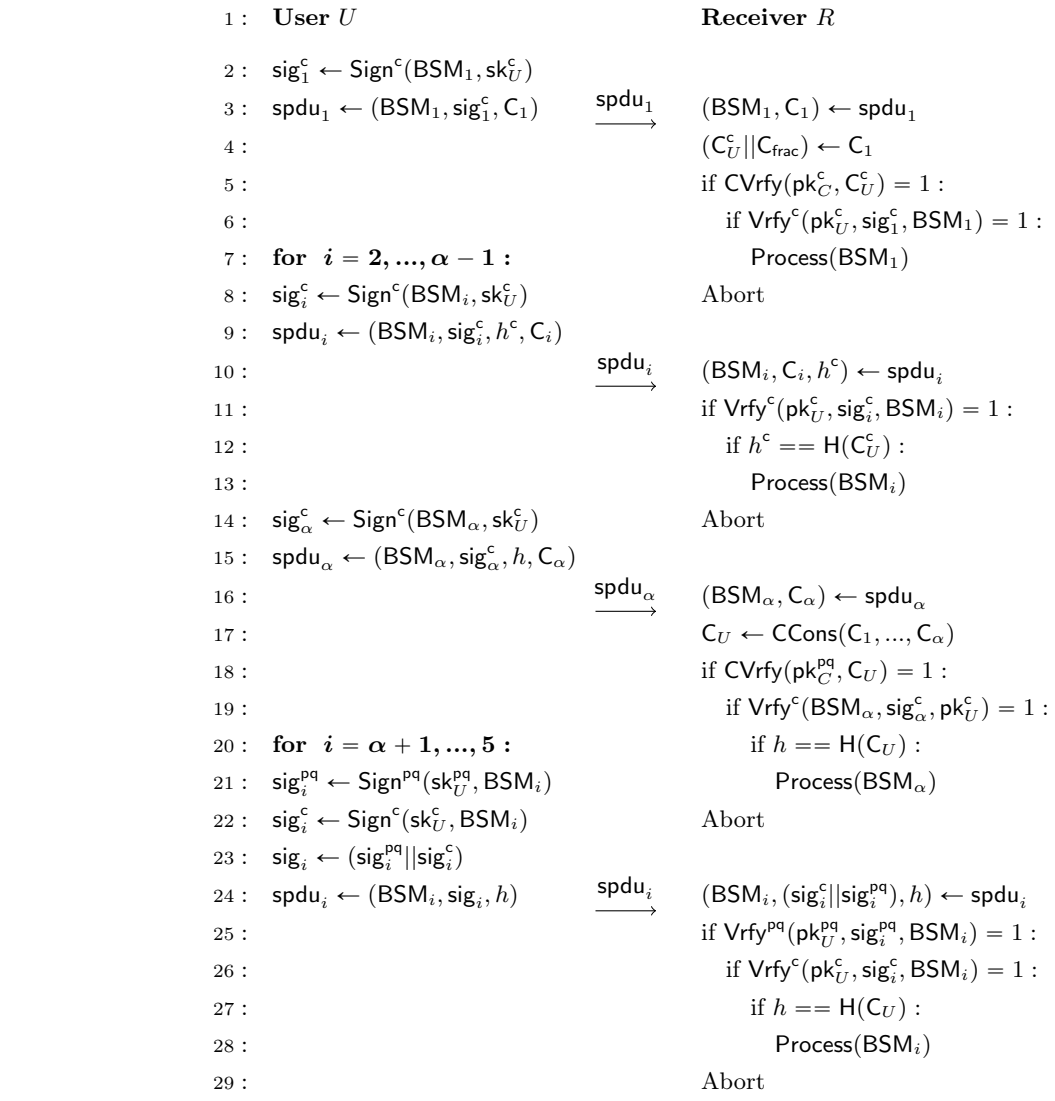

Figure 3: Pseudo-code description of true hybrid design to be repeated every five BSMs; we omit an explicit description of st.

signature sizes. Moreover, the public key sizes of Rainbow, GeMSS and Sphincs$^+$ are too large to be split into five fragments of sufficiently small size.

While our true hybrid design gives the strongest security guarantees, it is also the least flexible, motivating our backwards compatible design.

5.3 Backwards-compatible Hybrid Design

This design is essentially the same as the true hybrid approach described above. The difference lies in whether the receiver runs verification on the PQ signature or not, more concretely in the handling of the $[\alpha, 5]$-th SPDUs in SPDUVerify. We give the pseudo-code that is different from the true hybrid in Figure 4.

We assume that all users send and expect to receive the hybrid certificates, even if they do not possess the hardware capabilities to verify the PQ signature, in order to prevent rollback attacks. We note this enforcement only adds security for receivers who actually verify the PQ signature and certificates. The advantage of this approach is that vehicles whose *software* but not their *hardware* has been updated (for example, they are not equipped with dedicated Falcon hardware modules) will be able to do continue to verify ECDSA signatures nevertheless. Vehicles with the necessary hardware, however, will rely on PQ

1 :	**User** U		**Receiver** R
2 :	**for** $\boldsymbol{i = 1, ..., \alpha - 1}$ **:**		
3 :	Same as true hybrid, Figure 3		
4 :	$\mathsf{sig}^{\mathsf{c}}_{\alpha} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{BSM}_{\alpha}, \mathsf{sk}^{\mathsf{c}}_{U})$		
5 :	$\mathsf{spdu}_{\alpha} \leftarrow (\mathsf{BSM}_{\alpha}, \mathsf{sig}^{\mathsf{c}}_{\alpha}, h, \mathsf{C}_{\alpha})$		
6 :		$\xrightarrow{\mathsf{spdu}_{\alpha}}$	$(\mathsf{BSM}_{\alpha}, \mathsf{C}_{\alpha}) \leftarrow \mathsf{spdu}_{\alpha}$
7 :			$\mathsf{C}_U \leftarrow \mathsf{CCons}(\mathsf{C}_1, ..., \mathsf{C}_{\alpha})$
8 :			if PQ supported:
9 :			if $\mathsf{CVrfy}(\mathsf{pk}^{\mathsf{pq}}_{C}, \mathsf{C}^{\mathsf{pq}}_{U}, \mathsf{st}) = 0$:
10 :			Abort
11 :			if $h \neq \mathsf{H}(\mathsf{C}_U)$: Abort
12 :	**for** $\boldsymbol{i = \alpha + 1, ..., 5}$ **:**		if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{BSM}_{\alpha}, \mathsf{sig}^{\mathsf{c}}_{\alpha}, \mathsf{pk}^{\mathsf{c}}_{U}) = 0$:
13 :	$\mathsf{sig}^{\mathsf{pq}}_{i} \leftarrow \mathsf{Sign}(\mathsf{BSM}_{i}, \mathsf{sk}^{\mathsf{pq}}_{U})$		Abort
14 :	$\mathsf{sig}^{\mathsf{c}}_{i} \leftarrow \mathsf{Sign}(\mathsf{BSM}_{i}, \mathsf{sk}^{\mathsf{c}}_{U})$		$\mathsf{Process}(\mathsf{BSM}_{\alpha})$
15 :	$\mathsf{sig}_{i} \leftarrow (\mathsf{sig}^{\mathsf{pq}}_{i} \| \mathsf{sig}^{\mathsf{c}}_{i})$		
16 :	$\mathsf{spdu}_{i} \leftarrow (\mathsf{BSM}_{i}, \mathsf{sig}_{i}, h)$	$\xrightarrow{\mathsf{spdu}_{i}}$	$(\mathsf{BSM}_{i}, \mathsf{sig}_{i}, h) \leftarrow \mathsf{spdu}_{i}$
17 :			if $h \neq \mathsf{H}(\mathsf{C}_U)$: Abort
18 :			$\mathsf{sig}^{\mathsf{c}}_{i} \leftarrow \mathsf{sig}_{i}$
19 :			if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{BSM}_{i}, \mathsf{sig}^{\mathsf{c}}_{i}, \mathsf{pk}^{\mathsf{c}}_{U}) = 0$:
20 :			Abort
21 :			if PQ supported:$\mathsf{sig}^{\mathsf{pq}}_{i} \leftarrow \mathsf{sig}_{i}$
22 :			if $\mathsf{Vrfy}^{\mathsf{pq}}(\mathsf{BSM}_{i}, \mathsf{sig}_{i}, \mathsf{pk}^{\mathsf{pq}}_{U}) = 0$:
23 :			Abort
24 :			$\mathsf{Process}(\mathsf{BSM}_{i})$

Figure 4: Pseudo-code description of backwards compatible hybrid design to be repeated every five BSMs; we omit an explicit description of st.

security in addition to the security given by ECDSA, assuming the receiver runs the verification algorithms honestly. It is important to emphasize that the receiver can reconstruct and hash the certificate even if they are not able to verify the PQ signatures and certificates.

PQ Instantiations The *backwards compatible* instantiations are identical to the *true hybrid* as the same information is transmitted. Therefore, Falcon is the only suitable PQ scheme for this design.

5.4 Partially PQ Design

Our true and backwards-compatible hybrid as well as the purely PQ design, all lead to more than five times larger frames compared to the pure ECDSA design. Hence, during the PQ transition phase the safety threat by frame loss is much higher than the one by quantum attackers forging signatures on the SPDU. The reason is that during the transition strict requirements on the hardware (i.e., the upper bound $2,304$ bytes on the frame size) are necessary for safety (since more powerful hardware has yet to be developed and deployed), while powerful quantum computers to forge ECDSA signatures within a week cannot be built yet. This motivates our *partially PQ* design. The core idea is to allow signatures whose respective verification keys are only valid for short periods, to rely on classical security. However, a CA signs the user's ECDSA key pk_U using a PQ signature, i.e., the CA has two key pairs: ECDSA keys $(\mathsf{pk}^{\mathsf{c}}_{C}, \mathsf{pk}^{\mathsf{c}}_{C})$ and PQ keys $(\mathsf{sk}^{\mathsf{pq}}_{C}, \mathsf{pk}^{\mathsf{pq}}_{C})$. This means ECDSA keys that are refreshed more often, so they are only valid for a shorter time window, can be used to sign BSMs. Keys that are valid longer, like those used to generate certificates, have to be protected with PQC. Put differently, this design protects the *integrity* of the ECDSA key pk_U using ECDSA/ECQV *and* PQ signatures. Our approach of carefully analyzing the quantum powers is inspired by the *quantum*

annoying property of [40].

As in the previous design, we add backwards-compatibility by leaving it to the receiver's choice to verify the PQ certificates. As in Section 5.3, we assume that every user sends and expects to receive the hybrid certificate.

Description of the Partially PQ Design Let $\mathcal{P}$ be the partially PQ hybrid protocol defined as follows using the two signature schemes $\mathcal{S}_{\mathsf{c}}$ and $\mathcal{S}_{\mathsf{pq}}$. We give a pseudo-code description in Figure 5.

KGen$_C$ returns $(\mathsf{pk}_C, \mathsf{sk}_C)$ as in the true hybrid design.

CGen generates $\mathsf{C}_U = (\mathsf{C}_U^{\mathsf{c}} || \mathsf{C}_U^{\mathsf{pq}})$ for user key pair $(\mathsf{sk}_U^{\mathsf{c}}, \mathsf{pk}_U^{\mathsf{c}}) \in \mathrm{Im}(\mathsf{KGen}_{\mathsf{c}})$, i.e., $\mathsf{C}_U^{\mathsf{c}}$ is an ECDSA/ECQV-based certificate over $(U, \mathsf{pk}_U^{\mathsf{c}})$ with cbody_U. $\mathsf{C}_S^{\mathsf{pq}}$ is also over $(U, \mathsf{pk}_U^{\mathsf{c}})$ including a signature $\mathsf{sig}^{\mathsf{pq}} \leftarrow \mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_C^{\mathsf{pq}}, \mathsf{cbody}_U)$.

SPDUGen computes $\mathsf{sig}_i^{\mathsf{c}} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_U^{\mathsf{c}}, \mathsf{BSM}_i)$ and returns

$$\mathsf{spdu}_i = \begin{cases} (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{C}_i), & \text{for } i = 1 \\ (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h^{\mathsf{c}}, \mathsf{C}_i), & \text{for } i \in [2, \alpha - 1] \\ (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h, \mathsf{C}_i), & \text{for } i = \alpha \\ (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h), & \text{for } i \in [\alpha + 1, 5]. \end{cases}$$

SPDUVerify is defined as follows. For $i \in [1, \alpha - 1]$: as in the true hybrid design. For $i = \alpha$: $\mathsf{C}_U \leftarrow \mathsf{CCons}(\mathsf{C}_1, ..., \mathsf{C}_\alpha)$ If PQ is not supported, return 1 if $h = \mathsf{H}(\mathsf{C}_U) \wedge \mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1 \wedge \mathsf{spdu}_i$ of correct form. If PQ is supported, return 1 if *in addition* $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{C}_U, \mathsf{st}) = 1 \wedge \mathsf{pk}'_U = \mathsf{pk}_U$, with $\mathsf{pk}'_U \leftarrow \mathsf{C}_U^{\mathsf{pq}}$. If 1 would be returned, process BSM_i and update st with $\mathsf{H}(\mathsf{C}_U)$ in both cases. Otherwise, return 0. For $i \in [\alpha + 1, 5]$: if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1 \wedge h = \mathsf{H}(\mathsf{C}_U) \wedge \mathsf{spdu}_i$ of correct form, then process BSM_i and return 1. Otherwise, return 0.

PQ Instantiation Table 2 presents the frame sizes for each instantiation of the *Partially PQ* design for viable PQ schemes. The sizes of the ECDSA explicit/implicit certificate $\mathsf{C}_U^{\mathsf{c}}$ are 136/72 bytes. Moreover, we compute the total frame size including the ECDSA-signed BSM and fragments of C_U with different instantiations of α. Viable PQ schemes for this design are Falcon, Dilithium, XMSS, and Sphincs$^+$ as explained next. For Falcon, the size of the explicit cert $\mathsf{C}_U^{\mathsf{pq}}$ is 858 bytes as it includes a Falcon signature over an ECDSA key (see Table 2, column C_U). Therefore, it is not necessary to fragment C_U and one message is sufficient to communicate it (i.e., $\alpha = 1$). Hence, F_1 is 970 bytes and the remaining frames F_2, F_3, F_4, F_5 are 144 bytes each.

Dilithium, XMSS, and Sphincs$^+$, instantiations require larger values of α, as seen in Table 2, which translates to more messages being transmitted before the integrity of the ECDSA key can be guaranteed by both, classical and PQ signatures.

When considering Picnic, we deduced that even fragmentation into five parts is not small enough to communicate the certificate within five messages. Hence, Picnic is cannot be used in our partially PQ hybrid design.

Technically, the hybrid certificate using Rainbow or GeMMS could be communicated in less than five fragments, indicating that they could work in this design. Unfortunately, in practice they are likely not viable due to their very large public keys. For example, distributing the CA's hybrid certificate via P2PCD would take 70 frames (which translates to about 7 seconds) for Rainbow and 150 (i.e., about 15 seconds) for GeMSS, which we consider to be too many for practical purposes as during this time the safety is not guaranteed. [5]

[5] If all CA certificates are present on the car, then P2PCD would not be necessary, and GeMSS and Rainbow might be viable. However, for now P2PCD is required under IEEE 1609.2 and 1609.2.1 [8,17].

1 :	**User** U		**Receiver** R
2 :	$\mathsf{sig}_1 \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_U^{\mathsf{c}}, \mathsf{BSM}_1)$		
3 :	$\mathsf{spdu}_1 \leftarrow (\mathsf{BSM}_1, \mathsf{sig}_1, \mathsf{C}_1)$	$\xrightarrow{\mathsf{spdu}_1}$	$(\mathsf{BSM}_1, \mathsf{sig}_1, \mathsf{C}_U^{\mathsf{c}}) \leftarrow \mathsf{spdu}_1$
4 :			if $\mathsf{CVrfy}(\mathsf{pk}_C^{\mathsf{c}}, \mathsf{C}_U^{\mathsf{c}}) = 1$:
5 :			if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U, \mathsf{sig}_1, \mathsf{BSM}_1) = 1$:
6 :	**for** $i = 2, ..., \alpha - 1$:		$\mathsf{Process}(\mathsf{BSM}_1)$
7 :	$\mathsf{sig}_i \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_U^{\mathsf{c}}, \mathsf{BSM}_i)$		Abort
8 :	$\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h^{\mathsf{c}}, \mathsf{C}_i)$		
9 :		$\xrightarrow{\mathsf{spdu}_i}$	$(\mathsf{BSM}_i, \mathsf{sig}_i, \mathsf{C}_i, h^{\mathsf{c}}) \leftarrow \mathsf{spdu}_i$
10 :			if $h^{\mathsf{c}} == \mathsf{H}(\mathsf{C}_U^{\mathsf{c}})$
11 :			if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U, \mathsf{sig}_i, \mathsf{BSM}_i) = 1$:
12 :			$\mathsf{Process}(\mathsf{BSM}_i)$
13 :	$\mathsf{sig}_\alpha \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_U^{\mathsf{c}}, \mathsf{BSM}_\alpha)$		Abort
14 :	$\mathsf{spdu}_\alpha \leftarrow (\mathsf{BSM}_\alpha, \mathsf{sig}_\alpha, \mathsf{C}_\alpha, h^{\mathsf{c}})$		
15 :		$\xrightarrow{\mathsf{spdu}_\alpha}$	$(\mathsf{BSM}_\alpha, \mathsf{sig}_\alpha, \mathsf{C}_\alpha, h^{\mathsf{c}}) \leftarrow \mathsf{spdu}_\alpha$
16 :			$\mathsf{C}_U \leftarrow \mathsf{CCons}(\mathsf{C}_1, ..., \mathsf{C}_\alpha)$
17 :			if C_U is not hybrid : Abort
18 :			if PQ supported:
19 :			if $\mathsf{CVrfy}(\mathsf{pk}_C^{\mathsf{pq}}, \mathsf{C}_U^{\mathsf{pq}}) = 1$:
20 :			$\mathsf{pk}_U \leftarrow \mathsf{C}_U$
21 :			if $\mathsf{pk}'_U \neq \mathsf{pk}_U$: Abort
22 :			Abort
23 :			if $h == \mathsf{H}(\mathsf{C}_U)$:
24 :			if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U, \mathsf{sig}_\alpha, \mathsf{BSM}_i) = 1$:
25 :	**for** $i = \alpha + 1, ..., 5$:		$\mathsf{Process}(\mathsf{BSM}_\alpha)$
26 :	$\mathsf{sig}_i \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{BSM}_i, \mathsf{sk}_U^{\mathsf{c}})$		Abort
27 :	$\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}_i, h)$		
28 :		$\xrightarrow{\mathsf{spdu}_i}$	$(\mathsf{BSM}_i, \mathsf{sig}_i, h) \leftarrow \mathsf{spdu}_i$
29 :			if $h \neq \mathsf{H}(\mathsf{C}_U)$: Abort
30 :			if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_U, \mathsf{sig}_i, \mathsf{BSM}_i) = 1$:
31 :			$\mathsf{Process}(\mathsf{BSM}_i)$
32 :			Abort

Figure 5: Pseudo-code description of the Partially PQ Design to be repeated every five BSMs; we omit an explicit description of st.

5.5 Discussion

Across all requirements, our fundamental objective was to show how the maximum number of vehicles supported by the V2V system—i.e., the *system capacity*—is impacted by the addition of PQC alongside (or in place of) ECDSA.

The end-to-end latency of a BSM depends largely on the size of the frame containing that BSM, as larger frames take longer to transmit over the air than smaller ones. Therefore, a primary challenge of using PQC in V2V is that the larger key and signature sizes of PQC result in larger frames and, consequently, significant additional latency. Further, since only one vehicle can transmit at a time in DSRC, the longer transmission times resulting from larger frame sizes mean fewer vehicles can each transmit one BSM within a 100 ms BSM interval (as required in DSRC), thus reducing the capacity of the V2V system.

As a worst-case scenario, we considered our designs' performance under the slowest possible data rate allowed in DSRC (3 Mbps) which occurs when using binary phase-shift keying (BPSK) modulation and a 1/2

code rate [41]. As shown in Table 2, our calculations anticipate that our PQ-V2V designs all reduce system capacity by varying amounts. Illustrating the challenges of fully integrating PQC into DSRC-based V2V, Table 2 shows that our Pure Falcon design reduces system capacity by 82% (just 33 vehicles vs. 183 under ECDSA). However, our Partially-PQ designs are expected to perform better. In particular, our Partially-PQ Falcon design only reduces system capacity by about 41%, a significant but far less damaging impact than that of our Pure Falcon design. Importantly, we reiterate that the results in Table 2 are strictly for a worst-case scenario; i.e., they are *extreme minimums* for system capacity. Under more likely conditions, and potentially using other technologies with faster data rates than DSRC (e.g., C-V2X), this impact will be less severe. Altogether, our initial analysis contends that our designs for PQ-V2V are potentially viable and would be significantly more practical than a naive adoption of PQC in V2V. We leave a full exploration of our designs' performance under varying communication conditions and with other V2V technologies to future work.

Table 2: Resulting sizes of frames F_i (in bytes), maximum number of frames λ_{BSM}, maximum number of supported vehicles $v_{\max}$, and i-unforgeability of V2V Designs instantiated with viable PQ signature schemes and/or ECDSA (using explicit ECDSA certificates); Column's $|\mathsf{C}_U|$ ($|\mathsf{C}_i|$) gives size of certificate (fragments); shaded cells indicate SPDUs which are PQ-authenticated; (i-UF) under the assumption that ECDSA signatures cannot be forged within a week

PQ Scheme	$\lvert\mathsf{C}_U\rvert$	α	$\lvert\mathsf{C}_i\rvert$	F_1	F_2	F_3	F_4	F_5	$v_{\max}$	i-UF	
Pure ECDSA Design											
-	136	1	136	248	144	144	144	144	183	1-UF	-
True/Backwards Compatible Hybrid Design											
Falcon	1,649	1	1,649	1,835	834	834	834	834	31	1-UF	2-UF
Partially PQ Hybrid Design											
Falcon	858	1	858	970	144	144	144	144	107	1-UF	(2-UF)
Dilithium	2,588	2	1,294	1,406	1,406	144	144	144	54		(3-UF)
Sphincs$^+$	8,024	4	2,006	1,717	1,717	1,717	1,717	1,717	21		(4-UF)
XMSS	2,860	2	1,430	1,542	1,542	144	144	144	50		(3-UF)
Pure PQ Design											
Falcon	1,627	1	1,627	1,552	1,552	770	770	770	33	2-UF	2-UF

6 Related Work

Classical-PQ Hybrid Cryptography Transitioning to PQC has been explored in [13, 14] for TLS and SSH network protocols. The constraints in integrating PQ algorithms, such as lack of ability to replace (drop-in) or negotiate cryptographic algorithms, and limitations on sizes of keys or messages, have been highlighted. However, the constraints of V2V are very different from the ones in TLS/SSH, most notably in that vehicles cannot negotiate algorithms as messages are *broadcast* and need to be processed as soon as it is in the proximity of another vehicle. Foundations for adapting the public-key infrastructures for the transition to PQC to achieve security and functionality both in PQ-aware and -unaware systems (using backwards-compatible solutions) are laid in [12]. The authors propose theoretical hybrid signature schemes and show how to use them in TLS 1.2, S/MIME, and X.509 certificates. Our work is different in that while we take inspiration from their theoretical constructions, our designs are specifically tailored to the much more restricted V2V application.

PQC in V2V Authentication PQC has occasionally been examined against the threats posed by quantum computers to vehicles in a connected, V2X environment. These examinations have tended to focus more on intra-vehicle communications (e.g., wired connections between electronic control units (ECU)) than on external, wireless communication like V2V. For example, the performance impact of replacing classical algorithms with PQC for inter-ECU communication was explored in [18]. The authors thoroughly explored the challenges of using PQC in this particular automotive context, but their solutions focus on solving challenges like data encryption that do not apply to our work concerning V2V safety messages. Similarly, the authors of [42] explored the challenges of implementing PQC on a microcontroller for use in vehicles. While their motivation parallels ours–they, too, specifically call out the imminent vulnerability of classical cryptography to be attacked by a capable quantum computer–our work differs substantially from their focus on replacing the elliptic-curve-based data encryption and key exchange procedures defined in 1609.2. In fact, the authors

explicitly place development of a PQ signature scheme for V2V out of scope for their work [42]. To the best of our knowledge, our work is the first to undertake the specific challenge of devising and evaluating a PQC signature scheme for use in the specific, critical context of V2V safety messages.

7 Conclusion and Future Work

In this paper, we presented novel, classical-hybrid approaches for integrating PQC into the IEEE 1609.2 standard. We analyzed PQ algorithms in the context of V2V and then used our findings to construct novel, hybrid designs for PQ-V2V to use during the classical-to-PQ transition period. We showed how the large keys and signatures of PQC, until now a formidable obstacle to applying PQC in V2V, can be accommodated by our designs without excessively sacrificing V2V performance (i.e., number of supported vehicles).

In future work, we will develop a formal security analysis to demonstrate the security of our designs. In addition, we will evaluate our most promising PQ-V2V designs using real V2V hardware in an effort to further refine our designs. This will be a stepping stone towards future evaluation of our PQ-V2V designs using vehicle-mounted hardware in roadway tests, the logical next step of our work to develop our PQ-V2V designs into real-world systems. Finally, we intend to expand the scope of our theoretical and experimental analysis to include CAs and their associated infrastructure as we move towards a holistic, top-down approach to securing V2V against the approaching quantum threat.

Acknowledgements

We would like to thank Dan Brown and Matthew Campagna for clarifications on the ECDSA-ECQV security, Douglas Stebila and Fernando Virdia for fruitful discussions on the security of V2V protocols. Moreover, we are thankful to anonymous reviewers about suggestions how to extend earlier versions of this paper.

References

[1] National Highway Traffic Safety Administration, "Technical report 11078-101414-v2a," 2014, Accessed: Feb. 20, 2020. [Online]. Available: https://bit.ly/35EggyG

[2] J. Brodkin, "FCC takes spectrum from auto industry in plan to "supersize" Wi-Fi," Nov. 2020, Accessed: July 15, 2021. [Online]. Available: https://arstechnica.com/tech-policy/2020/11/fcc-adds-45mhz-to-wi-fi-promising-supersize-networks-on-5ghz-band/

[3] S. Patil, "How NR-based sidelink expands 5G C-V2X to support new advanced use cases," Qualcomm Technologies, Inc., Mar. 2020, Accessed: May 20, 2020. [Online]. Available: https://www.qualcomm.com/invention/5g/cellular-v2x

[4] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment: Enhancements for Next Generation V2X*, IEEE Standard P802.11bd.

[5] U.S. Department of Transportation, "Report to congress: Vehicle safety recall completion rates report," Dec. 2018, Accessed: November 9, 2021. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.gov/files/documents/18-3122_vehicle_safety_recall_completion_rates_report_to_congress-tag.pdf

[6] AARP, "How today's cars are built to last: Tech advances help vehicles stay safe and sound for more years than ever before," 2018, Accessed: July 14, 2021. [Online]. Available: https://www.aarp.org/auto/trends-lifestyle/info-2018/how-long-do-cars-last.html

[7] M. Mosca and M. Piani, "Quantum Threat Timeline Report 2020," Global Risk Institute in Financial Services(GRI), https://globalriskinstitute.org/publications/quantum-threat-timeline-report-2020/, 2021, accessed: 2021-12-01.

[8] *Wireless Access in Vehicular Environments–Security Services for Applications and Management Messages*, IEEE, 2016, iEEE Standard 1609.2-2016.

[9] W. Barker, W. Polk, and M. Souppaya, "Getting ready for post-quantum cryptography: Explore challenges associated with adoption and use of post-quantum cryptographic algorithms," *the publications of NIST Cyber Security White Paper (DRAFT), CSRC. NIST. GOV*, vol. 26, 2020.

[10] B. Brecht, D. Therriault, A. Weimerskirch, W. Whyte, V. Kumar, T. Hehn, and R. Goudy, "A security credential management system for V2X communications," *IEEE Trans. on Intelligent Transportation Systems*, vol. 19, no. 12, pp. 3850—3871, Dec. 2018.

[11] J. Howe, T. Prest, and D. Apon, "Sok: How (not) to design and implement post-quantum cryptography," in *Topics in Cryptology - CT-RSA 2021 - Cryptographers' Track at the RSA Conference 2021, Virtual Event, May 17-20, 2021, Proceedings*, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed., vol. 12704. Springer, 2021, pp. 444–477. [Online]. Available: https://doi.org/10.1007/978-3-030-75539-3_19

[12] N. Bindel, U. Herath, M. McKague, and D. Stebila, "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*, ser. Lecture Notes in Computer Science, T. Lange and T. Takagi, Eds., vol. 10346. Springer, 2017, pp. 384–405. [Online]. Available: https://doi.org/10.1007/978-3-319-59879-6_22

[13] E. Crockett, C. Paquin, and D. Stebila, "Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH," Cryptology ePrint Archive, Report 2019/858, 2019, https://eprint.iacr.org/2019/858.

[14] P. Kampanakis and D. Sikeridis, "Two post-quantum signature use-cases: Non-issues, challenges and potential solutions," in *Proceedings of the 7th ETSI/IQC Quantum Safe Cryptography Workshop*, Nov. 2019.

[15] *Wireless Access in Vehicular Environments–Security Services for Applications and Management Messages - Amendment 1*, IEEE Standard 1609.2a-2017, IEEE, 2017.

[16] *Wireless Access in Vehicular Environments–Security Services for Applications and Management Messages - Amendment 2*, IEEE Standard 1602.2b-2019, IEEE, 2019.

[17] *Wireless Access in Vehicular Environments (WAVE)–Certificate Management Interfaces for End Entities*, IEEE Standard 1609.2.1-2020, IEEE, 2020.

[18] P. Ravi, V. K. Sundar, A. Chattopadhyay, S. Bhasin, and A. Easwaran, "Authentication protocol for secure automotive systems: Benchmarking post-quantum cryptography," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, Seville, Spain, 2020.

[19] A. Huelsing, D. Butin, S. Gazdag, J. Rijneveld, and A. Mohaisen, "Xmss: extended merkle signature scheme," Internet Requests for Comments, RFC Editor, RFC 8391, May 2018.

[20] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments*, IEEE, 2010, iEEE Standard 802.11p.

[21] 3rd Generation Partnership Project, *Summary of Rel-14 Work Items - Technical Report 21.914 V14.0.0*, 2018.

[22] *Summary of Rel-16 Work Items*, 3rd Generation Partnership Project Technical Report 21.915 V16.0.0, 2020.

[23] C. F. Kerry and P. D. Gallagher, "Digital signature standard (DSS)," 2013, Accessed: July 14, 2021. [Online]. Available: http://people.csail.mit.edu/alinush/6.857-spring-2015/papers/dsa.pdf

[24] Internet Engineering Task Force, "RFC 5639: ECC Brainpool standard curves & curve generation," 2010, Accessed: July 15, 2021. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc5639

[25] *SEC 4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV)*, Standards for Efficient Cryptography, Certicom Research, 2013.

[26] *Dedicated Short Range Communication (DSRC) Systems Engineering Process Guidance for SAE J2945/X Documents and Common Design Concepts*, SAE Standard J2945 2017-12, SAE International, 2017.

[27] Internet Engineering Task Force, "RFC 8554: Leighton-micali hash-based signatures," 2019, Accessed: December 02, 2021. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8554

[28] W. L. Ross and W. Copan, "Digital signature standard (DSS)," 2013, Accessed: December 02, 2021. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5-draft.pdf

[29] F. Campos, T. Kohlstadt, S. Reith, and M. Stöttinger, "LMS vs XMSS: comparison of stateful hash-based signature schemes on ARM cortex-m4," in *Progress in Cryptology - AFRICACRYPT 2020 - 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20-22, 2020, Proceedings*, ser. Lecture Notes in Computer Science, A. Nitaj and A. M. Youssef, Eds., vol. 12174. Springer, 2020, pp. 258–277. [Online]. Available: https://doi.org/10.1007/978-3-030-51938-4_13

[30] T. Prest, P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "FALCON," National Institute of Standards and Technology, Tech. Rep., 2020, available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[31] V. Lyubashevsky, L. Ducas, E. Kiltz, T. Lepoint, P. Schwabe, G. Seiler, D. Stehlé, and S. Bai, "CRYSTALS-DILITHIUM," National Institute of Standards and Technology, Tech. Rep., 2020, available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[32] J. Ding, M.-S. Chen, A. Petzoldt, D. Schmidt, B.-Y. Yang, M. Kannwischer, and J. Patarin, "Rainbow," National Institute of Standards and Technology, Tech. Rep., 2020, available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[33] G. Zaverucha, M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, J. Katz, X. Wang, V. Kolesnikov, and D. Kales, "Picnic," National Institute of Standards and Technology, Tech. Rep., 2020, available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[34] A. Casanova, J.-C. Faugère, G. Macario-Rat, J. Patarin, L. Perret, and J. Ryckeghem, "GeMSS," National Institute of Standards and Technology, Tech. Rep., 2020, available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[35] A. Hulsing, D. J. Bernstein, C. Dobraunig, M. Eichlseder, S. Fluhrer, S.-L. Gazdag, P. Kampanakis, S. Kolbl, T. Lange, M. M. Lauridsen, F. Mendel, R. Niederhagen, C. Rechberger, J. Rijneveld, P. Schwabe, J.-P. Aumasson, B. Westerbaan, and W. Beullens, "SPHINCS+," National Institute of Standards and Technology, Tech. Rep., 2020, available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[36] *IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages*, IEEE Standard 1609.2, 2016.

[37] A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi, "A closer look at PKI: security and efficiency," in *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings*, ser. Lecture Notes in Computer Science, T. Okamoto and X. Wang, Eds., vol. 4450. Springer, 2007, pp. 458–475. [Online]. Available: https://doi.org/10.1007/978-3-540-71677-8_30

[38] U.S. Dept. of Transportation, "49 CFR Part 571: Notice of Proposed Rulemaking," 2017, Accessed: Jul. 30, 2021. [Online]. Available: https://www.govinfo.gov/content/pkg/FR-2017-01-12/pdf/2016-31059.pdf

[39] S. Hu, Q. A. Chen, J. Sun, Y. Feng, Z. M. Mao, and H. X. Liu, "Automated discovery of denial-of-service vulnerabilities in connected vehicle protocols," in *30th {USENIX} Security Symposium ({USENIX} Security)*, 2021.

[40] E. Eaton and D. Stebila, "The "quantum annoying" property of password-authenticated key exchange protocols," in *Post-Quantum Cryptography - 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20-22, 2021, Proceedings*, ser. Lecture Notes in Computer Science, J. H. Cheon and J. Tillich, Eds., vol. 12841. Springer, 2021, pp. 154–173. [Online]. Available: https://doi.org/10.1007/978-3-030-81293-5_9

[41] I. C. Society, "IEEE Std. 802.11p-2010," IEEE, Tech. Rep., 2010.

[42] T. Fritzmann, J. Vith, and J. Sepulveda, "Post-quantum key exchange mechanism for safety critical systems," 2019, Accessed: July 14, 2021. [Online]. Available: https://hss-opus.ub.ruhr-uni-bochum.de/opus4/frontdoor/deliver/index/docId/6653/file/Kapitel2.pdf

A Definitions of Cryptographic Primitives

Definition 2 (Digital Signature Scheme)**.** *A Digital Signature Scheme is defined as a tuple of algorithms* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$*, which are defined as follows:*

KGen *returns a public key* pk *and secret key* sk*.*

Sign *returns a signature* sig *on a message* m *using* sk*.*

Vrfy *returns 0 or 1. Upon input of a message* m*, a signature* sig*, and the public key* pk*, this returns 1 if the signature is valid. Otherwise, 0 is returned.*

$\mathcal{S}$ *is considered correct if* $Pr[\mathsf{Vrfy}(\mathsf{pk}, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}), \mathsf{m}) \mid (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}()] = 1$.

Next we give the definition of certified signature schemes [37]. As we aim at giving a generalized definition for explicit, implicit and hybrid certificates, our algorithms are defined very generically. Following [37], we assume that the pair (U, pk_U) is uniquely bound in the certificate C_U.

Definition 3 (Certified Signature Scheme)**.** *A certified signature scheme* $\mathcal{C} = (\mathsf{KGen}_C, \mathsf{CGen}(\mathsf{CGen}_C, \mathsf{CGen}_U), \mathsf{Sign}, \mathsf{Vrfy})$ *is defined via the following polynomial-time algorithms.*

KGen_C *returns a public key* pk_C *and secret key* sk_C *belonging to the certificate authority C.*

$\mathsf{CGen}(\mathsf{CGen}_C, \mathsf{CGen}_U)$ *is an interactive (two-party) public-key registration protocol, involving the user U and the CA C running their (randomized) sub-protocols* CGen_C *and* CGen_U*, respectively.* CGen_C *takes input a secret key* sk_C*;* CGen_U *takes input the identity U of a user and the public key* pk_C *corresponding to* sk_C*. As result of the interaction, the output of* CGen_C *is* $(U, \mathsf{pv}_U, \mathsf{C}_U)$*, where* pv_U *is a public key value* pv_U*, corresponding to a public key* pk_U*, and* C_U *is an issued certificate. If* C_U *is an explicit certificate,* $\mathsf{pv}_U = \mathsf{pk}_U$*; if it is implicit,* pv_U *is the reconstruction value; if it is a hybrid certificate combining two or more sub-certificates,* pv_U *is the concatenation of the corresponding public key values. The local output of* CGen_U *is* $(U, \mathsf{pk}_U, \mathsf{sk}_U, \mathsf{C}_U)$*, where* sk_U *is a secret key U will use to sign messages. The owner of* sk_C *should not learn* sk_U *during* CGen*. Either party can quit the execution prematurely, in which case the output of the party is set to* $\perp$*.*

Sign *is a (possibly) randomized signing algorithm. It takes input an identity U, a secret key* sk_U*, a certificate cert* C_U*, the CA's public key* pk_C *and a message* m*, and outputs a signature* sig*.*

Vrfy *is a deterministic verification algorithm. It takes input an identity U, a public key* pk_U*, a certificate* C_U*, a public key* pk_C*, a message* m*, and a signature* sig*, and outputs 0 or 1. In the latter case, we say that* sig *is a valid signature for* m *relative to* $(U, \mathsf{pk}_U, \mathsf{C}_U, \mathsf{pk}_C)$*. If* C_U *is an implicit certificate this also involves the reconstruction of the U's public key.*

Following [37], $\mathcal{C}$ *is correct if for all* m *and U it holds that if* $((U, \mathsf{pk}_U, \mathsf{C}_U), (U, \mathsf{pk}_U, \mathsf{sk}_U, \mathsf{C}_U)) \leftarrow \mathsf{CGen}(\mathsf{CGen}_C(\mathsf{sk}_C), \mathsf{CGen}_U(U, \mathsf{pk}_C))$ *for* $(\mathsf{pk}_C, \mathsf{sk}_C) \leftarrow \mathsf{KGen}_C()$*, then* $\mathsf{Vrfy}(U, \mathsf{pk}_U, \mathsf{C}_U, \mathsf{pk}_C, \mathsf{m}, \mathsf{Sign}(U, \mathsf{sk}_U, \mathsf{C}_U, \mathsf{pk}_C, \mathsf{m})) = 1$.