When Cryptography Needs a Hand: Practical Post-Quantum Authentication for V2V Communications

Geoff Twardokus
Rochester Institute of Technology
geoff.twardokus@mail.rit.edu

Nina Bindel
Sandbox AQ
nina.bindel@sandboxaq.com

Hanif Rahbari
Rochester Institute of Technology
hanif.rahbari@rit.edu

Sarah McCarthy
University of Waterloo
sarah.mccarthy@uwaterloo.ca

***Abstract*—We tackle the atypical challenge of supporting post-quantum cryptography (PQC) and its significant overhead in safety-critical vehicle-to-vehicle (V2V) communications, dealing with strict overhead and latency restrictions within the limited radio spectrum for V2V. For example, we show that the current use of spectrum to support signature verification in V2V makes it nearly impossible to adopt PQC. Accordingly, we propose a scheduling technique for message signing certificate transmissions (which we find are currently up to 93% redundant) that learns to adaptively reduce the use of radio spectrum. In combination, we design the first integration of PQC and V2V, which satisfies the above stringent constraints given the available spectrum. Specifically, we analyze the three PQ signature algorithms selected for standardization by NIST, as well as XMSS (RFC 8391), and propose a *Partially Hybrid* authentication protocol—a tailored fusion of classical cryptography and PQC—for use in the V2V ecosystem during the nascent transition period we outline towards fully PQ V2V. Our provably secure protocol efficiently balances security and performance, as demonstrated experimentally with software-defined radios (USRPs), commercial V2V devices, and road traffic and V2V simulators. We show our joint transmission scheduling optimization and *Partially Hybrid* design are scalable and reliable under realistic conditions, adding a negligible average delay (0.39 ms per message) against the current state-of-the-art.**

I. Introduction

Connected vehicle (CV) technologies, integral to emerging intelligent transportation systems, are among the safety requirements of advanced driver-assistance and, ultimately, autonomous driving systems [1]. CV technologies are proliferating globally under the umbrella of vehicle-to-everything (V2X) communication, wherein vehicle-to-vehicle (V2V) has the potential to drastically reduce serious roadway collisions [2] and, at the same time, enhance transportation system efficiency (reduce travel time, pollution, etc.). It requires each vehicle to regularly broadcast safety messages containing travel data (location, heading, etc.) to facilitate proactive movement coordination, such as collision avoidance, among vehicles. In some circumstances (e.g., non-line-of-sight scenarios), V2V messages could provide a critical chance to avert a collision [3].

Duly, V2V modules are already installed in thousands of vehicles on the road [4] and their adoption rate continues to accelerate; e.g., Ford and other major automakers just received regulatory approval for immediate deployment of V2V in new vehicles sold in certain U.S. states [5]. This momentum aligns with ongoing standardization efforts for expanded use cases of V2V (platooning, sensor data sharing, 3D mapping, etc.) to enhance safety, efficiency, and autonomy [6], [7]—IEEE has overhauled its Dedicated Short Range Communications (DSRC) protocol for V2V [8] and 3GPP continues to prioritize its alternative Cellular V2X (C-V2X) protocol in upcoming releases towards 6G [9]. Simultaneously, autonomous vehicles are improving [10]; autonomous driving systems are expected to be available with moderate price premium by the 2040s [11].

With the accelerating deployment of V2V, the safety of connected vehicles is increasingly put in jeopardy by the likelihood of adversaries gaining access to quantum computers within the lifetime of vehicles being sold today (12–15 years [12]). These vehicles may be on the roads until 2039 or beyond, and many experts believe there is a 50% or greater chance that a quantum computer powerful enough to break classical cryptography will be developed by then [13]. Meanwhile, the latest IEEE 1609.2 and 1609.2.1 standards for CV security [14], [15], which describe protocols for certificate-based authentication of broadcast V2V messages, rely solely on elliptic curve cryptography (ECC), and thus will not be effective against adversaries equipped with large quantum computers. In particular, the Elliptic Curve Digital Signature Algorithm (ECDSA) that is used for signing V2V safety messages to protect against spoofing, alteration, and replay attacks in both C-V2X and DSRC systems will be extremely vulnerable. An attacker who uses a quantum computer to forge valid ECDSA signatures on those messages could execute a variety of devastating attacks; e.g., sending fake emergency brake activation notifications in a fog bank or heavy precipitation (limited visibility) to cause vehicles to react by suddenly stopping or slowing down, resulting in anything from a severe traffic jam to a "pile-up" chain of rear-end collisions with innumerable injuries and, potentially, fatalities. So, guaranteeing the integrity and authenticity of V2V messages is paramount.

Once a vehicle is on the road, the hardware security module (HSM) it uses to store and manage classical cryptographic primitives cannot easily be disenrolled from the V2V system, adapted to new cryptographic primitives via over-the-air updates, or, due to poor consumer response rates [16], replaced via manufacturer recall once those attacks become possible. This means vehicles lack *crypto agility* [17]. With the lifetimes of new vehicles overlapping the possible realization of a quantum threat, it is critical to begin taking steps today towards securing connected vehicles against quantum attacks that can break ECDSA. Based on analysis we perform, though, it turns

Network and Distributed System Security (NDSS) Symposium 2024
26 February - 1 March 2024, San Diego, CA, USA
ISBN 1-891562-93-2
https://dx.doi.org/10.14722/ndss.2024.24267
www.ndss-symposium.org

out the problem is greater than simply manufacturing new vehicles with post-quantum (PQ) support (even once PQ HSMs are available). This is largely because the PQ algorithms selected for standardization by the National Institute of Standards and Technology (NIST) cannot be directly used in lieu of ECDSA under the constraints of current V2V communication standards.

In a V2V system, all vehicles must broadcast ten or more safety messages per second over shared and narrow radio spectrum. This necessitates each vehicle minimizing its transmission length and, by extension, the size of each message, to accommodate more nearby vehicles in this safety system—-maximizing *system capacity*. Additionally, the safety-critical nature and the potentially high volume of incoming messages mean that each must be processed within a few milliseconds of arrival, constraining signature *verification time*. A practical integration of post-quantum cryptography (PQC) with its overhead into V2V must not only support the above requirements by (1) complying with practical constraints on frame size and latency, it must also (2) support a generalizable scheme that is not bound to certain PQ algorithms and (3) facilitate *backwards compatibility* with those vehicles that do not support PQC.

We reveal that the need to start transitioning V2V towards quantum resistance and the severe constraints of current V2V standards require carefully enhancing the use of radio spectrum and prioritizing security guarantees based on how we expect the quantum threat to evolve over time. To this end, we first formulate a timeline for transitioning V2V to a quantum-secure ecosystem over the next few decades, emphasizing the requirements for a near-term solution that will provide *adequate* resistance to quantum attacks given today's hardware and regulations, and providing insights for a more holistic PQ paradigm for next-generation wireless technologies like 6G. Since the high overhead of PQC cannot be accommodated in the near term, we deduce it is necessary to deviate from industry standards in order to more efficiently use the allocated radio spectrum, inspiring our development of a machine-learning technique to opportunistically reduce redundant transmissions and make room for PQC. With that, we devise a partially *hybrid* authentication protocol for V2V that carefully combines classical and PQ cryptography while meeting the critical security guarantees for the near term in a transition to a fully quantum-secure future. Hybrid designs, supported by NIST [17], have previously been explored outside of V2V but consider looser size or latency constraints, as shown in Fig. 1. Moreover, we focus on the particular challenges of *authenticating* V2V message broadcasts over noisy wireless channels, whereas existing hybrid designs in embedded or vehicular systems tend to focus on confidentiality or access control [18], [19], [20], [21], [22], or else consider reliable, wired channels with bidirectional communication [23], [24], [25], [26], [27].

Contributions—We show analytically and experimentally that by combining our novel spectrum optimization technique and hybrid authentication protocol, PQC—despite its apparent incompatibility with V2V due to large signatures and keys—can in fact be integrated with IEEE 1609.2 and other V2V standards to kick-start the transition to a quantum-secure CV ecosystem. Specifically, we make the following contributions:

Roadmap for a PQ Transition in V2V: We show that simply adopting a hybrid design using any of the PQ signature schemes selected for NIST standardization [28] or the alternative eXtended Merkle Signature Scheme (XMSS) [29] is impossible under the constraints of V2V protocols. We analyze advancements in quantum computing and juxtapose them against the security requirements of V2V, synthesizing a timeline that quantifies exactly the quantum resistance that is needed to match likely threats over the next 11–15 years.

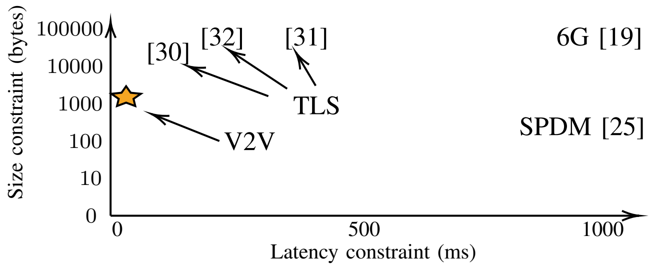

Fig. 1. Comparison of PQC integration constraints in related domains/works.

AI-Enabled V2V Spectrum Optimization: We are the first to reveal that current V2V security standards require vehicles to transmit too much redundant information; in particular, we show that more than 90% of certificate transmissions are redundant (i.e., the receivers already possess a message-signing certificate being shared). In response, we enable vehicles to function as intelligent agents who learn their environment and adjust their security postures by tuning certificate transmission intervals so as to more efficiently utilize the spectrum, improve transmission reliability, and even allow *signing some messages with PQC*. We re-purpose the IEEE 1609.2 peer-to-peer certificate sharing protocol to compensate for the increased intervals (only if needed) and to retain certificate dissemination latency.

Hybrid Design for V2V Authentication: We devise and instantiate a practical *Partially Hybrid* authentication protocol, the first hybrid design for V2V, that meets all of the unique requirements above to neatly integrate PQ into the 1609.2 security framework. Our design is partially hybrid because we utilize PQ signatures specifically to authenticate and protect the integrity of ECDSA message signing *keys*, while continuing to rely predominantly (see above) on ECDSA signatures to protect individual V2V messages. This provides robust protection against quantum forgery attacks in the near term, in which quantum computers may be able to break ECDSA within the one-week validity period of a pseudonym certificate but not within a (reduced) five-minute validity window of the signing key. We further define a threat landscape and introduce a novel security definition (i-unforgeability) for V2V protocols to formally prove our *Partially Hybrid* design, and the current IEEE 1609.2 protocol, are secure with respect to this definition.

Roadway Experiments and a Testbed for PQ-V2V: We evaluate the real-world practicality of our hybrid protocol against the above PQC constraints in V2V using a diverse set of hardware devices, along with open-source software we developed (see our artifacts in Appendix E), which we collectively present as a testbed for PQ V2V dubbed *PQ-V2Verifier*. In *PQ-V2Verifier*, we use a combination of software-defined radios (SDRs) and state-of-the-art commercial devices (Cohda MK6 [33]) with industry-standard ARMv8 V2V chipsets. Through experiments, including outdoor experiments with SDRs mounted in real vehicles on real roadways and extensive simulations using a custom PQC module we built for the VEINS simulator [34], we evaluate and confirm the scalability, reliability, and practicality of our PQ solution. We ultimately demonstrate that adopting our elaborated techniques adds an average of only 0.39 ms of additional delay to each V2V message compared with current standards and systems while significantly improving resistance against quantum attacks.

Scope—We focus on the authenticity and integrity of V2V safety messages against quantum threats in DSRC with insights for future C-V2X designs (we will show PQC in C-V2X is currently a non-starter). Confidentiality is out of our scope as safety messages are never encrypted—they are intentionally *not* confidential. As certificate management and distribution are much broader problems, we consider them only when they directly affect this work (with the exception of peer-to-peer certificate distribution, discussed in detail in Section IV).

Paper Organization—After providing necessary background in Section II, we establish the need for phased hybrid designs based on the evolving constraints of V2V over time in Section III. Following that, we present our spectrum optimization technique (Section IV) and our threat model and *Partially Hybrid* design, including suitable PQ instantiations and informal discussion of its security (Section V). We formally define V2V protocols and their security, and prove our scheme is secure, in Section VI. Our experimental evaluations are discussed in Section VII, and we conclude with related work and future directions in Sections VIII and IX, respectively.

II. A Primer on V2V Security

To achieve the safety and efficiency benefits of V2V, each vehicle broadcasts a digitally signed basic safety message (BSM) at least once every 100 ms. Each BSM contains motion and position information to enable other vehicles to coordinate their movements. Every BSM is signed and packed, along with security information needed for verification, into a Secure Protocol Data Unit (SPDU) as shown in Fig. 2. A frame containing the SPDU is then broadcast using DSRC [8] or C-V2X [35], two similarly decentralized protocols based on different communication technologies. Below, we describe the V2V security and communication protocols.

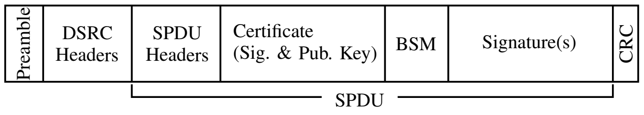

Fig. 2. SPDU structure in a physical-layer frame of DSRC.

A. V2V Security Standards

Security requirements and services for both DSRC and C-V2X are most recently defined in IEEE 1609.2-2022 [14] and IEEE 1609.2.1-2022 [15]. Among other things, 1609.2 specifies asymmetric cryptographic mechanisms and algorithms to securely exchange BSMs, while 1609.2.1 specifies certificate management and revocation requirements for vehicles. Of particular relevance, ECDSA is de facto mandated[1] to generate signatures (using either 256-bit or 384-bit elliptic curves specified in [14]). Beyond signatures, IEEE 1609.2 uses compact *pseudonym* certificates—in which the permanent user identity is replaced with a cryptographically unlinkable, ephemeral identifier—to protect the integrity of the public signature verification keys included in SPDU. Pseudonym

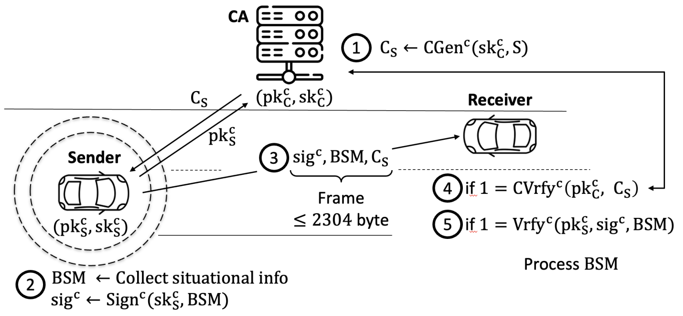
CA
① $C_S \leftarrow \mathrm{CGen}^c(sk_C^c, S)$
C_S
(pk_C^c, sk_C^c)
Receiver
pk_S^c
Sender
③ sig^c, BSM, C_S
Frame
≤ 2304 byte
④ if $1 = \mathrm{CVrfy}^c(pk_C^c, C_S)$
⑤ if $1 = \mathrm{Vrfy}^c(pk_S^c, sig^c, \mathrm{BSM})$
(pk_S^c, sk_S^c)
Process BSM
② BSM ← Collect situational info
$sig^c \leftarrow \mathrm{Sign}^c(sk_S^c, \mathrm{BSM})$

Fig. 3. Current V2V security protocol: (1) Issuer generates a pseudonym certificate over the vehicle's ECDSA key pk_S^c. (2) The vehicle signs a BSM using ECDSA. (3) The frame containing that BSM is broadcast. (4) The receiving vehicle verifies the pseudonym certificate using a certificate chain (potentially with P2PCD). (5) If the signature is valid, the BSM is processed.

certificates are typically rotated every five minutes[2] while each one is (currently) valid for at most one week [36], striking a balance between privacy and efficiency. For revocation, a single entry on a certificate revocation list can be used to efficiently revoke a large number of pseudonym certificates under 1609.2.1 [15]. It has been shown that this mechanism can be adapted to support PQ certificate revocation in V2X [37]. Fig. 3 illustrates secure V2V communication under 1609.2.

The 1609.2 standard defines both explicit and implicit certificates. Each *explicit* certificate within a SPDU must contain a complete verification key and a signature over it by an issuer (e.g., a Certificate Authority (CA)), whereas an *implicit* certificate includes only a shorter *reconstruction value* from which the complete verification key can be derived using a trusted root certificate. Implicit pseudonym certificates generated using the classical scheme in IEEE 1609.2 (Elliptic Curve Qu-Vanstone (ECQV) [38]) help minimize SPDU size: under IEEE 1609.2, one SPDU is at most 226 bytes using implicit vs. 330 bytes using explicit certificates. Since PQ implicit certificates that are smaller than PQ explicit ones have not yet been devised, however, we consider only explicit certificates in this paper.

Under current industry standards [39], a vehicle typically includes its full pseudonym certificate only in every fifth SPDU and transmits a hash of that certificate in the other 80% of messages. This minimizes the number of large frames and consequently maximizes system capacity (also see Section III). From this concept of delaying verification of several (here, up to 4) BSMs before one arrives bearing the certificate required to verify them all, we optimize in our designs how often a complete certificate must be transmitted (e.g., over the course of every 500 ms interval, across the transmission of 5 BSMs). We call this period a *certificate transmission cycle* and incorporate it as a critical element of our *fragmentation* method (in particular, fragmentation of large certificates that can then be transmitted using several SPDUs, see Section V-B).

IEEE 1609.2 further defines the peer-to-peer certificate distribution (P2PCD) protocol to support verification of pseudonym certificates. When a pseudonym certificate is being verified, its issuer's certificate must also be verified, and so on until this *certificate chain* is verified all the way up to a self-signed trusted root certificate. During this process, a vehicle

[1]The SM2 algorithm is also permitted [14, Clause 5.3.1.1], but it is very similar to ECDSA and seldom used outside of China.

[2]IEEE 1609.2 does not define this, but five minutes is a common estimate [36].

will generate a P2PCD *learning request* if it encounters an unknown certificate and attach that request to its next outgoing SPDU [14, § 8.1]. Any vehicle receiving this request that has knowledge of the requested certificate then generates a *learning response*. After a wait time randomly chosen from the discrete uniform distribution between $0-250$ ms, if the vehicle has not heard at least 3 other vehicles broadcast a learning response containing that certificate, it broadcasts its learning response [14]. This process works well for ECDSA, whose certificates are small enough that a learning response can fit within a single payload, but breaks down when certain PQ algorithms are used instead. As we discuss in Section V-D, this alone excludes some PQ algorithms as P2PCD is mandatory.

B. V2V Communication Technologies

DSRC and C-V2X are the two major V2V protocols used around the world defined for the physical and Media Access Control (MAC) layers. DSRC, an IEEE 802.11 protocol tailored for the high-mobility V2V environment, is the de facto standard in Europe [40], used in more than $100,000$ vehicles on the roads in Japan [41], and, as recently as 2020 [42]), the majority of V2V-equipped vehicles in the U.S. We focus on DSRC to develop our design due to its current dominance in several places in the world and the incompatibility of 5G C-V2X with current PQC, as we elaborate below.

A significant obstacle to integrating PQC into V2V is the maximum size of frame payloads. Under DSRC, payloads are capped at $2,304$ bytes regardless of data rate (i.e., modulation and coding scheme) or channel bandwidth [43, Table 9-25]. This is already a limiting constraint on any PQ-V2V design, but it is still significantly higher than the limit in C-V2X, a protocol based on 4G and 5G cellular technology. In C-V2X, the maximum payload size (considering only *practical* data rates) in a standard 10 MHz channel is a mere 437 bytes [44, Table A.8.3-1]. As this is insufficient to contain even a PQ signature—let alone signature and public key, together on the order of thousands of bytes for any of the three PQ algorithms standardized by NIST (see Section III)—the current iteration of 5G C-V2X cannot practically support PQC. Adapting C-V2X to suit PQC, or vice versa, would require significant changes in the design of C-V2X, FCC spectrum allocations, and/or standard PQ algorithms, which are out of scope for this paper.

III. Necessity of a Phased Hybrid Design

In contrast to protocols like Transport Layer Security (TLS), the strict frame size constraint in DSRC (see Fig. 1) rules out simply replacing ECDSA or using it alongside PQ alternatives. To illustrate that, we consider the three PQ signature algorithms selected for standardization by NIST, i.e., Falcon, Dilithium, and Sphincs$^+$ [45], as well as XMSS [46], whose relatively small keys and signatures make it the most promising algorithm for V2V among the hash-based alternative PQ algorithms recommended by NIST in [29].

Specifically, we choose the following instantiations, corresponding to NIST's security Level 1 [47]: `Falcon-512`, `Dilithium-II`, and `Sphincs`$^+$`-128s`. Level 1 PQ algorithms provide at least the same security against quantum adversaries (128 bits) as the 256-bit elliptic curves currently used for ECDSA in V2V. Therefore, a Level 1 PQ algorithm

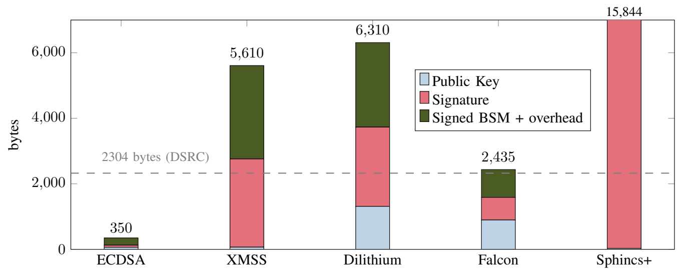

Fig. 4. Frame size (ECDSA and PQC) vs. DSRC payload constraint.

is appropriate to translate current security guarantees into the quantum era. For XMSS, it was necessary to use the Level 5 instantiation `XMSS-SHA2_16_256` because it can create at least $3,000$ unforgeable signatures (the minimum required to sign a BSM every 100 ms throughout each 5-minute period when we use one pseudonym certificate) from a single key. Proposed instantiations at lower security levels cannot meet this requirement; e.g, a proposed Level 3 instantiation can only generate $2^{10} = 1024$ unforgeable signatures [29]. We note that Level 5 instantiations of XMSS have no significant runtime penalty compared to Level 1 instantiations [48], so any impact on our experimental results is minimal.

Pure-PQ: Eventually, a fully PQ ecosystem will be desired; however, directly replacing ECDSA with a PQ algorithm towards a *Pure-PQ* design is unfortunately infeasible in current V2V. Considering the sizes of the public key, certificate, and signature of each PQ algorithm, which are plotted in Fig. 4 along with other frame elements shown in Fig. 2, it is apparent that any of the *Pure-PQ* variants would result in frames that exceed the limit of $2,304$ bytes. Signing every message with PQC and spreading a certificate (containing its PQ signature and a public key) across two or more frames could meet the frame size constraint for some algorithms (e.g., Falcon); however, it would make every frame hundreds of bytes larger, throttling the capacity of the system (see Section VII).

Fully Hybrid: To maintain current classical security guarantees and simultaneously add security against quantum adversaries, standardization agencies including NIST recommend (or at least support) using dual signatures [17]. In such a hypothetical design, certificates would be a concatenation of an ECDSA and a PQ certificate, and each BSM would be authenticated by an ECDSA-PQ dual signature. However, this straightforward approach runs into the same obstacles as the *Pure-PQ* design. Therefore we consider the following alternative *Fully Hybrid* design: The first few SPDUs (with the exact number depending on the instantiation) in a certificate transmission cycle contain BSMs signed using only ECDSA along with fragments of the sender's hybrid certificate. After a receiving vehicle obtains all of the fragments, subsequent messages in that certificate transmission cycle and beyond can be effectively protected with ECDSA-PQ dual signatures until a new pseudonym certificate is rotated in by the sender. Naturally, this additional security guarantee comes at the cost of having to send two certificates and two signatures, increasing the frame size to such an extent that *Fully Hybrid* is only possible using Falcon (and even then, only with significant overhead compared to our *Partially Hybrid* design described in Section V). We provide a pseudo-code description and formal analysis of our *Fully Hybrid* design, the details of the instantiation, and resulting frame sizes in Appendix D.

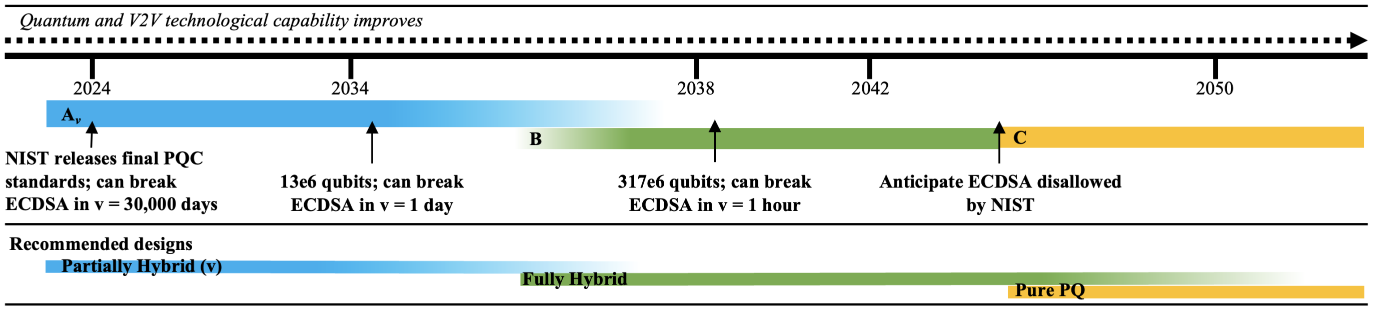
Quantum and V2V technological capability improves
2024
2034
2038
2042
2050
A_v
B
C
NIST releases final PQC standards; can break ECDSA in v = 30,000 days
13e6 qubits; can break ECDSA in v = 1 day
317e6 qubits; can break ECDSA in v = 1 hour
Anticipate ECDSA disallowed by NIST
Recommended designs
Partially Hybrid (v)
Fully Hybrid
Pure PQ

Fig. 5. Estimated timeline of the PQ transition period and how our proposed eras and designs align.

Phased Transition to PQ-V2V. Given the current impracticality of a *Fully Hybrid* design, we need a *Partially Hybrid* design that can be used to launch a multi-year *transition period* to move V2V towards a quantum-secure future. To this end, we first outline the transition period we envision for V2V, breaking it down into several phases ("eras") based on our analysis of trends and expert projections for the development of quantum computers in the coming years (see Fig. 5).

We define era A_v as starting today for new vehicles and lasting until ECDSA can be broken in fewer than v hours, where v is a variable dependent on quantum and V2V technology advances. During era A_v, ECDSA signatures remain secure against quantum attackers as long as signing certificates are valid for fewer than v hours. We construct our *Partially Hybrid* design specifically for era A_v. Extrapolating from IBM's 2019–2023 quantum computing data and their projections through 2026 [49] (assuming the current exponential growth of quantum computing power), and considering state-of-the-art estimates of the number of necessary qubits [50], we expect ECDSA will be breakable in v=1 hour as early as 2038.

The second era (B), which may overlap with A_v, will begin once new hardware and improved communication protocols allow new vehicles to rely on PQ authentication in a *Fully Hybrid* design (while older vehicles on the road cannot). Although it is difficult to predict when embedded PQ hardware will be widely available (and even harder to predict developments in communication protocols and spectrum allocation), companies are already beginning to market PQ co-processors for embedded devices [51] and Hardware Security Modules [52], so we hypothesize era B will begin around early to mid 2030s. The last transition (to era C) will occur when ECDSA is disallowed, which we expect $\sim$7 years after ECDSA is broken[3].

Our *Partially Hybrid* design is carefully tailored for the most urgent period, era A_v. We argue that this era is already underway and will run for the next $\sim$14 years, hence we assume *current* hardware and V2V protocol constraints in our design. Given the inherent unpredictability of more distant technological advances, we refrain from presenting designs for the other eras (e.g., a *Pure-PQ* design for era C) as that would be premature. A *key contribution* of this paper is demonstrating that PQC, despite its apparent incompatibility with V2V communication (due to large signatures and keys), can in fact be integrated with IEEE 1609.2 for use during the A_v era by combining our novel *Partially Hybrid* design and AI-based spectrum optimization, presented next.

[3]For instance, 3DES, the predecessor to AES [53], was broken by the Sweet32 attack [54] in 2016 and NIST will not disallow the algorithm until 2023 [55]. Hence we predict that ECDSA will be disallowed circa 2045.

IV. Spectrum Optimization for V2V Security

To facilitate transmitting the larger frames required for even a *Partially Hybrid* design without impairing communication reliability, it is critical to more efficiently use the limited radio spectrum allocated for V2V. Here, we investigate the excessive transmission of redundant information under current security standards and leverage CV proximity awareness to optimize spectrum usage. Further, the conserved spectrum could allow signing some critical BSMs with PQC, enhancing security.

A. Redundant Certificate Transmission in V2V

Despite the limited bandwidth available to V2V safety services (as little as one 10 MHz channel), the only attention that current industry standards for V2V security pay to efficient use of the available spectrum is permitting vehicles to transmit their full certificate only in every fifth SPDU (see Section II-A). Considering the typical 100 ms BSM interval, this means a vehicle broadcasts an SPDU containing its full pseudonym certificate (a "full-certificate SPDU") every 500 ms. We expose here a problem, common to DSRC and C-V2X, hitherto overlooked in prior work and standards: with vehicles often moving in similar directions at varying speeds of usually at most $\sim$130 km/hr (26 m/s), the vast majority of vehicles in range to receive one certificate will still be in range to receive the next one half a second later. While having some repeated transmissions in highly dynamic vehicular environments indeed improves reliability, we hypothesize most full-certificate SPDUs convey certificates to receivers who already have them, thus excessively occupying the channel.

We use simulations in VEINS [34] to quantify how many redundant certificates are transmitted in realistic scenarios. Using a custom V2V security module we built for VEINS (see Section VII-A), we consider an urban environment modeled on Erlangen, Germany, with moderately dense traffic (50 vehicles/km) spread across various roadways moving at average speeds between 80–150 km/hr. We configure all vehicles to transmit a BSM every 100 ms, with a full ECDSA certificate in every fifth SPDU, as per current standards. Each vehicle is set to log the total number of certificates it receives and mark down any received certificates that it already possesses. Over 100 iterations of a 5-minute simulation period, the median ratio of redundant certificates received to total certificates received across all vehicles turns out to be as high as 99.3%. This surprisingly high number clearly demonstrates that there are too many redundant certificate transmissions, suggesting that a longer interval between certificate-bearing SPDUs (as opposed to the current 500 ms) would be a safe option to free significant spectrum to support practical adoption of PQC.

B. Leveraging P2PCD for Faster Certificate Dissemination

While increasing the interval between certificate transmissions would reduce the amount of redundant information transmission over the air, one might ask whether doing so would increase the BSM processing delay (since a BSM containing a certificate *digest* could not be verified until the next full-certificate SPDU is received). To address this, we make use of the existing P2PCD protocol defined in IEEE 1609.2 by extending it from being used only to learn issuer certificates (see Section II-A) to also being used for learning both issuer *and pseudonym certificates*. Put differently, we use P2PCD to allow a vehicle to proactively request another vehicle's pseudonym certificate from peers who already possess it, thus making up for any possible added delays from an increased interval between full-certificate SPDUs transmissions. We note that learning responses are limited in number (see Section II-A) and smaller than typical SPDUs. As a consequence, transmitting more learning responses than in current systems when combined with increased certificate intervals still results in significant conservation of spectrum. As a concomitant benefit, leveraging P2PCD in this way allows a vehicle to obtain its peers' certificates more quickly than in current systems, since pseudonym certificates are shared on demand rather than fixed for distribution every 500 ms, thus reducing the overall delay in certificate distribution and BSM processing in current systems as well as our hybrid design.

Demonstrative Example. Under current industry standards [39], certificate-bearing SPDUs are 330 bytes while a P2PCD learning response (unsigned, per [14]) containing one certificate is only 172 bytes. In an arbitrary five-second window, a vehicle complying with these standards would transmit 10 full-certificate SPDUs, occupying the channel for the time it takes to transmit $10 \times 330 = 3300$ bytes. Alternatively, consider if the vehicle transmits a full-certificate SPDU only once per second, in which case it would send only $5 \times 330 = 1650$ bytes. Then, in the added 500 ms gap, it is possible that a few new vehicles enter communication range, receive a digest-bearing SPDU, and send P2PCD learning requests for the sender's pseudonym certificate. Under IEEE 1609.2, multiple learning requests for the same certificate within a few hundred milliseconds should trigger at most 3 learning responses (give or take, considering possible hidden terminals), for an additional $3 \times 172 = 516$ bytes. Thus, instead of transmitting 3300 bytes of full-certificate SPDUs in 5 seconds, a total of $1650 + 516 = 2166$ bytes would be sent to convey the same certificate, reducing spectrum usage by up to 34% (enough, e.g., to then sign 3 selected BSMs with Falcon). In Section VII, we experimentally demonstrate how our approach bears the results of this example out in practice.

C. AI-Enabled Spectrum Optimization

Having provided our insights, it remains to address exactly how often, and when, each vehicle should transmit a full-certificate SPDU. Since V2V is an extremely dynamic environment, it is intuitive that the optimal interval will not be the same under all conditions, and enumerating the wide variety of possible conditions would be daunting. Instead, we turn the vehicles into *intelligent* agents who leverage machine learning to autonomously adjust that interval based on their observations of the environment, besides their existing capability at the physical layer to dynamically adapt their transmission rate (in bits per second, or bps) based on wireless channel conditions to optimize transmission length.

Machine Learning Model. We use a random forest classifier, as random forests are particularly well-suited for V2V thanks to their relatively light weight, minimal storage space requirements, and logarithmic time predictions once trained. Based on parameters described below, our classifier takes advantage of real-time proximity awareness from the V2V service and predicts the best tuple of full-certificate SPDU interval (in seconds) and data rate (bps) to use under the observed conditions. We trained our model on a diverse set of samples collected from simulations in VEINS across which we varied the density of vehicles, vehicle speed, transmission rate, vehicle paths, etc. We labeled training samples based on their class (i.e., combination of data transmission rate and full-certificate interval) that most significantly reduced frame collisions (and loss) across all vehicles and runs. We consider data rates of 6 and 9 Mbps (the two most reliable data rates allowed in DSRC [56], [43] for unpredictable channel conditions) and certificate transmission intervals of 500, 1000, 2000, and 5000 ms for a total of $2 \times 4 = 8$ classes.

Learning Parameters. Our model incorporates a variety of features based both on the roadway environment (e.g., number of lanes, direction of travel, urban vs. rural) and communication channel metrics like signal-to-interference-and-noise ratio (SINR). Roadway parameters are static, and we assume a vehicle can update these for its current location as it moves, e.g. by using GPS data with pre-loaded street maps. Of greater note, two parameters from the environment turn out to have strong predictive value for our classifier. One is the *density* of vehicles around the transmitter, which we estimate as a moving average of the number of SPDUs received over time (since BSM interval is fixed at 100 ms), and the other one is the *distance* between the transmitter and neighboring vehicles, estimated based on a moving average received signal strength (RSS) for received SPDUs. These are significant factors in determining how heavily the channel is being used as well as how much interference is present, both of which impact the likelihood of frame loss and are therefore critical for selecting an appropriate data rate and transmission interval pair.

We selected final model parameters (e.g., number of trees) via ablation study. Our classifier achieves 94% accuracy on a test set (10% of samples), indicating good performance despite the uncertainty in dynamic vehicular environments. We show in Section VII that it is very effective in practice.

V. Partially Hybrid Authentication for V2V

During the PQ transition era A_{v} (see Fig. 5), strict requirements (e.g., the 2304-byte limit on payload size) are enforced to comply with the DSRC standard and facilitate sharing the limited bandwidth among many vehicles. Motivated by our analysis in Section III, we propose a *Partially Hybrid* design, proving it efficiently provides strong cryptographic protection against quantum adversaries during era A_{v}; specifically, it protects against adversaries who require at least time v to break ECDSA. We include parameter v in our design and assume that it can be gradually attenuated as attacks against ECDSA strengthen (e.g., $1 \leq \mathsf{v} \leq 30,000$ days—see Fig. 5). This is

TABLE I. PQ SECURITY AND VIABILITY OF OUR *Partially Hybrid* COMPARED TO ALTERNATIVE DESIGNS.

Design	Certificate	BSM Auth.	Performance
ECDSA (in use today)	○	○	Acceptable
Pure PQ	◖	-/◖	Unacceptable
Fully Hybrid	●	○/●	Unacceptable
Partially Hybrid	●	○$\rightarrow$ ●	Acceptable

- = none, ○ = ECDSA, ◖ = PQ, ● = ECDSA + PQ

based on current estimates of quantum computing [50], [13] and extrapolated from IBM's efforts [49]. Since we estimate such quantum computers will be built no earlier than $\sim$ 2038, our design allows a grace period to develop new hardware and improve the V2V communication protocols needed for a *Fully Hybrid* design. Our design is standard-compliant and practical, as shown in Section VII, and thus ready to use today.

Our core idea is to continue signing BSMs with classical cryptography (i.e., ECDSA), and with PQC whenever our spectrum optimization allows, while setting the validity period of the ECDSA verification keys to v, significantly reducing it from the current 1 week. We then require the ECDSA verification key in the pseudonym certificate to be signed using ECDSA *and* PQ signatures. Put differently, this design protects the integrity of the ECDSA pseudonym verification key pk_S using dual ECDSA-PQ signatures, as the issuer's keys (and hence the signatures on the pseudonym certificates) are used over much longer time periods and need to be protected against quantum attacks. Our approach of analyzing the quantum powers is inspired by the *quantum annoying* property of [57].

Table I compares our design to alternatives. In this section, we define our threat model, explain how our fragmentation approach accommodates large PQ certificates, (informally) describe our backwards-compatible design, and suggest different instantiations. A formal foundation is given in Section VI.

A. Threat Model

The security goal defined by the IEEE 1609.2 standard is, in part, to secure messages against spoofing and alteration attacks [14, §1.2]. Correspondingly, we assume the attacker's goal is to launch such attacks to make receivers accept fraudulently signed BSMs and/or treat forged certificates (excluding the root) as legitimate, thereby causing traffic delays, collisions, or other disruptive events.

Attacker's Capabilities. We assume the attacker can observe, drop, replay, or delay the sending or processing of legitimately generated and broadcast SPDUs; alter SPDUs, e.g., changing the BSM, changing/ dropping/ adding/ swapping the/a pseudonym certificate; enforce BSM contents that are then legitimately signed and broadcast by the targeted sender; and is unable to acquire more than one pseudonym certificate per pseudonym from CA (as is specified in [14]).

Assumptions. 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 generation and verification is correct and secure. In particular, we assume that CAs are honest, the root certificate cannot be forged, certificates are only generated for legitimate pseudonyms, and invalid certificates are detectable. Furthermore, we assume that all honestly generated SPDUs are verified by the verifier in the *same order* that they have been sent by the signer (handled by lower layers or using signed BSM timestamps) as long as the receiver stays in the transmission range. Moreover, we assume that communication errors during transmission are handled by lower layers.

TABLE II. SELECTED CERTIFICATE FIELDS OF PARTIALLY PQ HYBRID CERTIFICATE C_S; CHANGES TO CLASSICAL CERTIFICATE ARE SHADED.

	Certificate fields	Value	Explanation
	version	4	Version of certificate format
	type	0	Implicit or explicit
	issuer	C	identifies issuer
toBeSigned	id	S	identifies holder (pseudonym)
toBeSigned	validityPeriod	start, duration v	validity period
toBeSigned	verifyKeyIndicator	pk_S^c	public ECDSA key
toBeSigned	PQsignatureAlg	PQ	PQ signature scheme
toBeSigned	others		E.g., crarId, crlSeries, region
	PQ signature	$\mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_S^{\mathsf{pq}}, \mathsf{C}_S)$	PQ signature by issuer
	ECDSA signature	$\mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_S^{\mathsf{c}}, \mathsf{C}_S)$	ECDSA signature by issuer

Quantum Powers. We assume that quantum computers cannot break ECDSA immediately. Instead, we assume for our *Partially Hybrid* design, that a quantum attacker needs $1 \leq \mathsf{v} \leq 24$ hours. This, based on recent findings [50], requires between 13×10^6 (for $\mathsf{v} = 24$ hours) and 317×10^6 (for $\mathsf{v} = 1$ hour) qubits. We refer to Section I for an estimated timeline on the arrival of such quantum computers. We discuss the resulting validity periods of the pseudonym certificates below.

B. Certificate Generation and Fragmentation

We formally define our certificate fragmentation scheme to be able to give a general description of our design, as follows.

Hybrid Certificate. Let C_S denote the hybrid certificate, of which there are several types, indicated by the `version` field. We propose a new `version` that combines an (explicit) ECDSA certificate with a PQ-based certificate over the same ECDSA verification key. This means that the issuer C holds two key pairs: ECDSA keys $(\mathsf{sk}_C^{\mathsf{c}}, \mathsf{pk}_C^{\mathsf{c}})$ and PQ keys $(\mathsf{sk}_C^{\mathsf{pq}}, \mathsf{pk}_C^{\mathsf{pq}})$. We depict our proposed certificate structure including the most important fields in Table II.

Fragmentation. To satisfy size constraints on the frames, we will use *fragmentation*. More concretely, the first few SPDUs include fragments of the sender's hybrid certificate. After all fragments are received, the receiver can verify the integrity of the verification key using ECDSA and PQC, before using the ECDSA key to verify any signatures. Moreover, we define a function $\mathsf{CFrag}_\alpha : \mathsf{C} \rightarrow \{\mathsf{C}_1, ..., \mathsf{C}_\alpha\}$ which fragments a given certificate C into α parts. The number of fragments α is optimized based on the PQ algorithm used such that 1) α is minimal to transmit the entire hybrid certificate as soon as possible, 2) all frames are at most $2,304$ bytes, 3) the size of all frames used to transmit C_S is equal, to decrease the likelihood of frame loss due to large frames.

The inverse $\mathsf{CCons}_\alpha : \{\mathsf{C}_1, ..., \mathsf{C}_\alpha\} \rightarrow \mathsf{C}$ reconstructs a certificate from given fragments. We further define $\mathsf{H} : \{0,1\}^* \rightarrow \{0,1\}^{256}$ to be a hash function.

C. Informal Description

The pseudo-code description of our *Partially Hybrid* design in Fig. 6 uses the following notation. We consider one run of the protocol, which re-occurs every τ messages where τ is the number of SPDUs per certificate transmission cycle (see Section II-A). 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 (Appendix A provides definitions of digital signature schemes and corresponding notation). In addition, we denote the function checking the validity of the pseudonym certificate using the CA's public key by $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{C}_S) \in \{0, 1\}$. The sender's keys are $(\mathsf{pk}^{\mathsf{c}}_S, \mathsf{sk}^{\mathsf{c}}_S)$.

The proposed protocol can be divided into three stages, depending on the frame index $i \in [1, \tau]$, i.e., which of the τ messages in the certificate transmission cycle is the current one. We denote the i-th message by BSM_i and its signature by sig_i, packed into spdu_i. The first stage (lines 3-11) and last stage (lines 25-31) are the same as for the ECDSA-based design conditioned on the Boolean value b_c. This value is 1 if the entire hybrid certificate has been received *and* verified by the receiver within the five minutes that the pseudonym certificate has been used, (i.e., b_c is set to 1 in line 18). If $b_c = 0$, processing of the BSM is delayed (using function $\mathsf{Delay}(\mathsf{BSM})$ in line 11) until the hybrid certificate has been verified, as permitted implicitly under V2V standards (see Section II-A). In our design, and as long as underlying communication protocols continue to impose current strict frame size limits, we can set a timer to $\tau \times 100\,\text{ms}$ to obtain the hybrid certificate needed to verify and process the BSM.

The first and second part (lines 3-24) are used to communicate all fragments of the hybrid certificate. Then, the hybrid certificate is verified (line 16), which means (at minimum) the certificate validity period is checked and the PQ and ECDSA signatures are verified. If the certificate is valid, the verification key $\mathsf{pk}^{\mathsf{c}}_S$ is extracted (line 17). It is important to note that if a signature verification fails (for the certificate chain or on the BSM itself), the BSM will be discarded. This is indicated by *Abort* in our pseudo-code, and follows the reasoning in [58].

Our design will cease to be cryptographically secure once ECDSA can be broken within the validity period v (even though non-cryptographic methods can still be invoked to detect an abnormal BSM). However, for the foreseeable future it allows adapting agily to changes in the state-of-the-art in quantum computers. Namely, v can be decreased during certificate generation (therefore, more pseudonym certificates need to be generated as discussed below) and certificate verification. These changes would be in software only and could be easily made via over-the-air updates. However, to keep the pseudo-code as simple as possible, we omit this detail in Fig. 6.

Informal Security Analysis. There are essentially two attack vectors: (1) A quantum adversary could *forge a signature on a BSM*. The attacker sees a single key "in use" for a total of only $2 \cdot 5 = 10$ to $14 \cdot 5 = 70$ minutes, depending on v. Moreover, the attacker must generate the forgery within the validity period v. Since this would require quantum computers with with a large number of qubits, quantum adversaries during era A_{v} (see Fig. 5) will not be more powerful than classical ones during such an attack. (2) A quantum adversary could attempt to *forge the pseudonym certificate* generated by a CA. To prevent this, we add a PQ signature over the ECDSA public key, resulting in a hybrid certificate C_S. Assuming at least one signature (PQ or ECDSA) is unforgeable, this attack fails.

We note CVs use several non-cryptographic safeguards to prevent BSMs forgery when certificates have not been verified yet. For example, threat detection services compare BSMs from a given vehicle against its recent BSMs to detect anomalies, so a forgery would be limited in what it could plausibly claim (e.g., a BSM claiming a vehicle is stopped 100ms after its previous (legitimate) BSM reported moving at 60 km/hr would be clearly implausible). Further, on-board sensors like LiDAR or cameras corroborate BSMs; e.g., a front-facing camera would reveal that no vehicle is actually braking ahead despite what a forged BSM might report.

```
1 :  Sender $S$                                          Receiver $R$
2 :  $h \leftarrow \mathsf{H}(\mathsf{C}_S)$
3 :  for $i = 1, \ldots, \alpha - 1$ :
4 :  $\mathsf{sig}_i \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}^{\mathsf{c}}_S, \mathsf{BSM}_i)$
5 :  $\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}^{\mathsf{c}}_i, h, \mathsf{C}_i)$
6 :                      $\xrightarrow{\mathsf{spdu}_i}$ if $b_c = 1$ :
7 :                          if $h = \mathsf{H}(\mathsf{C}_S)$
8 :                              if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}^{\mathsf{c}}_S, \mathsf{sig}_i, \mathsf{BSM}_i) = 1$ :
9 :                                  $\mathsf{Process}(\mathsf{BSM}_i)$
10 :                         Abort
11 :                     $\mathsf{Delay}(\mathsf{BSM}_i)$
12 : $\mathsf{sig}_\alpha \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}^{\mathsf{c}}_S, \mathsf{BSM}_\alpha)$
13 : $\mathsf{spdu}_\alpha \leftarrow (\mathsf{BSM}_\alpha, \mathsf{sig}_\alpha, h, \mathsf{C}_\alpha)$
14 :                     $\xrightarrow{\mathsf{spdu}_\alpha}$ if $b_c = 0$ :
15 :                         $\mathsf{C}_S \leftarrow \mathsf{CCons}(\mathsf{C}_1, \ldots, \mathsf{C}_\alpha)$
16 :                         if $\mathsf{CVrfy}((\mathsf{pk}^{\mathsf{pq}}_C, \mathsf{pk}^{\mathsf{c}}_C), \mathsf{C}_S) = 1$ :
17 :                             $\mathsf{pk}^{\mathsf{c}}_S \leftarrow \mathsf{C}_S$ # extract
18 :                             $b_c \leftarrow 1$
19 :                             $\mathsf{Process}(\mathsf{BSM}_1, \ldots, \mathsf{BSM}_\alpha)$
20 :                         Abort
21 :                     if $h = \mathsf{H}(\mathsf{C}_S)$ :
22 :                         if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}^{\mathsf{c}}_S, \mathsf{sig}_\alpha, \mathsf{BSM}_i) = 1$ :
23 :                             $\mathsf{Process}(\mathsf{BSM}_\alpha)$
24 :                     Abort
25 : for $i = \alpha + 1, \ldots, \tau$ :
26 : $\mathsf{sig}_i \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{BSM}_i, \mathsf{sk}^{\mathsf{c}}_S)$
27 : $\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}_i, h)$
28 :                     $\xrightarrow{\mathsf{spdu}_i}$ if $h = \mathsf{H}(\mathsf{C}_S)$
29 :                         if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}^{\mathsf{c}}_S, \mathsf{sig}_i, \mathsf{BSM}_i) = 1$ :
30 :                             $\mathsf{Process}(\mathsf{BSM}_i)$
31 :                     Abort
```

Fig. 6. Pseudo-code description of the *Partially Hybrid* design to be repeated every τ BSMs; $b_c \leftarrow 0$ at the beginning of a new 5-minute window; if $b_c = 1$, the receiver knows C_S; PQ values and operations are shaded.

Viability. Current practice is to rotate the use of 20 pseudonym certificates throughout one week, with each used at most $3,000$ times per each five-minute rotation [59]. This means every certificate is used in at most 100 rotations (adding up to at most 300,000 signatures) for $\mathsf{v} = 1$ week. Since our *Partially Hybrid* design is only secure under the assumption that the pseudonym certificates are valid for at most one day, we advocate for increasing the number of certificates per week. The flexibility of our design and over-the-air updates allow decreasing the validity period v from one day to a few hours, likely even less. While decreasing the validity period toward the 5-minute (12-minute in ETSI standard [60]) absolute minimum would be even more conservative, using a certificate only once overall or keeping it valid for less than one hour might lead to practical complications [59], e.g., in defining the overlapping duration between rollovers (currently one hour) or related to managing a larger volume of certificates. If these limitations of current V2V protocols are resolved in the future (as ETSI is proposing to reduce the overlap to only a few minutes), v can be reduced further, extending the secure lifetime our *Partially Hybrid*.

TABLE III. RESULTING SIZES OF FRAMES F_i (IN BYTES) FOR THE *Partially Hybrid* DESIGN; $|\mathsf{C}_S|$ IS THE SIZE OF THE ENTIRE CERTIFICATE.

PQ Scheme	$\|\mathsf{C}_S\|$	α	F_1	F_2	F_3	F_4	F_5	β
Pure ECDSA Design								
-	162	1	330	200	200	200	200	1
Partially Hybrid Design								
Falcon	858	1	1026	204	204	204	204	2
Dilithium	2588	2	1462	1462	204	204	204	4
Sphincs$^+$	8024	4	2174	2174	2174	2174	204	8
XMSS	2860	2	1598	1598	204	204	204	3

Concretely, for $\mathsf{v} = 1$ day (resp., $\mathsf{v} = 2$ hours) the number of pseudonym certificates should be increased to at most 140 per week (resp., 1680), assuming[4] at most 20 different certificates used during v. While up to 1680 (in a worst-case scenario) represents a large increase in the required number of certificates per week, using the butterfly key technique already established in IEEE 1609.2.1 that allows a vehicle to obtain up to 2^{128} certificates per single request to a CA [15] and can be constructed using PQC [37], renders the overhead manageable. Another way to offload the computational burden of the CA could be batch signing, as recently resurrected for various PQ applications [61].

Backwards-Compatibility. The *Partially Hybrid* design can be deployed in a backwards-compatible manner by using software updates (installed over the air, during routine maintenance, etc.) to inform receivers about the hybrid certificate structure and allow those vehicles without PQ-capable cryptographic hardware (HSMs) to ignore the PQ certificate components (changing the certificate verification in line 16). To prevent rollback attacks, we assume all vehicles send and expect to receive the hybrid certificates, even if they do not possess the *hardware* capabilities to verify the PQ signature on the hybrid certificate. We note this enforcement only adds security for receivers who actually verify the PQ certificate. The advantage of this approach is that vehicles whose *software* but not *hardware* has been updated will be able to continue to verify ECDSA signatures. This also enables falling back to ECDSA-only, in a crypto-agile way, if any used PQC algorithms turn out to be insecure.

D. Instantiation

Table III presents the frame sizes for each instantiation of the *Partially Hybrid* design, assuming the certificate transmission cycle used under current standards ($\tau = 5$), calculated as follows. We compute the certificate size as $30 + |\mathsf{pk}| + |\mathsf{sig}_{\mathsf{pk}}|$, SPDU size as $|\mathsf{SPDU}| = 24 + |\mathsf{BSM}| + |\mathsf{certificate}| + |\mathsf{sig}_{\mathsf{BSM}}|$, and then the total MAC-layer frame size as $|F| = 40 + |\mathsf{SPDU}|$. Where $\mathsf{sig}_{\mathsf{pk}}$ and $\mathsf{sig}_{\mathsf{BSM}}$ are the signatures over the sender's public key and over an BSM, respectively.

The size of the ECDSA certificate $\mathsf{C}^{\mathsf{c}}_S$ is 162 bytes. Moreover, we compute the total frame size including the ECDSA-signed BSM and fragments of C_S with different instantiations of α (each frame also contains about 40 bytes of overhead for the encoded data structures). For our design to be practical, the number of frames needed to transmit C_S during the certificate transmission cycle (or share it via P2PCD learning response– see Section II-A) is critical, as explained next.

[4]This way, a certificate is only used in a total of at most 14 (resp., 2) rotations on average, potentially increasing the security and privacy since the certificates can be observed less often.

Certificate Transmission Cycle. Certificates based on Falcon, Dilithium, XMSS, and Sphincs$^+$ can be transmitted during the certificate transmission cycle as shown in Table III. For Falcon, C_S includes a Falcon signature over an ECDSA key and is 858 bytes. Therefore, it is not necessary to fragment C_S and one message is sufficient to communicate it (i.e., $\alpha = 1$). Hence, the payload size of the first frame F_1 is 1026 bytes and the payload size of the remaining frames F_2, F_3, F_4, F_5 are 204 bytes each. Dilithium, XMSS, and Sphincs$^+$ instantiations require larger values of α, which translates to more messages being transmitted before the integrity of the ECDSA key can be guaranteed by both classical and PQ signature schemes as well as lower system capacity (see Table IV).

P2PCD. Currently, under ECDSA, a certificate can fit in a single P2PCD learning response; however, we have established that hybrid certificates must be fragmented to meet the DSRC payload size constraint. Therefore, each P2PCD learning response in our design must also be fragmented. Moreover, before *each fragment* is transmitted, the vehicle will wait for a random interval (see Section II-A). In Table III, β indicates the number of frames required to completely convey a P2PCD learning response. The expected value of the uniformly distributed wait time is 125 ms, so *on average* we can expect Sphincs$^+$ will require 1000 ms to communicate all $\beta = 8$ fragments of a single learning response. For Dilithium, we would expect this to take about 500 ms, for XMSS, 375 ms, and 250 ms for Falcon. Based on this, Sphincs$^+$ would usually take longer than the required minimum certificate transmission cycle length (500 ms) to receive a learning response, and so we rule Sphincs$^+$ out. Dilithium, while on the edge of feasibility, is likely also not viable, while XMSS and Falcon should generally be acceptable within the context of P2PCD.

VI. FORMAL ANALYSIS OF THE *Partially Hybrid* DESIGN

We now formally analyze the security guarantees of our *Partially Hybrid* design. To this end, we first define V2V protocols and their security (i-unforgeability).

A. Formal Definition of V2V Protocols

We define V2V protocols in general and based on *certified signature schemes*—a combination of signature schemes and certificates formalized in [62]. We recall the original Definition 6 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* $i \in [1, \tau]$[5]*. Moreover, let* $\mathcal{M}$ *be a message space and* $\mathcal{S}$ *be the set of pseudonyms. A V2V protocol* $\mathcal{P} = (\mathsf{KGen}_C, \mathsf{CGen}(\mathsf{CGen}_C, \mathsf{CGen}_S), \mathsf{SPDUGen}, \mathsf{SPDUVerify})$ *is defined via the following polynomial-time algorithms*

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

$\mathbf{CGen}(\mathbf{CGen}_C, \mathbf{CGen}_S)$ *is an interactive protocol between the sender vehicle* $S \in \mathcal{S}$ *and the CA* C *following Definition 6 in Appendix A, generating a certificate* C_S *over* (S, pk_S)*. We refrain from describing the generation*

[5]We define V2V protocols via τ-SPDU cycles, see Section II-A.

process in detail as it is not finalized for V2V protocols yet [63]. We write $\mathsf{CGen}(\mathsf{sk}_C, S, \mathsf{pk}_C)$ *instead of* $\mathsf{CGen}(\mathsf{CGen}_C(\mathsf{sk}_C), \mathsf{CGen}_S(S, \mathsf{pk}_C))$ *for brevity.*

SPDUGen *corresponds to signing a BSM and generating the SPDU. Upon input an identity* $S \in \mathcal{S}$, *a secret key* sk_S, *a certificate* C_S, *a message* $\mathsf{BSM} \in \mathcal{M}$, *and* $i \in [1, \tau]$, *it returns an SPDU depending on* i, *including a signature* $\mathsf{sig} \leftarrow \mathsf{Sign}_i(\mathsf{sk}_S, \mathsf{BSM})$. *Depending on* $\mathcal{S}_i$, sig *might be the empty string.*

SPDUVerify *returns 0 or 1, upon input* S, pk_S, C_S, pk_C, *a state* st, *and* spdu. *It outputs* 1 *if* spdu *(including* BSM *and* sig*) is valid, and* 0 *otherwise.*

spdu *is valid if* $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{spdu}, \mathsf{st}) = 1$, $\mathsf{Vrfy}_i(\mathsf{pk}_S, \mathsf{sig}, \mathsf{BSM}) = 1$, *and* spdu *is of the correct form*[6] *depending on* i *(which in turn depends on* st*). The function* CVrfy *is not formally defined here, as it suffices to assume it checks all aspects of certificate validation. Depending on the instantiation this might include verifying the issuer's signature, certificate chain, certificate's expiration date, etc. Depending on the internal verification, the state* st *is updated accordingly within* CVrfy. Vrfy_i *follows Definition 6.*

For example, the current ECDSA-only design under [14] and [39] can be seen as a V2V protocol with $\mathcal{S}_1, ..., \mathcal{S}_5$ all being ECDSA and every BSM being signed (i.e., no empty signatures occur). For $i \mod \tau = 1$, spdu_i consists of a BSM, an ECDSA signature, and the pseudonym certificate. All other SPDUs consist of a BSM, 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.

Our notation allows a different signature scheme $\mathcal{S}_i$ for every SPDU in the certificate transmission cycle to cover cases where the first SPDU is not signed (which is allowed by IEEE 1609.2) or to use a hybrid scheme from, say, the second message on, while $\mathcal{S}_1$ is ECDSA as in the *Fully Hybrid* design.

We now describe the *Partially Hybrid* design following Definition 1 with $\mathcal{S}_\mathsf{c} = \mathcal{S}_1 = ... = \mathcal{S}_\tau$ and $\mathcal{S}_\mathsf{pq}$.

KGen$_C$ returns $(\mathsf{pk}_C, \mathsf{sk}_C)$ with $\mathsf{pk}_C = (\mathsf{pk}^\mathsf{c}_C || \mathsf{pk}^\mathsf{pq}_C)$, $\mathsf{sk}_C = (\mathsf{sk}^\mathsf{c}_C || \mathsf{sk}^\mathsf{pq}_C)$, $(\mathsf{sk}^\mathsf{c}_C, \mathsf{pk}^\mathsf{c}_C) \leftarrow \mathsf{KGen}^\mathsf{c}$, $(\mathsf{sk}^\mathsf{pq}_C, \mathsf{pk}^\mathsf{pq}_C) \leftarrow \mathsf{KGen}^\mathsf{pq}$.

CGen generates C_S for a sender key pk^c_S as in Section V-B.

SPDUGen returns (with $\mathsf{sig}^\mathsf{c}_i \leftarrow \mathsf{Sign}^\mathsf{c}(\mathsf{sk}^\mathsf{c}_S, \mathsf{BSM}_i)$ and C_i being certificate fragments)

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

SPDUVerify is defined as in Fig. 6 and text above, distinguishing between cases $i \in [1, \alpha-1]$, $i = \alpha$, and $i \in [\alpha+1, \tau]$.

B. Formal Security Guarantees

We now define the security of V2V protocols, namely i-unforgeability, and present our main theorem that is used to deduce Corollary 1 about the security of our designs. We model the security based on the unforgeability of certified signature schemes [62], with adaptions to V2V protocols. To model the capabilities of the attacker defined in our threat model (Section V-A), we introduce two different signing oracles, $\mathcal{O}_\mathsf{C}$ and $\mathcal{O}_\mathsf{spdu}$, which respectively correspond to generating a pseudonym certificate and SPDUs. Both oracles can be queried adaptively from respective messages spaces. This corresponds to an attacker that is able to enforce certain BSM content but not able to tamper with the sensors, and to request pseudonym certificates. The variable i indicates which SPDU in the certificate transmission cycle cannot be forged. For example, forging an SPDU that uses only ECDSA might be possible for a quantum adversary, while forging a later SPDU that uses a classical-PQ hybrid signatures is not achievable, as per Corollary 3 in Appendix D.

```
Oracle $\mathcal{O}_\mathsf{C}(\mathsf{sk}_C, U, \mathsf{corrupt})$:
  if $S \notin \mathcal{S} \vee (S, \cdot, \cdot, \cdot) \in \mathsf{ListPub}$: return $\perp$
  $((S, \mathsf{pk}_S, \mathsf{C}_S), (S', \mathsf{pk}_{S'}, \mathsf{sk}_{S'}, \mathsf{C}_{S'})) \leftarrow \mathsf{CGen}(\mathsf{sk}_C, S, \mathsf{pk}_C))$
  $\mathsf{ListPub} \leftarrow \mathsf{ListPub} \cup \{(S, \mathsf{pk}_S, \mathsf{C}_S)\}$
  if corrupt $= 0$: $\mathsf{ListSec} \leftarrow \mathsf{ListSec} \cup \{(S', \mathsf{pk}_{S'}, \mathsf{sk}_{S'}, \mathsf{C}_{S'})\}$
  else: #sender $S$ is corrupted
      $\mathsf{ListSec} \leftarrow \mathsf{ListSec} \cup \{(S', \perp, \perp, \perp)\}$
      $\mathsf{ListPub} \leftarrow \mathsf{ListPub} \cup \{(S', \mathsf{pk}_{S'}, \mathsf{sk}_{S'}, \mathsf{C}_{S'})\}$
  $q_\mathsf{C} \leftarrow q_\mathsf{C} + 1$
Oracle $\mathcal{O}_\mathsf{spdu}(S, \mathsf{pk}_S, \mathsf{C}_S, \mathsf{BSM}, i)$:
  if $\mathsf{m} \notin \mathcal{M}_S \vee (S', \perp, \perp, \perp) \in \mathsf{ListSec}$: return $\perp$
  $(S, \mathsf{pk}_S, \mathsf{sk}_S, \mathsf{C}_S) \leftarrow \mathsf{Find}(\mathsf{ListSec}, S, \mathsf{pk}_S, \mathsf{C}_S)$
  $\mathsf{spdu} \leftarrow \mathsf{SPDUGen}(S, \mathsf{sk}_S, \mathsf{C}_S, \mathsf{pk}_C, \mathsf{BSM}, i)$
  $q_\mathsf{spdu} \leftarrow q_\mathsf{spdu} + 1$, $Q_\mathsf{spdu} \leftarrow Q_\mathsf{spdu} \cup (S, \mathsf{pk}_S, \mathsf{C}_S, \mathsf{BSM}, i, \mathsf{spdu})$
  return spdu
```

Fig. 7. Oracles $\mathcal{O}_\mathsf{C}$ and $\mathcal{O}_\mathsf{spdu}$

Definition 2 (V2V i-Unforgeability)**.** *Let* $\mathcal{P} = (\mathsf{KGen}_C, \mathsf{CGen}, \mathsf{SPDUGen}, \mathsf{SPDUVerify})$ *be a V2V protocol. Let* $\mathcal{A}$ *be an efficient (classical/quantum) adversary and* $i \in [1, \tau]$. *We define the advantage of* $\mathcal{A}$ *against the security experiment* $\mathrm{Expt}^{i\text{-}\mathsf{UF}}_\mathcal{P}(\mathcal{A})$ *as*

$$Adv^{i\text{-}\mathsf{UF}}_\mathcal{P}(\mathcal{A}) = \Pr\left[\mathrm{Expt}^{i\text{-}\mathsf{UF}}_\mathcal{P}(\mathcal{A}) = 1\right].$$

We say that $\mathcal{P}$ *is i-unforgeable (*i-UF*) if* $Adv^{i\text{-}\mathsf{UF}}_\mathcal{P}(\mathcal{A})$ *is negligible in the security parameter* λ.

The experiment $\mathrm{Expt}^{i\text{-}\mathsf{UF}}_\mathcal{P}(\mathcal{A})$ *is as follows. Let* $\mathsf{ListPub}$ *and* $\mathsf{ListSec}$ *be lists maintained by the experiment, used to store public and secret information, respectively.* $\mathcal{A}$ *has access to* $\mathsf{ListPub}$ *and to all information sent between parties but not to* $\mathsf{ListSec}$. *At the beginning, set* $q_\mathsf{C} \leftarrow 0, q_\mathsf{spdu} \leftarrow 0$, $Q_\mathsf{spdu} \leftarrow \{\}$. *In addition, the experiment keeps a state* st. $\mathcal{A}$ *is allowed to query oracles* $\mathcal{O}_\mathsf{C}$ *and* $\mathcal{O}_\mathsf{spdu}$ *given in Fig. 7, having the power to corrupt senders. The CA* C *cannot be corrupted, i.e.,* sk_C *is never shared with* $\mathcal{A}$.[7] *Eventually* $\mathcal{A}$ *outputs a forgery* $(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}, \mathsf{spdu}^*)$. *The experiment returns 1 if* $\mathsf{SPDUVerify}(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}, \mathsf{pk}_C, \mathsf{spdu}^*, \mathsf{st}) = 1$ ***and*** *at least one of following conditions is satisfied:* $i)(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}) \in \mathsf{ListPub}$ *and* $(S^*, \mathsf{pk}_{S^*}, \cdot, \mathsf{BSM}^*, \cdot) \notin Q_\mathsf{spdu}$, $ii)(S^*, \mathsf{pk}_{S^*}, \cdot) \notin \mathsf{ListPub}$, *or* $iii)(S, \mathsf{pk}_{S^*}, \mathsf{C}_S) \in \mathsf{ListPub}$ *for* $S \neq S^*$ *with* $(S, \cdot, \cdot) \in \mathsf{ListPub}$.

[6]By "correct form", we mean the SPDU is of the sub-components associated with that value of $i \in [1, \tau]$, as defined in $\mathsf{SPDUGen}$.

[7]This is different from [62, Definition 4.1] and carries potential for extension.

The adversary's winning condition 1 covers alteration attacks, while winning condition 2 covers spoofing attacks. Eavesdropping attacks are considered in our model as we allow the attacker access to $\mathcal{O}_{\mathsf{spdu}}$. Replay attacks are implicitly considered in our model, as they are prevented by checking that the time stamp (which is included in the BSM) is fresh [14] (i.e., that the BSM was sent within the preceding $30s$ [39]). To circumvent this mechanism, the attacker would have to change the timestamp, thereby altering the BSM (which is covered by condition 1). Hence, i-UF covers the security goal defined in IEEE 1609.2 under the assumptions listed in Section V-A.

We deduce the security of our *Partially Hybrid*, the *Fully Hybrid* (see Appendix C), and the original ECDSA-based design (see Appendix C) from the following theorem[8]. Notably, we assume all vehicles have access to issuer certificates *immediately* (i.e., no P2PCD is needed). As discussed previously, this is idealized. Interestingly, this shows a limitation of provable security, as non-cryptographic verification and message delays do not increase *security* but still *safety*.

Theorem 1. *Let $i \in \{1, 2, \ldots, \tau\}$, and $\mathcal{S}_i = (\mathsf{KGen}_i, \mathsf{Sign}_i, \mathsf{Vrfy}_i)$ and $\mathcal{S}_C = (\mathsf{KGen}_C, \mathsf{Sign}_C, \mathsf{Vrfy}_C)$ be (hybrid) signature schemes. In addition, let $\mathcal{P} = (\mathsf{KGen}_C, \mathsf{CGen}, \mathsf{SPDUGen}, \mathsf{SPDUVerify})$ be a V2V protocol using (hybrid) pseudonym certificates, $\mathcal{S}_C$ to sign certificates, and $\mathcal{S}_i$'s to sign SPDUs. Moreover, let $\mathcal{A}$ be a (classical/quantum) adversary against $\mathrm{Expt}^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A})$, making at most q_{C} queries to $\mathcal{O}_{\mathsf{C}}$. Then we can construct adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$*

$$Adv^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A}) \le \frac{1}{q_C} Adv^{\mathsf{eUF}}_{\mathcal{S}_i}(\mathcal{B}_1) + Adv^{\mathsf{eUF}}_{\mathcal{S}_C}(\mathcal{B}_2),$$

where $\mathcal{B}_1$ and $\mathcal{B}_2$ each run in time time$(\mathcal{A})$ plus the time required to respond to $\mathcal{A}$'s queries.

Our proof follows [62, Theorem 4.3] (see Appendix B) with careful handling of the hybrid design. The core idea is to define two events corresponding to the adversary winning under condition 1 or 2. Intuitively, the events can be reduced to breaking the eUF security of $\mathcal{S}_i$ used in the generation of the i-th SPDU or to the eUF security of $\mathcal{S}_C$. As mentioned earlier, analyzing the certificate registration and management (winning condition 3) is out of scope of this paper.

It is important to point out that our security reduction assumes a stronger attacker than there is in reality. Namely, the adversary wins if they can forge the signature on a pseudonym certificate (condition 2). While this is a necessary condition, in practice it is not sufficient. Since certificates are verified through certificate chains, the adversary needs to actually forge their way up to the root certificate instead of just forging the pseudonym certificate. Future analysis might refine our result, based on the certificate verification used. eUF is sufficient for the next corollary; notably, there are security properties beyond unforgeability considered for PQ signatures [64], which we leave for future work.

Corollary 1 (Security of the Partially Hybrid)**.** *Let* H : $\{0,1\}^* \to \{0,1\}^{128}$ *be a second pre-image resistant hash function and $\mathcal{P}$ be the* Partially Hybrid *V2V protocol as described above. Moreover, assume the receiver supports the PQ signature scheme used in $\mathcal{P}$[9]. Suppose $\mathcal{A}$ is an i-UF adversary against $\mathrm{Expt}^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A})$ making at most q_C queries to $\mathcal{O}_{\mathsf{C}}$. Then we can construct adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, and $\mathcal{B}_4$ such that for all i, $Adv^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A})$ is upper bounded by*

$$\frac{1}{q_C} Adv^{\mathsf{eUF}}_{\mathcal{S}_{\mathsf{c}}}(\mathcal{B}_1) + \min(Adv^{\mathsf{eUF}}_{\mathcal{S}_{\mathsf{c}}}(\mathcal{B}_2), Adv^{\mathsf{eUF}}_{\mathcal{S}_{\mathsf{pq}}}(\mathcal{B}_3)) + Adv^{\mathsf{2PR}}_{\mathsf{H}}(\mathcal{B}_4).$$

Proof: We recall that for all i, $\mathcal{S}_i = \mathcal{S}_C = \mathcal{S}_{\mathsf{c}}$. Hence, forging signatures over the BSMs reduces to an ECDSA signature ($\mathcal{B}_1$). To bound the advantage to forge hybrid certificates for each i, note that we only process BSMs after the hybrid certificate has been verified. Since $\mathcal{S}_C$ returns the concatenation of signatures generated with $\mathcal{S}_{\mathsf{c}}$ and $\mathcal{S}_{\mathsf{pq}}$ and a certificate is only valid if both those signatures are valid, forging a certificate reduces to forging a classical and PQ signature over an ECDSA certificate body, (minimum of $\mathcal{B}_2$ and $\mathcal{B}_3$). Moreover, the SPDUs include the hash of, rather than the full, hybrid certificate. So we additionally include the advantage of an adversary finding a second pre-image ($\mathcal{B}_4$). ■

VII. Performance Evaluation

We now demonstrate the combination of our spectrum optimization technique and *Partially Hybrid* design is practical; i.e., that it is (1) scalable, supporting enough vehicles for real-world scenarios; (2) efficient, causing no unacceptable delay in BSM transmission or processing; and (3) reliable.

A. Experimental Setup

Testbed. We conduct indoor and outdoor experiments using *PQ-V2Verifier*, a testbed for V2V security that we developed as an overhaul and extension of *V2Verifier* [65], an open-source testbed that supports transmitting BSMs over the air with SDRs using DSRC and 1609.2-compliant BSM signing and verification using ECDSA. Building *PQ-V2Verifier* required implementing certificate verification functionalities (as per IEEE 1609.2.1) and reworking the existing signature generation and verification scheme to more efficiently handle large volumes of BSMs, along with other improvements. For cryptographic algorithm implementations we used popular, open-source libraries: Botan [66], for ECDSA, Dilithium, and XMSS; and *liboqs* [67], for Falcon and Sphincs$^+$. In our experiments, both in the laboratory and on the road with real vehicles, we run *PQ-V2Verifier* on laptops connected to Universal Software Radio Peripheral (USRP) SDRs to emulate real V2V devices. Further, we use real V2V devices with Qualcomm chipsets for some experiments (e.g., to evaluate how quickly PQ algorithms run on V2V processors) by deploying relevant modules of *PQ-V2Verifier*, cross-compiled for ARMv8 architecture, on those devices. We note that our PQC implementations are not fully optimized for ARMv8 (e.g., with NEON instructions); however, our objective is to establish the viability of our work and not to perform a scientific benchmarking exercise, so this is acceptable (though our results may not match formal benchmarking reports, e.g., [68]).

[8]Appendix A defines the advantage $\mathrm{Adv}^{\mathsf{eUF}}_{\mathcal{S}}(\mathcal{A})$ of $\mathcal{A}$ in breaking the existential unforgeability under chosen message attacks (eUF) of $\mathcal{S}$ and the advantage of $\mathcal{A}$ breaking second pre-image resistance $\mathrm{Adv}^{\mathsf{2PR}}_{\mathsf{H}}(\mathcal{A})$ of H.

[9]If the receiver does not support PQ, the security is the same as the ECDSA-based design, see Corollary 2.

TABLE IV. Impact of maximum frame duration (ms) on system capacity ($v_{\max}$) for *Partially Hybrid* vs. Pure ECDSA.

Design	F_1	F_2	F_3	F_4	F_5	$v_{\max}$
Pure ECDSA	0.880	0.533	0.533	0.533	0.533	165
Partially Hybrid						
Falcon	2.736	0.544	0.544	0.544	0.544	101
Dilithium	3.899	3.899	0.544	0.544	0.544	53
Sphincs$^+$	5.797	5.797	5.797	5.797	0.544	21
XMSS	4.261	4.261	0.544	0.544	0.544	49

Simulations. We use VEINS [34], a well-known combination vehicle traffic and DSRC simulator. VEINS models real-world challenges of wireless communication such as noise, interference, fading channels, etc., allowing us to evaluate performance under realistic conditions. However, VEINS does not natively support any V2V security protocols, much less PQC. Therefore, we developed a custom VEINS module, PQ-VEINS, that implements IEEE 1609.2 security (i.e., ECDSA signatures and certificates), our *Partially Hybrid* design instantiated for Falcon (the only PQ algorithm we find to be viable), and our AI spectrum optimization technique from Section IV. All of our simulations consider an urban environment (the most challenging scenario from a communications perspective) modeled on the real roadways, traffic signals, etc., of Erlangen, Germany. We evaluate our work in two scenarios representing light and heavy traffic densities, respectively set at 60 and 100 vehicles/km based on real-world data [69]. We run 100 iterations of each scenario using current (ECDSA) security, our *Partially Hybrid* design by itself, and our design combined with our spectrum optimization technique, collecting a variety of metrics that showcase the viability and benefits of our work.

B. Scalability

The system capacity of a V2V security protocol (denoted $v_{\max}$) is the maximum number of vehicles the system can support at a time in a given area. This is primarily constrained by *frame duration* and *signature verification time*.

Frame duration. Every vehicle using the channel at one time must be able to transmit at least one BSM per 100 ms interval without overlap (i.e., without blocking each other's transmissions), requiring frame durations to be sufficiently short that at least 100 vehicles (as per traffic data for real-world scenarios [69]) can each transmit one BSM per interval. Using two USRPs connected to laptops placed a few meters apart and each running *PQ-V2Verifier*, we set one USRP to transmit BSMs assuming a worst-case communication scenario; i.e., using the most reliable (i.e., lowest) data rate available in DSRC (6 Mbps). For each instantiation of our *Partially Hybrid* protocol (and ECDSA, for comparison), we measured the frame durations reported in Table IV and calculated the resulting $v_{\max}$. With respect to frame duration, our *Partially Hybrid* design can achieve $100 \leq v_{\max}$ only when instantiated with Falcon. For all other PQ algorithms, $v_{\max} \ll 100$, demonstrating they are unsuitable for V2V.

Signature verification time. To avoid delay in processing BSMs, a vehicle must be able to verify the signature of at least one BSM per interval from every other vehicle that is in communication range. As above, a practical system must support at least 100 vehicles, hence a viable PQ algorithm must be capable of verifying at least 100 signatures per 100 ms interval (i.e., a rate of ≥ 1 kHz). Using Cohda V2V devices [33], we measured signature verification times on their Qualcomm ARMv8 V2V chipsets for Sphincs$^+$, Falcon, Dilithium, and XMSS (and ECDSA, for comparison). For each algorithm, we report in Table V the average ($\bar{x}$) and standard deviation (σ) signature generation and verification times (based on 1000 executions). We found that even without ARMv8 optimizations, Falcon is able to verify 2,243 BSM signatures per second, yielding $v_{\max} = 224$. This not only meets the 100-vehicle requirement for real-world viability, it is higher than the system capacity ($v_{\max}$=101) achievable under the frame duration constraint. Thus, we conclude signature verification time does not actually reduce system capacity for *Partially Hybrid* under Falcon (whereas, e.g., Dilithium meets only the signature verification, not frame duration, constraint). We note that Falcon can also sign 32 messages per second, meeting the requirement to sign (and transmit) 10 BSMs per second.

TABLE V. Impact of sign/verify performance ($\bar{x}$) on system capacity ($v_{\max}$) for PQC vs. ECDSA on Qualcomm V2V chipset.

Sign (milliseconds)					
Algorithm	$\bar{x}$	σ	vs. ECDSA	$10^3/\bar{x}$ (Hz)	Acceptable?
ECDSA	7.820	0.141	1.00	127.881	Yes
Falcon	2.152	0.036	0.275	464.749	Yes
Sphincs$^+$	5.485	0.002	0.701	182.309	No
Dilithium	2.634	1.741	0.337	379.685	Yes
XMSS	1405.408	31.150	179.724	0.712	No
Verify (milliseconds)					
Algorithm	$\bar{x}$	σ	vs. ECDSA	$10^3/\bar{x}$ (Hz)	$v_{\max}$
ECDSA	0.001	0.001	1.00	675219	67521
Falcon	0.446	0.023	300.988	2243	224
Sphincs$^+$	5.436	0.191	3668.120	184	18
Dilithium	0.189	0.184	127.414	5299	529
XMSS	2.780	0.381	1877.402	359	35

C. Efficiency

In a practical system, the *end-to-end delay* for each BSM must not be excessive. The end-to-end delay for one BSM is the sum of frame duration, signature verification time at the receiver, and *waiting time* at the transmitter. As we have previously discussed the first two, only waiting time is remarked on here. Vehicles are supposed to transmit a BSM every 100 ms, but doing so at exact intervals is not possible due to contention-based channel access in DSRC. Every vehicle will generate a BSM precisely every 100 ms, but *transmitting* that BSM must wait for the channel to be idle (based on back-off timers in the CSMA/CA algorithm [43]). Thus, delay is incurred waiting for the channel to be available (measured in 18 μs time slots [43]). This waiting time is easily measured in VEINS; we then combine that with our results from hardware experiments to calculate total end-to-end delay.

Our simulation results for waiting time are shown in Table VI for each scenario and V2V authentication design, along with the transmission (Tx) and signature verification times previously obtained through hardware experiments (Tables V and IV). For ECDSA and our *Partially Hybrid* design without our spectrum optimization, we consider a certificate transmission cycle of 5, based on current standards [39], and a fixed (lowest) data rate of 6 Mbps as per [43]; for *Partially Hybrid* with our spectrum optimization, we consider vehicles that vary the length of their certificate transmission cycle and use an adaptive transmission rate, while using PQ signatures on a subset of BSMs as conditions allow, as described in Section IV. We further calculate the end-to-end BSM time and, for our *Partially Hybrid* instantiation with Falcon, end-to-end delay vs. Pure ECDSA. As Table VI shows,

TABLE VI. EXPERIMENTAL RESULTS FOR *Partially Hybrid* (*PH*) WITH FALCON VS. (CURRENT) ECDSA. ALL TIMES ARE MEDIANS PER-BSM, IN μs, AT DIFFERENT VEHICLE DENSITIES PER KILOMETER (d_v).

d_v	Design	Wait	Tx	Verify	End-to-end time	Delay	FLR
60	ECDSA	28.0	602.4	272.0	902.4	—	0.38
	PH	31.0	982.4	555.0	1568.4	666.0	0.49
	PH + AI	30.4	722.3	555.0	1307.7	250.2	0.41
100	ECDSA	29.0	602.4	272.0	903.4	—	0.42
	PH	35.0	982.4	555.0	1572.4	670.0	0.68
	PH + AI	33.8	714.5	555.0	1303.3	399.9	0.45

even without our spectrum optimization, our *Partially Hybrid* design adds only about 0.6 ms of delay to each BSM in both light and dense traffic conditions, which is extremely good considering the high overhead of PQC. Moreover, with our spectrum optimization we do even better, reducing the added per-BSM delay to just 0.25 and 0.39 ms under light and dense traffic conditions, respectively. This showcases how our optimization significantly reduces the overhead of adopting PQC and encourages *jointly* deploying our optimization and *Partially Hybrid* design for a practical solution.

D. Reliability

Showing our *Partially Hybrid* design is practical requires demonstrating it does not cause an unacceptable increase in BSM loss when deployed in a V2V network. To that end, we measured frame loss across all vehicles in our VEINS simulations for each scenario and V2V protocol design. Frame loss is simply calculated in VEINS as the total number of lost frames at a receiver due to either bit errors (e.g., from low SINR) or from transmitting at the same time as another vehicle; in Table VI, we report the median *frame loss ratio* (FLR), which we define as the ratio of lost frames to total frames detected by a receiver, across all vehicles in each configuration. As Table VI shows, when we deploy our *Partially Hybrid* design by itself, FLR increases significantly—by 0.11 (+28.9%) and 0.26 (+61.9%), respectively, for light and dense traffic scenarios. However, under our combined *Partially Hybrid* design and spectrum optimization, FLR increases vs. ECDSA by just 0.03 (+7.8% and +7.1%, respectively) under both light and dense traffic scenarios. These results illustrate the low overhead of our approach, as well as why our *combined* techniques are best suited for immediate deployment.

Beyond simulations, we show our protocol can be deployed on real vehicles through on-road, outdoor experiments with USRPs running *PQ-V2Verifier* mounted in real vehicles. Using two vehicles, we placed one USRP and one laptop in each, configured each USRP to transmit BSMs at the standard 10 Hz rate (secured using either ECDSA or *Partially Hybrid* with Falcon) and drove each vehicle on a circuit around our campus at Rochester Institute of Technology, maintaining 15–20 m spacing between vehicles and an average speed of 32 km/hr. Analyzing the logs, we find that our *Partially Hybrid* design performed well, with a frame loss rate of just 7.4% (vs. 1.2% for ECDSA). This minimal increase in frame loss corroborates the viability and low overhead of our design, demonstrating our *Partially Hybrid* is ready to roll out right away.

VIII. RELATED WORK

Transitioning from classical (quantum-vulnerable) cryptography to PQC via hybrid designs is well-studied (e.g, in [70], [71], [30], [32], [72], [24], [31], [26], [27]); however, as discussed in Section I and shown in Fig. 1, most such work is for wired systems or other domains with less restrictive (or completely different) constraints. Prior works on PQC in embedded systems and wireless networks are more closely related to our work on V2V. However, many existing works on embedded systems (e.g., [23], [24], [25], [26], [27]) do not consider the above constraints; for wireless systems, PQC has mostly been considered only for key exchange or encryption (in contrast to our focus on authentication) as in [19] (PQ key establishment protocols to encrypt 5G identifiers) and [73] (encryption for video streaming systems). In the vehicular domain, PQC is scantly represented. Most works focus on wired, intra-vehicle communication (e.g., [20], [21]), a totally different problem than inter-vehicle V2V. The few works that consider V2V and 1609.2 tend to focus on proposing alternatives (e.g., [21]) rather than developing a backwards-compatible transition protocol like our *Partially Hybrid*. To the best of our knowledge, we are the first to undertake the specific challenge of devising PQ *authentication* for V2V that can be integrated into the IEEE 1609.2 standard and kickstart the transition to a quantum-secure CV ecosystem.

IX. CONCLUSIONS AND FUTURE WORK

In this paper, we laid out and advanced a path to making PQ authentication viable in V2V communications. We revealed the excessively redundant transmission of signing certificates in today's V2V, developed an AI-enabled technique to optimize spectrum usage and reduce such unnecessary data transmission, and proposed a *Partially Hybrid* protocol that integrates PQC into the IEEE 1609.2 security standard for reliable, low-latency, quantum-resilient V2V communication during the first period of transition from an ECDSA to a fully hybrid ecosystem. We carefully instantiated our *Partially Hybrid* with NIST-approved PQ signature algorithms (e.g., Falcon), formally proved its security, and showed through rigorous in- and outdoor (roadway) experiments that the combination of this design with our spectrum optimization is practical, scalable, and adds minimal delay for safety messages. Our results strongly support Falcon as the most viable PQ algorithm for V2V applications, giving insights we hope will help developing future iterations of V2V standards.

In future work, we intend to expand the scope of our theoretical and experimental analysis to fully include CAs and their associated infrastructure as we move towards a holistic approach to securing V2V against the quantum threat. In addition, we will consider security properties beyond unforgeability, including resistance against attacks such as exclusive ownership and message-bound signatures as analyzed in [64]. Finally, we will consider the complex challenge of designing a practical PQ-V2V protocol for C-V2X. With its radically different physical layer structure, C-V2X imposes much stricter constraints (e.g., a maximum payload size a fraction of that in DSRC) that require significant further investigation.

ACKNOWLEDGMENTS

We thank Dan Brown, Matthew Campagna, Douglas Stebila, and Fernando Virdia for their insights on cryptographic protocols for V2V. This material is based on work supported in part by NSERC Discovery and Accelerator Supplement Grants

RGPIN-2016-05146; Public Works and Government Services Canada Contract 2L 165-180499/001/sv; the NSERC Alliance Program and Public Works Government Services Canada; the University of Waterloo Institute for Quantum Computing; and in part by the U.S. National Science Foundation (NSF) under Award No. 2239931. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

REFERENCES

[1] M. A. Khan, H. El-Sayed, S. Malik, T. Zia, J. Khan, N. Alkaabi, and H. Ignatious, "Level-5 autonomous driving—are we there yet? A review of research literature," *ACM Computing Surveys*, vol. 55, no. 2, pp. 1–38, Jan. 2022.

[2] J. Owens, "Highly automated vehicles: Federal perspectives on the deployment of safety technology," Testimony before the United States Senate Committee on Commerce, Science & Transportation, Nov. 2019. [Online]. Available: https://bit.ly/3Ry7pIe

[3] M. E. Graham, "US Department of Transportation's Vehicle-To-Everything (V2x) Communications Summit: Preparing for V2x deployment," National Transportation Safety Board (NTSB), Washington, D.C., USA, Aug. 2022.

[4] J. Brodkin, "FCC takes spectrum from auto industry in plan to "supersize" Wi-Fi," Nov. 2020, Accessed: July 15, 2021. [Online]. Available: https://bit.ly/3PlIm8C

[5] Federal Communications Commission, "Request for waiver of 5.9 GHz band rules to permit initial deployment of Cellular Vehicle-to-Everything technology," FCC DA-23-343, Apr. 2023.

[6] H. Bagheri, M. Noor-A-Rahim, Z. Liu, H. Lee, D. Pesch, K. Moessner, and P. Xiao, "5G NR-V2X: Toward connected and cooperative autonomous driving," *IEEE Communications Standards Magazine*, vol. 5, no. 1, pp. 48–54, 2021.

[7] K. Sehla, T. M. T. Nguyen, G. Pujolle, and P. B. Velloso, "Resource allocation modes in C-V2X: From LTE-V2X to 5G-V2X," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8291–8314, Jun. 2022.

[8] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Next Generation V2X*, IEEE Standard 802.11bd, Mar. 2023.

[9] *Release 18*, 3rd Generation Partnership Project (3GPP), 2022. [Online]. Available: https://www.3gpp.org/release18

[10] Steven Loveday, "Mercedes-Benz to start selling Level 3 Drive Pilot in Germany," May 2022, Accessed: July 13, 2022. [Online]. Available: https://bit.ly/3Ptx0zh

[11] T. Litman, *Autonomous vehicle implementation predictions*. Victoria, Canada: Victoria Transport Policy Institute, Jun. 2023.

[12] Bureau of Transportation Statistics, "Average age of automobiles and trucks in operation in the United States," 2022, Accessed: June 21, 2023. [Online]. Available: https://bit.ly/3PosWR3

[13] M. Mosca and M. Piani, "Quantum threat timeline report 2021," Global Risk Institute in Financial Services (GRI), 2022, Accessed: July 18, 2022. [Online]. Available: https://globalriskinstitute.org/publications/2022-quantum-threat-timeline-report/

[14] *Wireless Access in Vehicular Environments—Security Services for Application and Management Messages*, IEEE Approved Draft Standard 1609.2-2022, 2022.

[15] *Wireless Access in Vehicular Environments–Certificate Management Interfaces for End Entities*, IEEE Standard 1609.2.1-2022, 2022.

[16] NHTSA, "Report to Congress: Vehicle safety recall completion rates report," Aug. 2021, Accessed: Sep. 21, 2023. [Online]. Available: https://bit.ly/3t0A3aO

[17] 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," *NIST CyberSecurity White Paper*, Apr. 2021.

[18] A. A. Agarkar, M. Karyakarte, and H. Agrawal, "Post quantum security solution for data aggregation in wireless sensor networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Seoul, Korea, May 2020.

[19] V. Q. Ulitzsch, S. Park, S. Marzougui, and J.-P. Seifert, "A post-quantum secure subscription concealed identifier for 6G," in *Proc. ACM Conf. Security and Privacy in Wireless and Mobile Netw. (WiSec)*, San Antonio, TX, 2022, pp. 157–168.

[20] T. Fritzmann, J. Vith, and J. Sepulveda, "Post-quantum key exchange mechanism for safety critical systems," 2019, Accessed: Sep. 21, 2023. [Online]. Available: https://bit.ly/3PrVPfh

[21] 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 Int. Symp. Circuits and Systems (ISCAS)*, Seville, Spain, Oct. 2020.

[22] R. Adelin, C. Nugier, É. Alata, V. Nicomette, V. Migliore, and M. Kaâniche, "Facing emerging challenges in connected vehicles: a formally proven, legislation compliant, and post-quantum ready security protocol," *J. Comput. Virology and Hacking Techniques*, vol. 18, pp. 425–452, Dec. 2022.

[23] W. Wang, B. Jungk, J. Wälde, S. Deng, N. Gupta, J. Szefer, and R. Niederhagen, "XMSS and embedded systems," in *Proc. Int. Conf. Sel. Areas in Cryptography*, Waterloo, Ontario, Aug. 2019, pp. 523–550.

[24] K. Bürstinghaus-Steinbach, C. Krauß, R. Niederhagen, and M. Schneider, "Post-quantum TLS on embedded systems: Integrating and evaluating Kyber and SPHINCS+ with mbed TLS," in *Proc. ACM Asia Conf. Comput. Commun. Security (AsiaCCS)*, Oct. 2020, pp. 841–852.

[25] J. Yao, K. Matusiewicz, and V. Zimmer, "Post quantum design in SPDM for device authentication and key establishment," *Cryptography*, vol. 6, no. 4, p. 48, 2022.

[26] R. Gonzalez and T. Wiggers, "KEMTLS vs. post-quantum TLS: Performance on embedded systems," in *Proc. Int. Conf. Security, Privacy, Appl. Cryptography Eng.*, Jaipur, India, Dec. 2022, pp. 99–117.

[27] G. Tasopoulos, J. Li, A. P. Fournaris, R. K. Zhao, A. Sakzad, and R. Steinfeld, "Performance evaluation of post-quantum TLS 1.3 on resource-constrained embedded systems," in *Proc. Int. Conf. Information Security Practice and Experience*, Taipei, Taiwan, Nov. 2022, pp. 432–451.

[28] *Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process*, NIST IR 8413, 2022.

[29] *Recommendation for Stateful Hash-Based Signature Schemes*, NIST SP 800-208, 2020.

[30] 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. [Online]. Available: https://eprint.iacr.org/2019/858

[31] A. A. Giron, J. P. A. do Nascimento, R. Custódio, and L. P. Perin, "Post-quantum hybrid KEMTLS performance in simulated and real network environments," Cryptology ePrint Archive, 2022.

[32] P. Kampanakis and D. Sikeridis, "Two post-quantum signature use-cases: Non-issues, challenges and potential solutions," in *Proc. ETSI/IQC Quantum Safe Cryptography Workshop*, Seattle, WA, Nov. 2019.

[33] Cohda Wireless, "MK6–Cohda Wireless," 2023, Accessed: Nov. 12, 2023. [Online]. Available: https://bit.ly/3FSPNzU

[34] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," *IEEE Trans. Mobile Comput.*, vol. 10, no. 1, pp. 3–15, Jan. 2011.

[35] *Summary of Rel-16 Work Items*, 3GPP TR 21.915 V16.0.0, 2020.

[36] 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.

[37] P. S. L. M. Barreto, J. E. Ricardini, M. A. Simplicio Jr., and H. K. Patil, "qSCMS: Post-quantum certificate provisioning process for V2X," Cryptology ePrint Archive, Feb. 2019. [Online]. Available: https://eprint.iacr.org/2018/1247

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

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

[40] O. Haran and R. Shallom, "V2X technology trends and market evolution in Europe and China," Aug. 2021, Accessed: Jul. 29, 2022. [Online]. Available: https://bit.ly/453fDeH

[41] M. Slovick, “Toyota, Lexus commit to DSRC V2X starting in 2021,” May 2018, Accessed: July 29, 2022. [Online]. Available: https://web.archive.org/web/20200928152943/https://innovation-destination.com/2018/05/16/toyota-lexus-commit-to-dsrc-v2x-starting-in-2021/

[42] Federal Communications Commission, “In the matter of use of the 5.850-5.925 GHz band (ET Docket No. 19-138),” Nov. 2020.

[43] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2020, 2020.

[44] *User Equipment (UE) radio transmission and reception*, 3GPP TS 36.101 V17.3.0, 2021.

[45] NIST, “Post-quantum cryptography: Round 3 submissions,” 2020. [Online]. Available: https://bit.ly/3PueE18

[46] A. Huelsing, D. Butin, S. Gazdag, J. Rijneveld, and A. Mohaisen, *XMSS: eXtended Merkle Signature Scheme*, Internet Engineering Task Force, May 2018.

[47] “Submission requirements and evaluation criteria for the post-quantum cryptography standardization process,” NIST, 2016. [Online]. Available: https://tinyurl.com/4hwr5amk

[48] P. Schwabe, D. Stebila, and T. Wiggers, “Post-quantum TLS without handshake signatures,” in *Proc. ACM Conf. Comput. Commun. Security (CCS)*, Nov. 2020, pp. 1461–1480.

[49] IBM, “The IBM quantum development roadmap,” https://www.ibm.com/quantum/roadmap, 2022, Accessed: Nov. 22, 2023.

[50] M. Webber, V. Elfving, S. Weidt, and W. K. Hensinger, “The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime,” *AVS Quantum Sci.*, vol. 4, no. 1, Mar. 2022.

[51] PQShield, “PQSoC hardware for embedded devices,” 2023, Accessed: Mar. 24, 2023. [Online]. Available: https://bit.ly/40SdGkK

[52] Crypto4A, “Crypto4A’s QxHSM forever revolutionizes the hardware security module - quantum safe hardware security modules,” May 2023, Accessed: Nov. 29, 2023. [Online]. Available: https://bit.ly/3R3xQDF

[53] *Advanced Encryption Standard*, NIST FIPS 197, 2001.

[54] K. Bhargavan and G. Leurent, “On the practical (in-)security of 64-bit block ciphers: Collision attacks on HTTP over TLS and OpenVPN,” in *Proc. ACM Conf. Comput. Commun. Security (CCS)*, Vienna, Austria, Oct. 2016, pp. 456–467.

[55] *Transitioning the Use of Cryptographic Algorithms and Key Lengths*, NIST Special Publication 800-131A Rev. 2, 2019.

[56] W. Anwar, N. Franchi, and G. Fettweis, “Physical layer evaluation of V2X communications technologies: 5G NR-V2X, LTE-V2X, IEEE 802.11bd, and IEEE 802.11p,” in *Proc. IEEE Veh. Technol. Conf. (VTC)*, Honolulu, HI, USA, Sep. 2019.

[57] E. Eaton and D. Stebila, “The “Quantum Annoying” property of password-authenticated key exchange protocols,” in *Proc. Int. Workshop Post-Quantum Cryptography (PQCrypto)*, Daejon, Korea, Jul. 2021.

[58] 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 *Proc. USENIX Security Symp.*, Aug. 2021.

[59] NHTSA, “Federal Motor Vehicle Safety Standards; V2V communications: 49 CFR 571,” Apr. 2017.

[60] *Intelligent Transport Systems (ITS); Security; Pre-standardization study on pseudonym change management*, ETSI Technical Report 103 415 V1.1.1, Apr. 2018.

[61] C. Aguilar-Melchor, M. R. Albrecht, T. Bailleux, N. Bindel, J. Howe, A. Hülsing, D. Joseph, and M. Manzano, “Batch signatures, revisited,” Cryptology ePrint Archive, Paper 2023/492, Apr. 2023.

[62] A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi, “A closer look at PKI: Security and efficiency,” in *Proc. Int. Workshop Public Key Cryptography*, Beijing, China, Apr. 2007, pp. 458–475.

[63] *Federal Motor Vehicle Safety Standards; V2V Communications*, USDOT Proposed Rule 82 FR 3854, 2017, Accessed: Jul. 30, 2021.

[64] C. Cremers, S. Düzlü, R. Fiedler, M. Fischlin, and C. Janson, “BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures,” in *Proc. IEEE Symp. Security & Privacy*, May 2021.

[65] G. Twardokus, J. Ponicki, S. Baker, P. Carenzo, H. Rahbari, and S. Mishra, “Targeted discreditation attack against trust management in connected vehicles,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, (Virtual) Montreal, QC, Canada, Jun. 2021.

[66] “Botan: Crypto and TLS for modern C++,” 2023, Accessed, Nov. 28, 2023. [Online]. Available: https://botan.randombit.net

[67] N. Allen, M. Anvari, E. Crockett, N. Drucker, V. Gheorghiu, S. Gueron, C. Paquin, T. Lepoint, S. Mishra, and D. Stebila, “liboqs – nist-branch: C library for quantum-resistant cryptographic algorithms,” 2018, https://github.com/open-quantum-safe/liboqs.

[68] H. Becker, V. Hwang, M. J. Kannwischer, B.-Y. Yang, and S.-Y. Yang, “Neon NTT: Faster Dilithium, Kyber, and Saber on Cortex-A72 and Apple M1,” *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, pp. 221–244, Nov. 2021.

[69] G. Twardokus and H. Rahbari, “Towards protecting 5G sidelink scheduling in C-V2X against intelligent DoS attacks,” *IEEE Trans. Wireless Commun.*, vol. 22, no. 11, pp. 7273–7286, Nov. 2023.

[70] R. Azarderakhsh, R. Elkhatib, B. Koziel, and B. Langenberg, “Hardware deployment of hybrid PQC: SIKE+ ECDH,” in *Proc. Int. Conf. Security Privacy Commun. Systems*, Canterbury, UK, Nov. 2021, pp. 475–491.

[71] D. Ghinea, F. Kaczmarczyck, J. Pullman, J. Cretin, R. Misoczki, S. Kölbl, L. Invernizzi, E. Bursztein, and J.-M. Picod, “Hybrid post-quantum signatures in hardware security keys,” *Google Research*, 2022.

[72] D. Stebila, S. Fluhrer, and S. Gueron, “Hybrid key exchange in TLS 1.3,” Internet Engineering Task Force, Internet-Draft draft-ietf-tls-hybrid-design-05, Aug. 2022, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/05/

[73] A. Cohen, R. G. D’Oliveira, S. Salamatian, and M. Médard, “Network coding-based post-quantum cryptography,” *IEEE J. Sel. Areas Commun.*, vol. 2, no. 1, pp. 49–64, Mar. 2021.

[74] D. R. L. Brown, M. J. Campagna, and S. A. Vanstone, “Security of ECQV-certified ECDSA against passive adversaries,” Cryptology ePrint Archive, 2009. [Online]. Available: https://ia.cr/2009/620

[75] N. Bindel, U. Herath, M. McKague, and D. Stebila, “Transitioning to a quantum-resistant public key infrastructure,” in *Proc. Int. Workshop on Post-Quantum Cryptography (PQCrypto)*, Utrecht, The Netherlands, Jun. 2017, pp. 384–405.

[76] “NDSS’24 Call for Artifacts,” Sep. 2023, Accessed: Sep. 30, 2023. [Online]. Available: https://bit.ly/3MSJyQB

Appendix A
Cryptographic Primitive Definitions and Security

Definition 3. *A Digital Signature Scheme is a tuple of algorithms $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$, 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.*

Definition 4 (eUF). *Let $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a digital signature scheme. Let $\mathcal{A}$ be an efficient (classical/quantum) adversary. We define the advantage of $\mathcal{A}$ against the security experiment $\mathrm{Expt}^{\mathsf{eUF}}_{\mathcal{S}}(\mathcal{A})$ as:*

$$Adv^{\mathsf{eUF}}_{\mathcal{S}}(\mathcal{A}) = \Pr\left[\mathrm{Expt}^{\mathsf{eUF}}_{\mathcal{S}}(\mathcal{A}) = 1\right].$$

We say that $\mathcal{S}$ is secure against Existential Unforgeability under Chosen Message Attack (eUF*) if $Adv^{\mathsf{eUF}}_{\mathcal{S}}(\mathcal{A})$ is negligible in the security parameter λ.*

$\mathrm{Expt}^{\mathsf{eUF}}_{\mathcal{S}}$ *is defined as follows. At the beginning, set $q_{\mathsf{sig}} \leftarrow 0$ and $Q_{\mathsf{sig}} \leftarrow \{\}$. The challenger calls* KGen *to return a public key* pk *and secret key* sk*, and passes* pk *to $\mathcal{A}$. $\mathcal{A}$ may query the signing oracle $\mathcal{O}_{\mathsf{Sign}}$ on messages in the message space $\mathcal{M}_{\mathcal{S}}$ as shown in Fig. 8. Eventually, $\mathcal{A}$ outputs a message-signature pair $(\mathsf{m}^*, \mathsf{sig}^*)$. The experiment returns 1 if* $\mathsf{Vrfy}(\mathsf{pk}, \mathsf{sig}, \mathsf{m}) = 1$ *and such that $(\mathsf{m}^*, \cdot) \notin Q_{\mathsf{sig}}$.*

Definition 5 (2PR). *Let* $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{256}$ *be a hash function and* $\mathcal{A}$ *be an efficient (classical/quantum) adversary. We define the advantage of* $\mathcal{A}$ *against* $\mathrm{Expt}_{\mathsf{H}}^{\mathsf{2PR}}(\mathcal{A})$ *as*

$$Adv_{\mathsf{H}}^{\mathsf{2PR}}(\mathcal{A}) = \Pr\left[\mathrm{Expt}_{\mathsf{H}(x)}^{\mathsf{2PR}}(\mathcal{A}) = 1 | x \in \{0,1\}^*\right].$$

We say that H *is secure against second Pre-image Resistance attacks (*2PR*) if* $Adv_{\mathcal{S}}^{\mathsf{2PR}}(\mathcal{A})$ *is negligible in* λ.

$\mathrm{Expt}_{\mathsf{H}}^{\mathsf{2PR}}$ *is defined as follows. At the beginning, set* $q_{\mathsf{H}} \leftarrow 0$ *and* $Q_{\mathsf{H}} \leftarrow \{\}$. *The challenger passes* $x' \in \{0,1\}^*$ *to* $\mathcal{A}$. $\mathcal{A}$ *may query the hashing oracle* $\mathcal{O}_{\mathsf{H}}$ *on elements* $x \in \{0,1\}^*$ *as shown in Fig. 8. Eventually* $\mathcal{A}$ *outputs an element* $x^* \in \{0,1\}^*$. *The experiment returns 1 if* $\mathsf{H}(x^*) = \mathsf{H}(x')$ *and* $x^* \neq x'$.

A. Certified Signature Scheme

The definition of certified signature schemes below encompasses explicit, implicit and hybrid certificates. Following [62], we assume, (S, pk_S) is uniquely bound in the certificate C_S. We leverage the unforgeability of a certified signature scheme [62, Definition 4.1] to define i-unforgeability in Def. 2.

Definition 6. *A certified signature scheme* $\mathcal{C} = (\mathsf{KGen}_C, \mathsf{CGen}(\mathsf{CGen}_C, \mathsf{CGen}_S), \mathsf{Sign}, \mathsf{Vrfy})$ *is defined via the following polynomial-time algorithms.*

KGen_C *returns a key pair* $(\mathsf{pk}_C, \mathsf{sk}_C)$ *belonging to the CA* C.

$\mathsf{CGen}(\mathsf{CGen}_C, \mathsf{CGen}_S)$ *is an interactive (two-party) public-key registration protocol, involving the sender* S *and the CA* C *running their (randomized) sub-protocols* CGen_C *and* CGen_S*, respectively.* CGen_C *takes input* sk_C*;* CGen_S *takes input the identity* S *of a sender and* pk_C *corresponding to* sk_C*. As result of the interaction, the output of* CGen_C *is* $(S, \mathsf{pv}_S, \mathsf{C}_S)$*, where* pv_S *is a public key value* pv_S*, corresponding to a public key* pk_S*, and* C_S *is an issued certificate. If* C_S *is an explicit certificate,* $\mathsf{pv}_S = \mathsf{pk}_S$*; if it is implicit,* pv_S *is the reconstruction value; if it is a hybrid certificate combining two or more sub-certificates,* pv_S *is the concatenation of the corresponding public key values. The local output of* CGen_S *is* $(S, \mathsf{pk}_S, \mathsf{sk}_S, \mathsf{C}_S)$*, where* sk_S *is used by* S *to sign messages.* C *should not learn* sk_S *during* CGen*. Either party can quit the execution prematurely, in which case the output of the party is set to* $\perp$.

Sign *is a signing algorithm. It takes input* S, sk_S, C_S, pk_C, *and* m, *and outputs a signature* sig.

Vrfy *is a deterministic verification algorithm. It takes input* S, pk_S, C_S, pk_C, m, *and* sig, *and outputs 0 or 1. In the latter case, we say that* sig *is a valid signature for* m *relative to* $(S, \mathsf{pk}_S, \mathsf{C}_S, \mathsf{pk}_C)$*. If* C_S *is an implicit certificate this also involves the reconstruction of* pk_S.

```
Oracle $\mathcal{O}_{\mathsf{Sign}}(\mathsf{sk}, \mathsf{m})$:
1  if $\mathsf{m} \notin \mathcal{M}_S$: return $\perp$
2  $\mathsf{sig} \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$
3  $q_{\mathsf{Sign}} \leftarrow q_{\mathsf{Sign}} + 1$
4  $Q_{\mathsf{Sign}} \leftarrow Q_{\mathsf{Sign}} \cup (\mathsf{m}, \mathsf{sig})$
5  return $\mathsf{sig}$
```

```
Oracle $\mathcal{O}_{\mathsf{H}}(x)$:
1  if $x \notin \{0,1\}^*$: return $\perp$
2  $y \leftarrow \mathsf{H}(x)$
3  $q_{\mathsf{H}} \leftarrow q_{\mathsf{H}} + 1$
4  $Q_{\mathsf{H}} \leftarrow Q_{\mathsf{H}} \cup (x, y)$
5  return $y$
```

Fig. 8. Oracles $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{H}}$

Appendix B
Proof of Theorem 1

Our proof follows [62, Proof of Theorem 4.3]. Let $\mathcal{A}$ be an adversary against the i-UF of $\mathcal{P}$. Denote by RegKey the event that in experiment $\mathrm{Expt}_{\mathcal{P}}^{i\text{-}\mathsf{UF}}(\mathcal{A})$, $\mathcal{A}$ outputs a valid forgery with respect to an identity and key that have been properly registered (i.e., condition 1). Moreover, we denote by $\mathsf{UnregKey}$ the event that $\mathcal{A}$ outputs a valid forgery with respect to an identity S^* and a key pk_{S^*} such that pk_{S^*} had not been registered for S^* (i.e., condition 2). In addition, we define the event $\mathsf{DoubleReg}$ which is that $\mathcal{A}$'s forgery is such that the public key of the forgery is registered for two honest senders (i.e., condition 3 while not condition 2). We call a sender honest, if $\mathsf{corrupt} = 0$ during the respective registration query, otherwise a sender is dishonest/corrupt. Immediately,

$$\mathrm{Adv}_{\mathcal{P}}^{i\text{-}\mathsf{UF}}(\mathcal{A}) \leq \Pr[\mathsf{RegKey}] + \Pr[\mathsf{UnregKey}] + \Pr[\mathsf{DoubleReg}].$$

As analyzing the certificate registration and management is out of scope of this paper, we assume $\mathsf{DoubleReg}$ does not occur (i.e., $\Pr[\mathsf{DoubleReg}] = 0$). In what follows, we bound $\Pr[\mathsf{RegKey}]$ and $\Pr[\mathsf{UnregKey}]$ by constructing adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ against the eUF experiment of $\mathcal{S}_i$ and $\mathcal{S}_C$, respectively.

Adversary $\mathcal{B}_1$ plays the eUF experiment for $\mathcal{S}_i$ for fixed i. That means $\mathcal{B}_1$ takes as input pk^* and has access to $\mathcal{O}_{\mathsf{Sign}_i}(\mathsf{sk}^*, \cdot)$ with $(\mathsf{pk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KGen}^{(i)}$. $\mathcal{B}_1$ runs $\mathcal{A}$ in a black-box way, and simulates all inputs and interactions for $\mathcal{A}$ with experiment $\mathrm{Expt}_{\mathcal{P}}^{i\text{-}\mathsf{UF}}(\mathcal{A})$. In particular, $\mathcal{B}_1$ can simulate the response of all honest parties, including that of the CA as described below. Before starting the execution of $\mathcal{A}$, $\mathcal{B}_1$ selects a random position $j \leftarrow_\$ \{1, ..., q_C\}$ and generates $(\mathsf{pk}_C, \mathsf{sk}_C) \leftarrow \mathsf{KGen}_C$. $\mathcal{B}_1$ handles all registration requests by simulating the honest protocol CGen, giving $\mathcal{A}$ access to all transcripts of the registration protocol, except for the j-th registration query. If the latter occurs, the sender's public key pk_{S_j} for the queried S_j is set using pk^* either directly or in a concatenation as a hybrid key.[10] $\mathcal{A}$'s signature queries $(S, \mathsf{pk}_S, \mathsf{C}_S, \mathsf{BSM}, i')$ are answered as follows. If $S = S_j$ and $i' = i$, $\mathcal{B}_1$ queries BSM to $\mathcal{O}_{\mathsf{Sign}_i}$ and uses the response to generate the SPDU following the form defined by i. Otherwise, $\mathcal{B}_1$ can answer the signature queries as $\mathcal{B}_1$ knows all other signing keys. Eventually, $\mathcal{A}$ outputs a forgery $(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}, \mathsf{spdu}^*)$ with spdu^* supposed to be an SPDU of form corresponding to the i-th SPDU. If $\mathcal{A}$'s forgery is valid and $\mathsf{pk}_{S^*} = \mathsf{pk}^*$ (or the hybrid key involving pk^*), $\mathcal{B}_1$ extracts the signature under $\mathcal{S}_i$[11] and outputs the corresponding message-signature pair as its own forgery.

We now analyze the advantage of $\mathcal{B}_1$. The simulation of the environment of $\mathcal{A}$ is perfect. Moreover, we know that BSM^* has never been asked to $\mathcal{O}_{\mathsf{spdu}}$ (since it is a valid forgery of $\mathcal{A}$), hence $\mathcal{B}_1$ also never asked the corresponding message signed by $\mathcal{S}_i$ to $\mathcal{O}_{\mathsf{Sign}_i}$. Thus, $\mathcal{A}$ outputs a valid forgery for a certificate registered by an honest sender with probability $\Pr[\mathsf{RegKey}]$. With probability $\frac{1}{q_C}$, $\mathcal{B}_1$ had set this registered key to (involving) pk^*. Thus, $\Pr[\mathsf{RegKey}] \leq \frac{1}{q_C}\mathrm{Adv}_{\mathcal{S}_i}^{\mathsf{eUF}}(\mathcal{B}_1)$.

[10] $\mathcal{B}_1$ can generate the corresponding certificate including the signature even if this would involve $\mathcal{S}_i$ since a signing key different from sk^* would be used.

[11] Since $\mathcal{A}$'s forgery is valid, which means the SPDU is of the i-th form, we know an $\mathcal{S}_i$ signature is included in spdu^*.

Continuing with $\mathcal{B}_2$, it takes as input pk^* and has access to $\mathcal{O}_{\mathsf{Sign}_C}(\mathsf{sk}^*, \cdot)$, with $(\mathsf{pk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KGen}_C$. $\mathcal{B}_2$ runs $\mathcal{A}$ in a black-box way, simulating all inputs to $\mathcal{A}$ during $\mathrm{Expt}^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A})$. Namely, $\mathcal{B}_2$ sets $\mathsf{pk}_C = \mathsf{pk}^*$. It can still simulate all registration responses to $\mathcal{A}$, using $\mathcal{O}_{\mathsf{Sign}_C}(\mathsf{sk}^*, \cdot)$ as needed. More concretely, if $\mathcal{A}$ wants to register a *corrupt* sender, $\mathcal{B}_2$ receives pk_{S_l} to be registered for S_l. Otherwise $\mathcal{B}_2$ generates $(\mathsf{pk}_{S_l}, \mathsf{sk}_{S_l})$, the corresponding certificate body cbody_{S_l}, and queries $\mathcal{O}_{\mathsf{Sign}_C}(\mathsf{sk}^*, \mathsf{cbody}_{S_l})$ for the needed certificate signature. $\mathcal{A}$'s queries to $\mathcal{O}_{\mathsf{spdu}}$ can be answered trivially by $\mathcal{B}_2$ since it possesses all secret keys of the honest senders.

Eventually, $\mathcal{A}$ outputs its forgery $(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}, \mathsf{spdu}^*)$. If it is valid, it means that $\mathcal{A}$ (with probability $\Pr[\mathsf{UnregKey}]$) produced a valid forgery with respect to an unregistered sender (i.e., $(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}) \notin \mathsf{ListPub}$). Adversary $\mathcal{B}_2$ then extracts cbody^* and sig^* from C_{S^*}, and returns $(\mathsf{cbody}^*, \mathsf{sig}^*)$ as its forgery. It is clear that $\mathcal{B}_2$'s simulation for $\mathcal{A}$ is perfect. Next we explain that $(\mathsf{cbody}^*, \mathsf{sig}^*)$ is a valid forgery for S_C under pk^*. Indeed we know that $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{spdu}^*, \mathsf{st}) = 1$. As $(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}) \notin \mathsf{ListPub}$, spdu^* must include C_{S^*} (and not just, e.g., a hash of the certificate). Hence, internally of CVrfy, it must hold that $\mathsf{Vrfy}_C(\mathsf{sk}^*, \mathsf{cbody}^*, \mathsf{sig}^*) = 1$. Furthermore, since $(S^*, \mathsf{pk}_{S^*}, \mathsf{C}_{S^*}) \notin \mathsf{ListPub}$, $\mathcal{B}_2$ never has to ask $\mathcal{O}_{\mathsf{Sign}_C}$ to sign cbody^*. Thus, it holds that $\Pr[\mathsf{UnregKey}] \leq \mathsf{Adv}^{\mathsf{eUF}}_{\mathcal{S}_C}(\mathcal{B}_2)$.

Appendix C
Security of ECDSA-Based V2V Protocols

In what follows we analyze the security of the original ECDSA-based V2V protocol. We first describe the ECDSA-based design $\mathcal{P} = (\mathsf{KGen_c}, \mathsf{CGen}, \mathsf{SPDUGen}, \mathsf{SPDUVerify})$ using explicit certificates in our notation from Definition 1 with $\mathcal{S}_C = \mathcal{S}_i = \mathcal{S}_\mathsf{c} = (\mathsf{KGen_c}, \mathsf{Sign_c}, \mathsf{Vrfy_c})$.

CGen returns $((S, \mathsf{pk}_S, \mathsf{C}_S),\ (S, \mathsf{pk}_S, \mathsf{sk}_S,\ \mathsf{C}_S))$ with $(\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{KGen}_C$ and C_S being an explicit certificate over pk_S and S, including $\mathsf{sig} \leftarrow \mathsf{Sign_c}(\mathsf{sk}_C, \mathsf{cbody}_S)$.

SPDUGen takes input S, sk_S, C_S, pk_C, BSM, and i. If $i = 1 \mod \tau$, $\mathsf{spdu}_i \leftarrow (\mathsf{BSM}, \mathsf{Sign_c}(\mathsf{sk}_S, \mathsf{BSM}), \mathsf{C}_S)$. Otherwise, $\mathsf{spdu}_i \leftarrow (\mathsf{BSM}, \mathsf{Sign}_{\mathsf{ECDSA}}(\mathsf{sk}_S, \mathsf{BSM}), \mathsf{H}(\mathsf{C}_S))$.

SPDUVerify$(\mathsf{C}_S, \mathsf{pk}_C, \mathsf{spdu}, \mathsf{st})$**:** If $i = 1 \mod \tau$, let $(\mathsf{BSM}, \mathsf{sig}, \mathsf{C}_S) \leftarrow \mathsf{spdu}_i$ and $h \leftarrow \mathsf{H}(\mathsf{C}_S)$. If $\mathsf{Vrfy_c}(\mathsf{pk}_S, \mathsf{BSM}, \mathsf{sig}) = 1$ and $\mathsf{CVrfy}(\mathsf{pk}_C, \mathsf{C}_S, \mathsf{st}) = 1$, update st with h and return 1. Otherwise return 0. If $i \neq 1 \mod \tau$, let $(\mathsf{BSM}, \mathsf{sig}, \mathsf{C}_S, h') \leftarrow \mathsf{spdu}_i$, and return 1 if $\mathsf{Vrfy_c}(\mathsf{pk}_S, \mathsf{BSM}, \mathsf{sig}) = 1$ and $h = h'$, and 0 otherwise.

Corollary 2 (Classical Security of ECDSA-Based V2V with Explicit Certificates)**.** *Let* $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{128}$ *be a second pre-image resistant hash function and* $\mathcal{P}$ *be the ECDSA-based V2V protocol using explicit ECDSA-based certificates described above. Suppose* $\mathcal{A}$ *is a* classical i-UF *adversary against* $\mathrm{Expt}^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A})$ *making at most* q_C *queries to* $\mathcal{O}_\mathsf{C}$*. Then we can construct adversaries* $\mathcal{B}_1$*,* $\mathcal{B}_2$*, and* $\mathcal{B}_3$ *such that*

$$\begin{aligned}
Adv^{1\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A}) &\leq \frac{1}{q_C} Adv^{\mathsf{c}}_{\mathcal{S}_\mathsf{c}}(\mathcal{B}_1) + Adv^{\mathsf{eUF}}_{\mathcal{S}_\mathsf{c}}(\mathcal{B}_2), \textit{ and} \\
Adv^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A}) &\leq \frac{1}{q_C} Adv^{\mathsf{eUF}}_{\mathcal{S}_\mathsf{c}}(\mathcal{B}_1) + Adv^{\mathsf{eUF}}_{\mathcal{S}_\mathsf{c}}(\mathcal{B}_2) \\
&\quad + Adv^{\mathsf{2PR}}_{\mathsf{H}}(\mathcal{B}_3), \textit{ for } i \in [2,5].
\end{aligned}$$

Proof: This corollary follows from Theorem 1. For $i = \tau$, we instantiate both $\mathcal{S}_i$ and $\mathcal{S}_C$ with ECDSA, i.e., $\mathcal{S}_\mathsf{c}$ for the first SPDU. Therefore, we bound the advantage of the adversary by the summation of the advantages of an adversary forging an ECDSA signature over the BSM ($\mathcal{B}_1$) and forging a signature over the certificate ($\mathcal{B}_2$). For $i = [2, .., 5]$, it still holds that $\mathcal{S}_i = \mathcal{S}_C = \mathcal{S}_\mathsf{c}$. However, the SPDUs include only the hash, and not the full pseudonym certificate. Therefore we additionally include the advantage of an adversary finding a second pre-image ($\mathcal{B}_3$) . This is because the adversary can win as soon as they find C' such that $\mathsf{H}(\mathsf{C}') = \mathsf{H}(\mathsf{C}_{S^*})$. ∎

Bounding $\mathrm{Adv}^{i\text{-}\mathsf{UF}}_{\mathcal{P}}(\mathcal{A})$ with $\mathrm{Adv}^{\mathsf{2PR}}_{\mathsf{H}}(\mathcal{B}_3)$ seems to be an obstacle of our security reduction. In practice it is not enough to find a second pre-image, as the adversary would still in addition need to forge the signature on the SPDU since the receiver verifies the SPDU with the public key known from *prior* SPDUs (given in the state st). The obstacle stems from the fact that our security model assumes an adversary that is more powerful than it actually is in practice. Namely, we allow the adversary to include a public key in their forgery, while in practice the public key given in the certificate (received in earlier SPDUs) is used. Since SHA-256 is used for all instantiations of V2V protocols, assuming second pre-image resistance is not an additional constraint in practice.

ECDSA-ECQV-based V2V. IEEE 1609.2 allows to use implicit certificates based on ECQV [38] instead of explicit ECDSA-based certificates. [74] analyzes the composability of ECQV and ECDSA and proves security against passive adversaries. To the best of our knowledge, there is no security analysis of ECQV/ECDSA against active adversaries. As this paper is mostly concerned with PQ alternatives that need to use explicit certificates, we do not analyze the security of ECDSA-ECQV-based V2V regarding i-UF.

Appendix D
Fully Hybrid Design

We describe our *Fully Hybrid* design using the formal V2V protocol definition introduced in Section VI-A, provide its pseudo-code description in Fig. 9, and discuss instantiations.

Let $\mathcal{P}$ be the *Fully Hybrid* protocol using the two signature schemes $\mathcal{S}_\mathsf{c}$ and $\mathcal{S}_\mathsf{pq}$.

KGen_C returns $(\mathsf{pk}_C, \mathsf{sk}_C)$ as in the *Partially Hybrid* design.

CGen returns $\mathsf{C}_S = (\mathsf{C}^\mathsf{c}_S || \mathsf{C}^\mathsf{pq}_S)$ with C^c_S over pk^c_S, C^pq_S over $\mathsf{pk}^\mathsf{pq}_S$, $(\mathsf{sk}^\mathsf{c}_S, \mathsf{pk}^\mathsf{c}_S) \leftarrow \mathsf{KGen}^\mathsf{c}$, $(\mathsf{sk}^\mathsf{pq}_S, \mathsf{pk}^\mathsf{pq}_S) \leftarrow \mathsf{KGen}^\mathsf{pq}$.

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

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

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

```
Sender $S$                                                   Receiver $R$
$\mathsf{sig}_1^{\mathsf{c}} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{BSM}_1, \mathsf{sk}_S^{\mathsf{c}})$
$\mathsf{spdu}_1 \leftarrow (\mathsf{BSM}_1, \mathsf{sig}_1^{\mathsf{c}}, \mathsf{C}_1)$   $\xrightarrow{\mathsf{spdu}_1}$   $(\mathsf{BSM}_1, \mathsf{C}_1) \leftarrow \mathsf{spdu}_1$
                                                             $(\mathsf{C}_S^{\mathsf{c}} || \mathsf{C}_{\mathsf{frac}}) \leftarrow \mathsf{C}_1$
                                                             if $\mathsf{CVrfy}(\mathsf{pk}_C^{\mathsf{c}}, \mathsf{C}_S^{\mathsf{c}}) = 1$ :
                                                               if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_S^{\mathsf{c}}, \mathsf{sig}_1^{\mathsf{c}}, \mathsf{BSM}_1) = 1$ :
for $i = 2, \ldots, \alpha - 1$ :                                  $\mathsf{Process}(\mathsf{BSM}_1)$
$\mathsf{sig}_i^{\mathsf{c}} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{BSM}_i, \mathsf{sk}_S^{\mathsf{c}})$                 Abort
$h^{\mathsf{c}} \leftarrow \mathsf{H}(\mathsf{C}_S^{\mathsf{c}})$
$\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}_i^{\mathsf{c}}, h^{\mathsf{c}}, \mathsf{C}_i)$
                                   $\xrightarrow{\mathsf{spdu}_i}$   $(\mathsf{BSM}_i, \mathsf{C}_i, h^{\mathsf{c}}) \leftarrow \mathsf{spdu}_i$
                                                             if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_S^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1$ :
                                                               if $h^{\mathsf{c}} == \mathsf{H}(\mathsf{C}_s^{\mathsf{c}})$ :
                                                                 $\mathsf{Process}(\mathsf{BSM}_i)$
$\mathsf{sig}_\alpha^{\mathsf{c}} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{BSM}_\alpha, \mathsf{sk}_S^{\mathsf{c}})$                 Abort
$h \leftarrow \mathsf{H}(\mathsf{C}_S)$
$\mathsf{spdu}_\alpha \leftarrow (\mathsf{BSM}_\alpha, \mathsf{sig}_\alpha^{\mathsf{c}}, h, \mathsf{C}_\alpha)$
                                   $\xrightarrow{\mathsf{spdu}_\alpha}$   $(\mathsf{BSM}_\alpha, \mathsf{C}_\alpha) \leftarrow \mathsf{spdu}_\alpha$
                                                             $\mathsf{C}_S \leftarrow \mathsf{CCons}(\mathsf{C}_1, \ldots, \mathsf{C}_\alpha)$
                                                             if $\mathsf{CVrfy}(\mathsf{pk}_C^{\mathsf{pq}}, \mathsf{C}_S) = 1$ :
                                                               if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{BSM}_\alpha, \mathsf{sig}_\alpha^{\mathsf{c}}, \mathsf{pk}_S^{\mathsf{c}}) = 1$ :
for $i = \alpha + 1, \ldots, \tau$ :                                   if $h == \mathsf{H}(\mathsf{C}_S)$ :
$\mathsf{sig}_i^{\mathsf{pq}} \leftarrow \mathsf{Sign}^{\mathsf{pq}}(\mathsf{sk}_S^{\mathsf{pq}}, \mathsf{BSM}_i)$                     $\mathsf{Process}(\mathsf{BSM}_\alpha)$
$\mathsf{sig}_i^{\mathsf{c}} \leftarrow \mathsf{Sign}^{\mathsf{c}}(\mathsf{sk}_S^{\mathsf{c}}, \mathsf{BSM}_i)$                 Abort
$\mathsf{sig}_i \leftarrow (\mathsf{sig}_i^{\mathsf{pq}} || \mathsf{sig}_i^{\mathsf{c}})$
$\mathsf{spdu}_i \leftarrow (\mathsf{BSM}_i, \mathsf{sig}_i, h)$   $\xrightarrow{\mathsf{spdu}_i}$   $(\mathsf{BSM}_i, (\mathsf{sig}_i^{\mathsf{c}} || \mathsf{sig}_i^{\mathsf{pq}}), h) \leftarrow \mathsf{spdu}_i$
                                                             if $\mathsf{Vrfy}^{\mathsf{pq}}(\mathsf{pk}_S^{\mathsf{pq}}, \mathsf{sig}_i^{\mathsf{pq}}, \mathsf{BSM}_i) = 1$ :
                                                               if $\mathsf{Vrfy}^{\mathsf{c}}(\mathsf{pk}_S^{\mathsf{c}}, \mathsf{sig}_i^{\mathsf{c}}, \mathsf{BSM}_i) = 1$ :
                                                                 if $h == \mathsf{H}(\mathsf{C}_S)$ :
                                                                   $\mathsf{Process}(\mathsf{BSM}_i)$
                                                             Abort
```

Fig. 9. Pseudo-code description of the *Fully Hybrid* design to be repeated every τ BSMs.

Discussion on PQ Security. As we first need to transmit the entire hybrid certificate before we can sign and verify the hybrid PQ-ECDSA signatures, the first α message(s) of each certificate transmission cycle are only protected by ECDSA. Ideally, all messages should be authenticated by ECDSA *and* PQ signatures. However, embedding both the PQ certificate and signature in the first frame would incur a large frame size. Losing an important BSM due to its large frame size poses a more severe risk than a quantum adversary who would need to successfully run a very precise attack on the first SPDU in the certificate transmission cycle. We formalize the security of our design using explicit ECDSA certificates.

Corollary 3 (i-UF of the *Fully Hybrid*). *Let* $\mathsf{H} : \{0,1\}^* \rightarrow \{0,1\}^{128}$ *be a hash function and* $\mathcal{P}$ *be the* Fully Hybrid *V2V protocol using explicit hybrid PQ-ECDSA certificates as described above. Suppose* $\mathcal{A}$ *is an* i-UF *adversary against* $\mathrm{Expt}_{\mathcal{P}}^{i\text{-}\mathsf{UF}}(\mathcal{A})$ *making at most* q_C *queries to* $\mathcal{O}_{\mathsf{C}}$*. Then we can construct adversaries* $\mathcal{B}_1^{\mathsf{c}}$, $\mathcal{B}_1^{\mathsf{pq}}$, $\mathcal{B}_2^{\mathsf{c}}$, $\mathcal{B}_2^{\mathsf{pq}}$, *and* $\mathcal{B}_3$ *such that* $Adv_{\mathcal{P}}^{i\text{-}\mathsf{UF}}(\mathcal{A}) \leq$

$$\begin{cases} \frac{1}{q_C} Adv_{\mathcal{S}_{\mathsf{c}}}^{\mathsf{eUF}}(\mathcal{B}_1^{\mathsf{c}}) + Adv_{\mathcal{S}_{\mathsf{c}}}^{\mathsf{eUF}}(\mathcal{B}_2^{\mathsf{c}}), \text{ for } i = 1 \\ \frac{1}{q_C} Adv_{\mathcal{S}_{\mathsf{c}}}^{\mathsf{eUF}}(\mathcal{B}_1^{\mathsf{c}}) + Adv_{\mathcal{S}_{\mathsf{c}}}^{\mathsf{eUF}}(\mathcal{B}_2^{\mathsf{c}}) + Adv_{\mathsf{H}}^{\mathsf{2PR}}(\mathcal{B}_3), \text{ for } i \in [2, \alpha], \\ \frac{1}{q_C} \min\left(Adv_{\mathcal{S}_{\mathsf{c}}}^{\mathsf{eUF}}(\mathcal{B}_1^{\mathsf{c}}), Adv_{\mathcal{S}_{\mathsf{pq}}}^{\mathsf{eUF}}(\mathcal{B}_1^{\mathsf{pq}})\right) + \min\left(Adv_{\mathcal{S}_{\mathsf{c}}}^{\mathsf{eUF}}(\mathcal{B}_2^{\mathsf{c}}),\right. \\ \qquad \left. Adv_{\mathcal{S}_{\mathsf{pq}}}^{\mathsf{eUF}}(\mathcal{B}_2^{\mathsf{pq}})\right) + Adv_{\mathsf{H}}^{\mathsf{2PR}}(\mathcal{B}_3), \text{ for } i \in [\alpha + 1, \tau]. \end{cases}$$

Proof: The corollary follows from Theorem 1 as follows.

TABLE VII. Resulting sizes of frames F_i (in bytes) for our *Fully Hybrid* design; $|\mathsf{C}_S|$ is the size of the entire certificate.

PQ Scheme	$\lvert\mathsf{C}_S\rvert$	α	F_1	F_2	F_3	F_4	F_5	β
Pure ECDSA Design								
-	162	1	330	200	200	200	200	1
Fully Hybrid Design								
Falcon	1723	1	1894	894	894	894	894	2

For $i = 1$, we have $\mathcal{S}_i = \mathcal{S}_C = \mathcal{S}_{\mathsf{c}}$, and the SPDU includes the full ECDSA certificate, forging signatures over the BSM or the certificate reduces to adversaries forging an ECDSA signature over the BSM ($\mathcal{B}_1^{\mathsf{c}}$) or forging an ECDSA signature over the certificate ($\mathcal{B}_2^{\mathsf{c}}$), respectively. For $i \in [2, \alpha]$, the SPDU differs in that it includes the hash instead of the full, ECDSA certificate, so we additionally include the advantage of an adversary finding a second pre-image ($\mathcal{B}_3$). For $i \in [\alpha+1, \tau]$, all signatures are concatenations of $\mathcal{S}_{\mathsf{c}}$ and $\mathcal{S}_{\mathsf{pq}}$ signatures, so the advantage of forging signatures over the BSM or the certificate is the minimum of forging either a classical or PQ signature, as *both* of them would have to be forged for the adversary to successfully forge the hybrid signature. In addition, we include the advantage of finding a second pre-image ($\mathcal{B}_3$) as it is still a possibility for the adversary to mount this attack on the hash of the full certificate to win. ■

Backwards-Compatibility. Like the *Partially Hybrid*, our *Fully Hybrid* design can be extended to be backwards-compatible. The difference between Fig. 9 and its backwards-compatible variant lies in whether the receiver verifies the PQ signature or not. (More concretely, in the handling of the $[\alpha, \tau]$-th SPDUs in SPDUVerify). As before (see Section V-C), we assume all vehicles send and expect to receive hybrid certificates, even if they do not possess the *hardware* capabilities to verify the PQ signature, in order to prevent rollback attacks. We also require each sender's certificate to indicate whether the sender has PQ-signing capabilities, so the verifier knows whether to run PQ-verification, and the fact this is signed by the CA prevents an adversary mounting a rollback attack. We note this is not included in our implementation, as we assume PQ capabilities for all vehicles. Following [75], security of both schemes can only be guaranteed for honest sender/receivers and if both PQ and ECDSA signatures are verified. Generating PQ signatures does not give extra security for receivers without hardware updates compared to pure ECDSA V2V ; i.e., if the receiver supports PQ and processes the algorithms as above, the security is as per Corollary 3. Otherwise, the security is the same as for pure ECDSA as per Corollary 2 (see Appendix C).

Instantiation and Resulting Frame Sizes. Only signature algorithms whose associated certificates can be sent in five or fewer fragments α can be used in the *Fully Hybrid* design, due to the five-message minimum in the certificate transmission cycle of the protocol. As mentioned in Section D, the only viable instantiation on current hardware and under the current size constraints is Falcon, with resulting frame sizes reported in Table VII. We chose to instantiate $\alpha = 1$. To be more concrete, using explicit ECDSA certificates the frame F_1 is 1894 bytes, as it contains the BSM, the ECDSA signature, and the entire certificate C_S (see Table VII, column "F_1"). As explained in Section V-D, β indicates how many frames are necessary during P2PCD. Although Dilithium and XMSS certificates can be split similarly to Falcon, their signatures alone exceed the $2,304$-byte payload limit; therefore, we do not instantiate our design using them.

Appendix E
Artifact Appendix

A. Description & Requirements

We provide three artifacts: (1) a pre-configured virtual machine (VM), *PQ-Benchmarks*, with our PQ algorithm and ECDSA benchmarking code installed; (2) a pre-configured VM, *PQ-V2Verifier*, with our *PQ-V2Verifier* testbed installed; and (3) a pre-configured VM, *PQ-VEINS*, with the VEINS simulator and our PQ VEINS module installed.

1) How to access: The VMs containing the artifacts that were used for evaluation and awarding of badges are permanently archived at https://doi.org/10.5281/zenodo.10160535. Archival versions of the source code for the three artifacts run by these VMs are also available archivally at

- PQ-Benchmarks: https://doi.org/10.5281/zenodo.10189511
- PQ-V2Verifier: https://doi.org/10.5281/zenodo.10189528
- PQ-VEINS: https://doi.org/10.5281/zenodo.10189557

2) Hardware dependencies: No hardware is required beyond a "commodity PC." As defined by the NDSS 2024 Call for Artifacts [76], "[a] commodity desktop machine is defined as one with an x86-64 CPU with 8 cores and 16 GB of RAM running a recent Linux or Windows operating system."

3) Software dependencies: Each VM is provided in Open Virtual Appliance (OVA) format. We used Version 7.0.8 r156879 (Qt5.15.2) of Oracle's free VirtualBox software to create and test the artifact VMs, and we recommend evaluators use the same version to ensure consistency. The specific supported operating systems and required software packages for each artifact are enumerated in the READMEs of the repositories linked above.

4) Benchmarks: None.

B. Artifact Installation & Configuration

Each artifact can be evaluated by downloading the OVA file for the relevant VM from the DOI record above, launching VirtualBox, and importing the VM (File → Import Appliance). When importing is complete, each VM can be booted up by selecting it in the left sidebar and clicking the green arrow ("Start") icon in the top navigation menu. For the *PQ-V2Verifier* and *PQ-Benchmarks* VMs, log in when prompted as "User" with the password "password". On the *PQ-VEINS* VM, log in with the username "veins" and password "veins". Specific instructions from that point on for each VM are provided in the relevant sections below.

C. Major Claims

- (C1): We measure signature generation and verification times for Sphincs$^+$, Falcon, Dilithium, XMSS, and ECDSA using *liboqs* [67] and Botan [66]. As demonstrated in experiment (E1), our source code is publicly available and facilitates benchmarking these algorithms on either ARM or x86_64 chipsets, thus allowing evaluation on V2V devices like we used for our experiments (see [33]), other embedded systems, or traditional PCs (as in this evaluation).
- (C2): Our *PQ-V2Verifier* testbed provides the necessary elements to implement and experiment with post-quantum authentication protocols in an IEEE 1609.2-compliant V2V environment; in particular, Falcon support is packaged to enable experimentation with the only NIST-approved algorithm we find likely to be feasible for real-world V2V systems. Experiment (E2) demonstrates the availability and functionality of *PQ-Verifier*.
- (C3): Our PQ-VEINS post-quantum module for the VEINS V2V simulation tool provides a novel environment for evaluating the practicality of V2V authentication protocols (both classical and post-quantum) in realistic environments. Experiment (E3) demonstrates the availability of PQ-VEINS and showcases the functionality of our code via simplified examples based on the experimental code used for our paper.

D. Evaluation

1) Experiment (E1)—Feasibility Benchmarks (15 minutes): This experiment supports claim (C1) by demonstrating that our code is **available** and can be compiled successfully for ARM or x86 processors, and by confirming our code is **functional** by locally executing the x86 version.

[Preparation] Follow the instructions under *Artifact Installation & Configuration* to download, launch, and log into the *PQ Benchmarks* VM.

[Execution] The source code for our benchmarking is preloaded in the `pq-benchmarks` directory. Open a terminal and move into this directory with the command:

```
cd $HOME/pq-benchmarks
```

This directory contains the source code to build test binaries for ECDSA, Falcon, Dilithium, Sphincs$^+$, and XMSS. The provided `Makefile` can be used to build all or a subset of these binaries for x86_64, ARMv8, or both architectures. For simplicity, we provide a script to build all of these binaries (for both architectures) and run the x86 versions of each binary in the VM, all with a single command:

```
./functional_test.bash
```

In Section VII-B, we evaluate each algorithm based on its average and standard deviation runtime for 1000 executions (see Table V). However, some of the algorithms (e.g., XMSS, Sphincs$^+$) can take a few seconds or more for each signature generation or verification, which prolongs the experiments. Since access to the V2V devices we used for our experiments (see [33]) is not expected for the evaluator, reproducibility of these results is neither expected nor targeted. Hence, we reduce the number of iterations over each algorithm to 10 in order to reduce experiment length. If desired, the evaluator can easily increase the number of iterations by adjusting the value of the `ITERATIONS` variable defined in the `util.h` source file and rebuilding the project.

[Results] As the test binary for each algorithm completes, the average and standard deviation for the runtime of that algorithm will be displayed in the terminal. These results are also written to text files (e.g., `ecdsa_results.txt`) for more convenient inspection.

Our source code is permanently available at the DOI record above for *PQ-Benchmarks*. The successful compilation of our source code for ARM and x86 architectures, as well as the successful execution of the x86 versions within the VM, confirm the functionality of this artifact. Since the devices we used to obtain the results in Table V are very likely not available to the evaluator, our results are not expected to be reproducible, but this experiment sufficiently demonstrates the availability and functionality of our code to the greatest extent possible in the absence of this speciality hardware.

2) Experiment (E2)—Using PQ-V2Verifier (≤ 30 minutes): This experiment supports claim (C2) by demonstrating the **availability** and **functionality** of our *PQ-V2Verifier* code.

[Preparation] Follow the instructions under *Artifact Installation* to download, launch, and access the *PQ-V2Verifier* VM.

[Execution] See the instructions under "Running PQ-V2Verifier in test mode" in the project `README` file to run a proof-of-functionality example using *PQ-V2Verifier*.

[Results] The access to our *PQ-V2Verifier* source code on GitHub and the ability of the evaluator to run the basic example described by the project README demonstrate both the availability and functionality of this artifact. No specific experimental results are collected; the available and functional artifact is itself the expected outcome.

3) Experiment (E3)—Using PQ-VEINS (up to 1 hour): Run a simple scenario in VEINS that demonstrates how our implementation of V2V security protocols (classical and post-quantum) facilitates studying the impact of V2V security on network performance. This experiment supports claim (C3) by demonstrating how we integrate and evaluation PQC (Falcon) and classical (ECDSA) V2V security into VEINS.

Please note that we do not provide our complete implementation of the *Partially Hybrid* protocol in PQ-VEINS, nor do we provide our implementation of the AI-based spectrum optimization described in Section IV, as including these elements in simulations results in significantly exceeding the computational time and resource constraints described in the Call for Artifacts. However, these elements will be made available on GitHub in the *PQ-VEINS* repository archived under the above DOI.

[Preparation] Follow the instructions under *Artifact Installation* to download, launch, and log into the *PQ-VEINS* VM.

[Execution] While not necessary to complete this experiment, the evaluator may wish to review the project README file for context on PQ-VEINS before beginning. Proceed by opening a new terminal window and launching the VEINS background process:

```
veins_launchd -vv
```

Next, open a new terminal window and navigate to the PQ-VEINS application directory:

```
cd $HOME/src/veins/examples/veins
```

There are several pre-configured options for simulations to run, each combining a different type of V2V security (i.e., ECDSA or Falcon) with a specific data rate (BPSK 1/2 or BPSK 3/4), the use or absence of P2PCD for learning pseudonym certificates (see Section IV-B), etc. We have included a comprehensive run file, `runs.sh`, which enumerates all of the possible pre-configured scenarios (note, we do not recommend actually executing this file, as completing all of the available configurations will take multiple days of computing time). The options can be conveniently viewed in a terminal with the command `cat runs.sh`.

For simplicity and a rapid proof of the functionality of PQ-VEINS, the fastest configuration is ECDSA with BPSK 1/2 and no P2PCD in use. This configuration can be executed with the command:

```
./run -u Cmdenv -c V2V-ECDSA-BPSK12-NOP2PCD-500MS
```

By default, this will repeat the simulation 100 times, each time with a different seed, to collect a reasonable distribution of results. However, for evaluation purposes, the repetition can be stopped at any time (after allowing at least a few minutes for 1–2 iterations to complete) with `Ctrl+C`, and partial results up to that point will be written to the disk upon program exit.

[Results] As our AI and *Partially Hybrid* components of PQ-VEINS cannot be evaluated within the constraints of the Call for Artifacts, we do not attempt to reproduce specific results from the paper. However, results from these simplified example implementations (which still demonstrate the availability and functionality of our PQ-VEINS module) are written to the `results` folder and can be examined if desired. First, use the post-processing utility that we provide in order to convert the simulation results from scalar vectors into a more convenient CSV format:

```
python3 ./process_results.py
```

This will create one CSV file in the `results` directory for each iteration of the simulation(s) that were completed.

E. Notes

The experiments described in this appendix are intentionally simplified and streamlined for the artifact evaluation process. Complete versions of source code, along with installation instructions bypassed herein, are available in the public, open-source code repositories archived above. We further encourage the reader to look at the README files included with each artifact for additional documentation on installing and configuring our software for use in other projects.