An Integrated Safe and Secure Approach for Authentication and Secret Key Establishment in Automotive Cyber-Physical Systems

Naresh Kumar Giri[1], Arslan Munir[2], and Joonho Kong[3]

[1] Intel Corporation, Santa Clara, CA 95054, USA,
ngiri@ksu.edu,

[2] Kansas State University, Manhattan, KS 66506, USA,
amunir@ksu.edu

[3] Kyungpook National University, Daegu 41566, South Korea
joonho.kong@knu.ac.kr

Abstract. In this paper, we propose an integrated safe and secure approach for operation in automotive cyber-physical systems (CPS). The proposed approach incorporates a novel protocol for authentication and secret key establishment for electronic control units (ECUs) in automotive CPS. The approach leverages certificates and elliptic curve cryptography (ECC) for authentication and secret key establishment, and symmetric encryption and hash-based message authentication codes for providing confidentiality and integrity, respectively, for messages on in-vehicle bus. To incorporate safety primitives, the approach leverages multicore ECUs and provide fault tolerance by redundant multi-threading (FT-RMT), FT-RMT enhanced by quick error detection (FT-RMT-QED), and FT-RMT with lightweight check-pointing (CP). The proposed approach ensures that the simultaneous integration of security and safety primitives in intra-vehicle ECU communication does not violate real-time constraints of automotive CPS applications. We demonstrate the proposed approach through a steer-by-wire case study. Results verify that our proposed approach integrates confidentiality, integrity, authentication, and secret key establishment in intra-vehicle networks without violating real-time constraints even in the presence of errors in computation and transmission.

Keywords: Automotive, cyber-physical systems, fault tolerance, security, authentication, key establishment

1 Introduction and Motivation

Modern vehicles are equipped with a multitude of sensors, radio interfaces, and digital processors, also known as electronic control units (ECUs), that are connected with each other via in-vehicle networks, such as controller area network (CAN), CAN with flexible data-rate (CAN FD), local interconnect network

(LIN), and media oriented systems transport (MOST) [12]. However, most of the contemporary automotive ECUs and in-vehicle networks do not have built-in security and/or safety primitives thus making automotive systems susceptible to security and safety vulnerabilities. Attackers can infiltrate into in-vehicle networks through a compromised ECU and can read/alter messages, which enables the attackers to control many safety critical systems such as disabling brakes, stopping the engine, opening doors, changing heating and cooling, and turning on/off lights [8, 11]. Cyber-physical attributes of modern automotive systems directly relates security vulnerabilities to automobile's physical safety and dependability.

In addition to security vulnerabilities, modern automobiles are also susceptible to electronic failures. Harsh operating environments, external noise, and radiations make automotive cyber-physical systems (CPS) susceptible to *permanent*, *transient* and *intermittent faults*. Automotive CPS have stringent safety requirements as stipulated by ISO 26262 [6]. ISO 26262 requires that at least one critical fault must be tolerated by automobiles without loss of functionality. Thus, both security and safety primitives need to be incorporated in automotive CPS. However, simultaneous integration of security and safety in automotive CPS is challenging. The biggest challenge in the simultaneous integration of security and safety is to avoid violation of the automotive CPS application's hard real-time constraints.

Previous works [12, 14] have incorporated symmetric cryptography primitives, such as advanced encryption standard (AES) for confidentiality along with hash-based message authentication code (HMAC) for message integrity. The symmetric cryptography, however, requires pre-shared symmetric keys between communicating parties. Most of the existing works assume that these keys are pre-programmed by the original equipment manufacturers (OEMs) during vehicle manufacturing. Nevertheless, OEMs tend to use identical keys across series of ECUs and even vehicles, which makes an entire series of ECUs and vehicles vulnerable to security attacks when a single key is compromised. Furthermore, stored symmetric keys can be easily extracted using side-channel analysis (SCA) attacks, which render storage of permanent symmetric keys in ECUs susceptible to vulnerabilities.

In this paper, we propose a safe and secure approach for *symmetric (session) key establishment* as well as regular operation in automotive CPS. The proposed key establishment protocol eliminates the need for storing symmetric keys permanently in ECUs thus preventing an attacker from gaining access to symmetric keys through SCAs. Our proposed scheme ensures that only authenticated ECUs can participate in communication over the intra-vehicle network. The ECU authentication and secret key establishment in our approach leverages certificates and elliptic curve cryptography (ECC). ECC is chosen over other asymmetric cryptography approaches (e.g., RSA, Elgamal) because ECC provides higher security with comparatively shorter key lengths. ECC implementations, therefore, have lower computation complexity and are more suitable for applications having real-time deadlines such as automotive CPS.

To address the safety requirements stipulated by ISO 26262 [6], we have incorporated various fault tolerance (FT) approaches such as FT by redundant multi-threading (FT-RMT), FT-RMT enhanced with quick error detection (FT-RMT-QED), and FT-RMT with checkpointing (FT-RMT-CP). Our main technical contributions are as follows:

- Proposal of an integrated safety and security approach that simultaneously incorporates security (key establishment, confidentiality, integrity, and authentication) and FT primitives while adhering to real-time constraints of automotive CPS. We demonstrate this approach through a steer-by-wire (SBW) case study.
- Proposal of a certificate-based authentication scheme for ECUs leveraging ECC to ensure that only authenticated ECUs can participate in in-vehicle communication.
- Proposal of a novel symmetric (session) key establishment protocol to enable the ECUs to communicate securely over the in-vehicle networks.
- Safety integration through various FT approaches such as FT-RMT, FT-RMT-QED, and FT-RMT-CP.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 presents the proposed integrated safety and security approach. The proposed certificate-based ECU authentication and symmetric key establishment mechanisms are elaborated in Section 4. Section 5 discusses the SBW system and its timing model, which is used as a case study to verify our proposed approach. Section 6 discusses the results. Finally, Section 7 presents concluding remarks.

2 Related Work

Many previous works have studied security of automotive systems. Koscher et al. [8] analyzed internal and external attack surfaces through which an attacker could control automotive subsystems. The authors practically demonstrated an attack on a car through onboard diagnostics (OBD) port by using a self-developed software. The authors successfully controlled radio, instrument panel cluster, body controller, engine, brakes, and heating, ventilation, and air conditioning (HVAC) subsystems. Rouf et al. [5] studied the security and privacy of a tire pressure monitoring system (TPMS). Huang et al. [4] classified the in-vehicle network in three different layers: control layer, middle layer and external interface layer, and studied the security vulnerabilities at each layer. All these works have motivated the necessity of security and authentication mechanisms within in-vehicle networks for realizing secure and dependable automotive CPS.

Lin et al. [9] proposed integration of message authentication codes (MACs) in CAN data frames to prevent masquerade and replay attacks. Wolf et al. [17] proposed a vehicular hardware security module (HSM) that provided hardware support for symmetric cryptography, asymmetric cryptography, hash function, and pseudorandom number generator. However, the HSM did not support any

FT features which are crucial for safe operation of modern automobiles. Fassak et al. [2] proposed a protocol for authenticating ECUs and establishing session keys between them on the CAN bus using ECC. However, if a new ECU was added to the CAN bus, the protocol required the storage of all other ECUs to be updated with the public key of the new ECU. Furthermore, the protocol utilized truncated MACs which could increase the probability of collision between hashes.

The safety for automotive embedded system has been studied in some previous works. Beckschulaze et. al. [1] have studied different FT approaches on dual-core micro-controllers. Munir et al. [12] and Poudel et al. [14] have proposed multicore ECU based design for secure and dependable cybercars. However, these works did not discuss symmetric key establishment and distribution for automotive CPS.

3 Integrated Safety and Security Approach

Figure 1 provides an overview of our proposed integrated safe and secure approach for cybercar design. In this work, we focus on CAN FD as the vehicular network, however, our approach is equally applicable for CAN and FlexRay. The figure shows the operations involved at both the sending and receiving CAN FD nodes to integrate safety and security primitives.

3.1 Safety

To address the safety requirements stipulated by ISO 26262 [6], we incorporate various FT approaches such as FT-RMT, FT-RMT-QED, and FT-RMT-CP. The FT-RMT uses two different threads and a dual-core architecture to compute the same safety-critical computation. The results of the two threads are matched at the end of the computation to detect any error. If an error occurs during computation, recomputation is carried out in both the threads. This recomputation fixes the errors that are caused by transient faults which constitutes majority of errors in automotive CPS. The FT-RMT-QED enhances FT-RMT with quick error detection (QED) mechanism [12]. In FT-RMT-QED, the main thread executes original instructions and the check instructions, which are inserted at different points in the program/computation, whereas another thread executes duplicated instructions. The FT-RMT-QED permits earlier detection of errors in the program via inserted checks as compared to error detection at the end of the program in FT-RMT. The FT-RMT-CP introduces checkpoints at various portions of the safety-critical code. When an error/fault occurs in the program, the execution resumes from previous (last) checkpoint, which removes the need of re-executing the whole program during errors. The checkpoint is lightweight because the checkpointing in FT-RMT-CP stores minimum state information just enough to resume the computation.

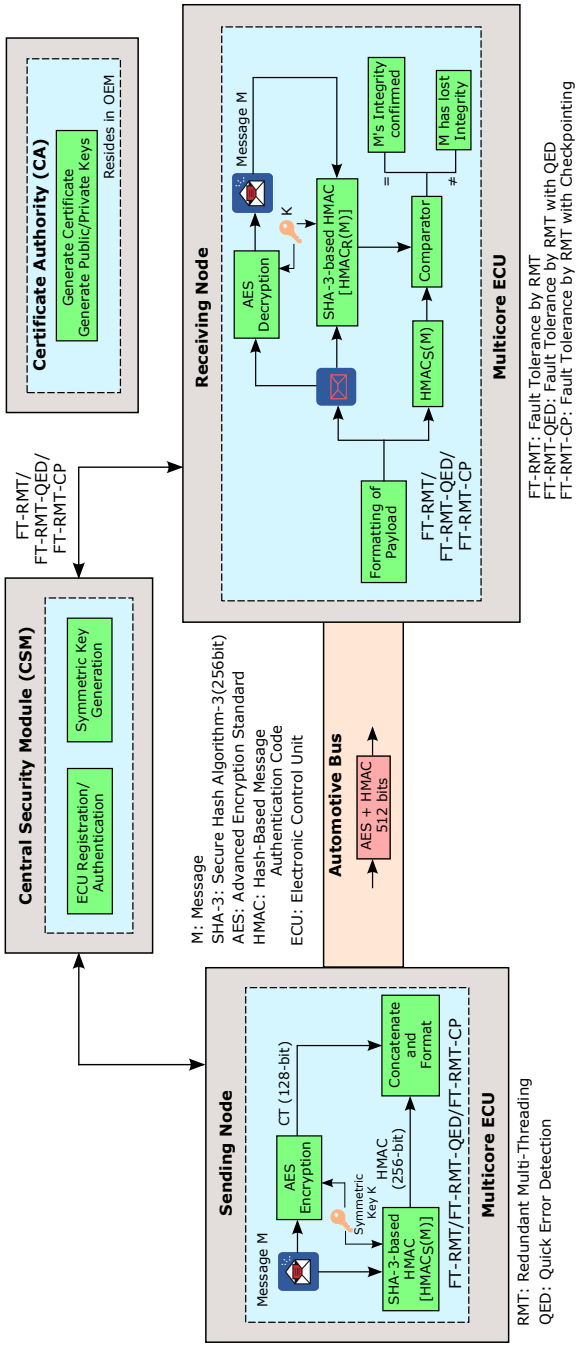

Fig. 1. An Integrated safe and secure approach for in-vehicle networks.

3.2 Security Threat Model

Assuming an adversary has gained access to in-vehicular network, this section briefly discuses the associated security threat model against which our proposed approach provides resilience [12, 14].

Threat 1—Passive Eavesdropping & Traffic Analysis: An attacker may perform passive eavesdropping which means he/she can sniff, steal, and analyze all the traffic information from intra-vehicular network to obtain critical information about driver, vehicle, and navigation routes, which can put the driver and passenger at great risk.

Threat 2—Active Eavesdropping & Message Injection: An attacker may conduct spoofing attacks by actively injecting/modifying messages in the in-vehicle network. Moreover, by injecting well targeted messages, an adversary might be able to gain additional information from the system reaction via active eavesdropping.

Threats 1 and *2* are possible in the absence (or breaking) of data confidentiality, integrity, and authentication in in-vehicle networks. To address the threat of active and passive attacks, security primitives can be incorporated, such as encryption for providing confidentiality to discourage passive attacks and message authentication codes to discourage active attacks.

Threat 3—Key Extraction from Storage: The approaches that can be used to counter *Threats 1* and *2* typically use symmetric key cryptography because of less computation overhead as compared to public key cryptography. The symmetric key cryptography, however, requires a symmetric key which the previous works [12, 14] assume is stored in a secure memory. However, an adversary can conduct SCAs on memory to extract the stored symmetric key, which can compromise not only the single ECU or single vehicle but may also compromise a whole series of vehicle, because same series of ECUs tend to use the same symmetric key. To tackle *Threat 3*, a symmetric key needs to be generated during vehicle startup to prevent key extraction attacks from storage. Furthermore, the key needs to be refreshed periodically to prevent replay attacks. The proposed approach develops a solution for symmetric key generation and distribution for automotive systems.

3.3 Security

To provide resilience against the considered threat model (Section 3.2), we use AES-128 for providing message confidentiality and HMAC based on SHA-3 (Secure Hash Algorithm-3) for ensuring message integrity. The proposed approach uses "encrypt-and-MAC". The receiving node decrypts the message and compare the HMAC calculated on the receiving side with the one received from the sender. If the two HMACs are equal, the message is authentic otherwise the message has lost its integrity. Another key aspect of the proposed approach is ECU authentication and key establishment. Our proposed approach incorporates a

central security module (CSM), which is responsible for ECU registration, authentication, and symmetric key establishment. In Figure 1, CA is certificate authority that generates the necessary keys (i.e., public and private keys) for all ECUs and the CSM. The CA has its own public and private keys. The CA's public key is shared with all ECU nodes, whereas the private key is stored secretly. The CA can be automotive OEM.

4 Authentication and Secret Key Establishment

This section discusses the proposed certificate-based ECU authentication and symmetric key establishment mechanisms.

4.1 Certificate Generation

Each ECU $i \forall\, i \in \{1, 2, \ldots, n_E\}$, where n_E denotes the total number of ECUs on the in-vehicle bus, will have a public key k_{pub,E_i} and a private key k_{pr,E_i}, that is, $k_{E_i} = (k_{pub,E_i}, k_{pr,E_i})$. The CA is responsible generating these public and private keys for each ECU. The CA also generates the certificate for each ECU_i by combining the ECU's public key k_{pub,E_i} and its identity ID_{E_i} with the signature S_{E_i} generated over k_{pub,E_i} and ID_{E_i} using the private key of the CA $k_{pr,\mathrm{CA}}$. The ID of each ECU is assigned by the OEM (e.g., an ECU's serial number can serve as the ECU's ID).

4.2 ECU Authentication

In our approach, authentication of ECUs is done by certificate verification. By using $k_{pub,CA}$, we can verify if the signature is legitimate or not. The approach uses elliptic curve digital signature algorithm (ECDSA) [10] for the signature generation and verification.

4.3 Symmetric Key Establishment Protocol

After verifying all the ECUs, the CSM generates new symmetric key and distributes it to all ECUs, the process known as *key establishment*. This process is repeated after certain time period T to refresh the keys periodically. Figure 2 presents the proposed certificate-based protocol for symmetric key establishment. The proposed protocol comprises of five key steps as described below.

Step 1: At the vehicle start up, the CSM advertises/broadcasts its public key, ID, and certificate to all other ECUs in the vehicle.All ECUs verify the authenticity of CSM.

Step 2 (ECU Side): After CSM authentication, the ECU i which requires registration or authentication generates a random nonce r_i. For authentication of ECU i, its certificate and nonce is sent to the CSM in encrypted form because

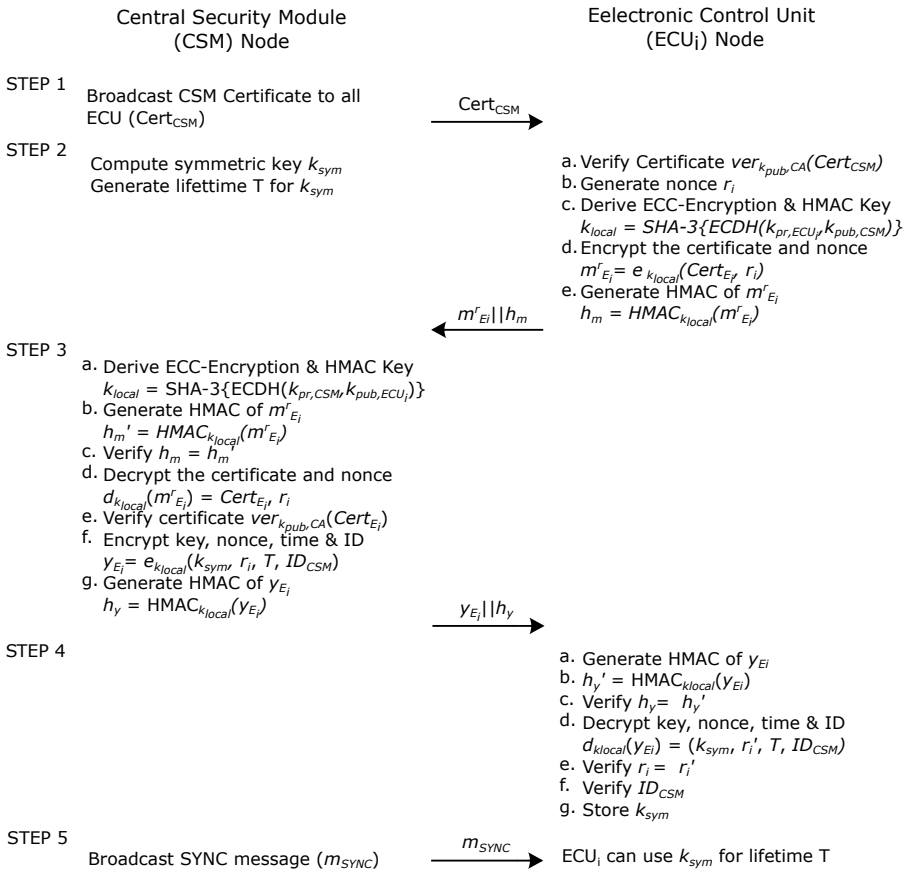

Fig. 2. Proposed symmetric key establishment protocol.

only the CSM (and no other malicious ECU) should be able to retrieve the certificate and nonce of the ECU i. Hence, a common secret is generated using the private key of ECU i and the public key of CSM, which is then transformed to obtain local key k_{local}. The local key is used to encrypt the message (ECU i certificate and r_i) and generate HMAC of the message h_m. The encrypted message $m^r_{E_i}$ and h_m are concatenated ($m^r_{E_i}||h_m$) and sent to the CSM. Here, HMAC is appended to the message to maintain the message integrity.

Step 2 (CSM Side): While ECUs are authenticating the CSM, the CSM generates a new symmetric keys k_{sym} and the lifetime T for k_{sym}.

Step 3: After receiving the request message $m^r{}_{E_i}$ along with the hash h_m, the CSM verifies the HMAC to find out the integrity of the message. To calculate the HMAC at CSM, the CSM needs to have local key k_{local} which was used at ECU side (step 2). The local key is again generated using the common secret of private key of CSM and public key of ECU i. After verification of HMAC, the CSM decrypts the message, and verifies the certificate $Cert_{E_i}$ using the public key of CA. After this verification, the CSM encrypts the generated symmetric keys k_{sym} (in step 2 at CSM), nonce r_i, lifetime T, and ID of the CSM ID_{CSM}, denoted as message $y_{E_i}$with k_{local}. The CSM also calculates the hash h_y of y_{E_i} and then send this response message y_{E_i} concatenated with h_y back to the ECU i.

Step 4: The received message packet y_{E_i} is tested for message integrity by verifying the HMAC, and then decrypted by the ECU i using its local key (generated in step 2). The ECU i also verifies the random nonce r_i received from the CSM and its original random nonce. Furthermore, the ECU i verifies ID_{CSM} with the one obtained from the CSM certificate in Step 1. After successful verification, the ECU accepts the symmetric key k_{sym}.

Step 5: Finally, the CSM broadcast SYNC (synchronization) message instructing all ECUs to use the newly generated symmetric key for time T.

Each ECU starts communicating with other nodes using this newly established key for symmetric encryption and HMAC for regular operation. After time period T, each ECU requests for a new key for key refreshment by sending a message request as in step 2 ECU side (Figure 2). The CSM then distributes new symmetric keys to ECUs as shown in Figure 2. The proposed protocol uses ECC for the computation of steps 2, 3 and 4 explained above. The algorithm used in step 2(c, d, and e) and step 3(a, b, c, d) are based on the steps of elliptic curve integrated encryption scheme (ECIES) [10].

5 Case Study: Steer-by-Wire Subsystem

A SBW system replaces the heavy mechanical steering column with an electronic system. The SBW subsystem provides two functions: front axle (FA) control (FAC) and hand-wheel (HW) force feedback (HWF). The FAC controls the wheel direction according to hand-wheel, whereas HWF provides the mechanical like

feedback to hand-wheel. The rotation on hand-wheel is sensed by hand-wheel sensors and sensed values are fed as the input to HW sensor (HWS) ECU1. HWS ECU1 processes the information to determine the commands for the front axel actuator (FAA) ECU1, which are then sent through the in-vehicle network (e.g., CAN FD bus) to the FAA ECU1. The FAA ECU1 processes the received CAN FD packet to extract the commands sent from HWS ECU1 and then turns the actuators accordingly to rotate the wheels. In this work, we only focus on FAC part to compute the response time and error resilience of our proposed approach.

The delay between the driver's request at HWS and the corresponding response at FAA has significant impact on the reliability of SBW subsystem. The end-to-end delay/response time (τ_r) is regarded as a quality of service (QoS) metric, however, it becomes a reliability metric, which can be defined in terms of *behavioral reliability*, if this delay exceeds a critical threshold value ${\tau_r}^{max}$ as the driver can lose control of the vehicle beyond this threshold. The behavioral reliability is the probability that the worst-case response time is less than the critical threshold. This threshold value is defined by automotive OEMs. The response delayτ_r time is given by following equation

$$\tau_r = \tau_p + \tau_m + \tau_s, \tag{1}$$

where, τ_p is pure delay, τ_m is mechatronic delay, and τ_s sensing delay. The mechatronic delay is introduced by the actuators (electric motor in SBW system case). The sensing delay is the delay introduced due to sensing and sampling of measurements. Since τ_m and τ_s can be bounded by a constant and can vary for different kind of sensors and actuators, we focus on τ_p for our analysis [16]. Here τ_p includes ECUs computational delay for processing the control algorithm, computational delay for processing the incorporated security and safety primitives, and transmission delay including bus arbitration. To ensure safe operation of the vehicle as governed by behavioral reliability, τ_p should be less than or equal to the maximum tolerable pure delay τ_p^{max}, that is, $\tau_p \leq \tau_p^{max}$. Mathematically, τ_p for the FAC function can be written as,

$$\tau_p = rcc1 \cdot \tau_{hws}^{ecu1} + rtc \cdot \tau_{bus} + rcc2 \cdot \tau_{faa}^{ecu1} \leq \tau_p^{max}, \tag{2}$$

where τ_{hws}^{ecu1} and τ_{faa}^{ecu1} denote the computation time at HWS-ECU1 and FAA-ECU1, respectively; τ_{bus} represents the transmission time for a message on in-vehicle bus from HWS-ECU1 to FAA-ECU1; $rcc1$ and $rcc2$ represent the number of recomputations that are needed to be done at HWS-ECU1 and FAA-ECU1, respectively; and rtc represents the number of retransmissions required for an error-free transmission of a secure message over in-vehicle bus.

The pure delay τ_p for FAC can be considered for two cases: (i) delay during regular operation τ_p^R, and (ii) delay during key refreshment operation τ_p^K.

1. Delay During Regular Operation: The delay during regular operation comprises of encryption/decryption and HMAC computation delay at the sending and receiving nodes plus the message transmission delay on in-vehicle network, that is,

$$\tau_p^R = rcc1 \cdot \tau_{hws}^{ecu1,R} + rtc \cdot \tau_{bus} + rcc2 \cdot \tau_{faa}^{ecu1,R} \leq \tau_p^{max}, \quad (3)$$

where $\tau_{hws}^{ecu1,R}$ and $\tau_{faa}^{ecu1,R}$ denote the computation time at HWS ECU1 and FAA ECU1, respectively, during regular operation.

2. Delay During Key Refreshment Operation: The delay during key refreshment operation comprises of the delay during regular operation and the delay due to key refreshment operation. The τ_p^K for the CSM refreshing the key for HWS ECU1 (Figure 2) can be written as

$$\tau_p^K = rcc1 \cdot \tau_{hws}^{ecu1,R} + rtc \cdot \tau_{bus} + rcc2 \cdot \tau_{faa}^{ecu1,R} + rcc3 \cdot \tau_{hws}^{ecu1,K} \leq \tau_p^{max}, \quad (4)$$

where $\tau_{hws}^{ecu1,K}$ denotes the delay for key refreshment at HWS ECU1 and $rcc3$ represents recomputations required to yield an error-free result at HWS ECU1.

6 Result and Discussion

6.1 Experimental Setup

For ECC implementation in key establishment protocol, we have used a prime field curve P-192 from NIST [3]. We have used AES-128 for providing confidentiality and SHA-3 256 for providing message integrity. We have implemented the proposed approach that includes ECU authentication, and key establishment protocol in NVIDIA's Jetson TX2 platform, which has four 64-bit ARM Cortex-A57 cores running *Ubuntu 14.04.4 LTS* at $2.0GHz$. The future generations of automotive ECUs are expected to possess comparable compute capabilities [13]. The code for providing confidentiality, integrity, authentication, and key establishment is written in *C* language. *OpenMp* is used for FT-RMT implementation. For the SBW system, we assume the steering wheel sensor sampling rate of 400Hz, which corresponds to the sampling/sensing delay τ_s of $2.5ms$ [7]. We simulate our SBW system in Vector CANoe [15].

6.2 Timing Analysis

We have measured the timing response of the key-establishment process in different FT settings. For timing analysis, we inject soft errors at different points in the program. Our approach emulates bit flipping in the program/memory due to noise and/or radiation from the environment.

Performance Analysis of Key Establishment: Table 1 shows execution times of different steps of the key establishment protocol (Figure 2) in various FT operational modes assuming no error in the program/computations. Table 1 does not depict computation time for Step 1 and Step 5 because in these steps, CSM broadcasts a precomputed message packet, which does not require computation. Results indicate that the overhead incurred by various FT approaches is insignificant (less than 1%) because computation time of the protocol steps is large as compared to the overhead.

Table 1. Performance Analysis of Key Establishment

Operational Modes	Algorithm Steps (time in μsec)			
	Step 2		Step 3	Step 4
	ECU Side	CSM Side	CSM Side	ECU Side
NFT	5214.63	216.51	5224.71	1563.63
FT-RMT	5259.53	219.80	5226.57	1565.86
FT-RMT-QED	5306.83	220.42	5300.08	1571.29
FT-RMT-CP	5352.41	219.71	5300.93	1585.75

Table 2. Effect of Errors on Performance (For Step 2 ECU Side)

Error Location	Operational Modes (time in μsec)		
	NFT	FT-RMT-QED	FT-RMT-CP
@ Verification of Certificate	10223.79	8483.39	8760.96
@ Key generation for ECC	10191.91	9951.42	6837.83
@ ECC-Encryption	10202.22	10153.84	5692.02
@ ECC-HMAC	10323.72	10224.69	5747.74

Effect of Errors on Performance: Table 2 shows the execution time of Step 2 on ECU side of the key establishment protocol (Figure 2) in the presence of errors. In this experiment, a single soft error is injected at different points in the program. If a single error occurs in FT-RMT mode, the entire function is recomputed to rectify the error as the error is detected at the end of the computation in FT-RMT. This results in the computation time with error to be 2× of the computation time without the error. In FT-RMT-QED, if the error occurs during the start of program/computation, the computation overhead is much less as compared to FT-RMT. However, if error occurs near the end of the computation, the computation overhead is almost the same as that of FT-RMT. The FT-RMT-CP only repeats the execution of those parts where the error has occurred and thus the recomputation overhead depends on the size of code between the two checkpoints. Results indicate that FT-RMT-CP performs best as compared to other FT approaches and considerably reduces the computation time when error occurs near the end of computation. For example, FT-RMT-CP provides 44.21% and 43.94% reduction in computation time as compared to NFT and FT-RMT-QED, respectively, when the error occurs in ECC encryption of Step 2 on ECU side.

Performance Analysis of Checkpointing: Figure 3 shows the execution time of Step 3 of the key establishment protocol (Figure 2) with varying number of checkpoints and multiple errors injected at different points in the computation. The number of checkpoints varies from one to six whereas the number of errors introduced varies from one to five. The errors have been uniformly distributed over the program in order to provide a fair evaluation. Results indicate that as the number of errors increases, the computational time increases linearly.

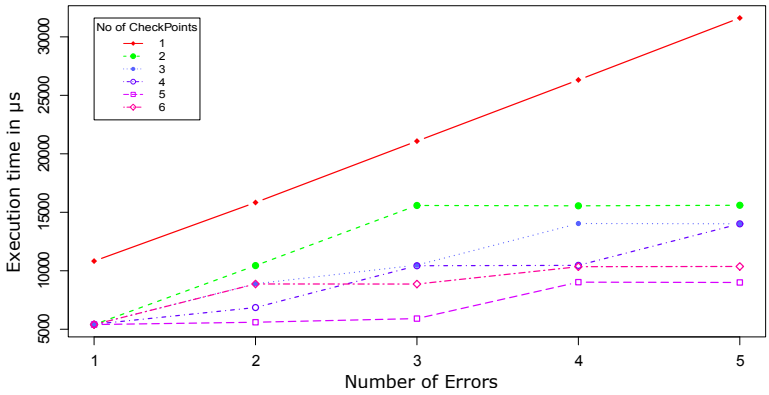

Fig. 3. Effect of checkpoints and errors on performance.

Furthermore, as the number of inserted checkpoints increases, the time required to rectify the error decreases and thus the overall computation time decreases.
Performance Analysis of Regular Operation: Table 3 shows the temporal performance of regular operation, which comprises of encryption/decryption of 2 blocks of 128-bit AES and a 256-bit HMAC, at sender and receiver nodes. The results show that encryption and HMAC at the sender node takes longer time as compared to decryption and HMAC at the receiver node. This is because at the receiver node, the decryption computation is accelerated by using precomputed tables. We observe that FT-RMT at the sender node has 16.2% overhead and the receiver node has 25.2% overhead.

Table 3. Timing of Regular Operation at Sender and Receiver Nodes

Operational Mode	Sender Node (μsec)	Receiver Node (μsec)
NFT	130	87
FT-RMT	151	109

6.3 Feasibility Analysis

We have conducted the feasibility analysis of the proposed approach (including the key establishment protocol) to verify that the proposed mechanisms do not violate the real-time constraints of automotive CPS. From extrapolation of QoS score $\mathcal{S}$ versus τ_p [16], we determine that for $\mathcal{S}$ of 11.08, the critical limit for τ_p is 16 ms.

Regular Operation: As shown in Table 3, the computational time at the sender node for encryption (2-blocks of 128-bit) and HMAC (256-bit digest) is 0.151 ms,

and the computational time at the receiving node for decryption and HMAC is 0.109 ms. The message transmission time using CAN FD obtained through Vector CANoe simulations is 0.12 ms (for a packet of 512 bits) [14]. The total computational and transmission delay without error is 0.38 ms (i.e., 0.151 ms + 0.109 ms + 0.12 ms) for one 512-bit packet of CAN FD (payload of CAN FD is 64 bytes). Furthermore, in a period of 16 ms at least 6 readings are taken by the HW sensor as $\lfloor \tau_p^{max}/\tau_s \rfloor = \lfloor 16ms/2.5ms \rfloor = \lfloor 6.4 \rfloor = 6$. Considering one block of AES can store the reading of one sample, 3 CAN FD frames (as 1 frame contains two AES blocks and thus can hold 2 sensor readings) need to be transmitted within a period of 15 ms without losing any sample value. We note that for CAN protocol, the encrypted and HMAC-ed message transmission would require multiple CAN frames [12].

To resolve the errors in computation, FT-RMT performs recomputation and in case of errors in transmission, erroneous packets are retransmitted. For this study, we assume that in one end-to-end communication, at most two errors can occur in each component/node (i.e., maximum two errors at the sender node, two errors at the receiver node, and two errors in transmission) in the worst case. Experimental results reveal that the total time taken to resolve (i.e., recomputation in case of computation errors and retransmission in case of transmission errors) two errors occurring in each of the components (sender, receiver, transmission) is 0.76 ms. Hence, within τ_p^{max} of 15 ms and for 3 packets, time taken to compute and resolve at most 2 errors is 2.28 ms. These results verify the feasibility of regular operation (Section 5) of SBW system using the proposed approach.

Key Establishment and Refreshment: We also measure the timing of key establishment and refreshment protocol. Considering that each of the communication messages between the ECU and the CSM in the key establishment protocol can be accomplished with one CAN FD packet, then using results from Table 1, the time taken for the key establishment protocol using FT-RMT-CP can be calculated as 0.12 ms (Step 1 communication) + 0.22 ms (Step 2 CSM side) + 5.35 ms (Step 2 ECU side) + 0.12 ms (Step 2 communication) + 5.3 ms (Step 3) + 0.12 ms (Step 3 communication) + 1.58 ms (Step 4) + 0.12 ms (Step 5 communication) = 12.93 ms. The time taken for key refreshment is 12.81 ms (12.93 ms - 0.12 ms = 12. 81 ms) since key refreshment takes Steps 2 to 5 in Figure 2. These results verify that the key refreshment can take place along with the regular operation within the time constraints specified by desired QoS, that is, 12.81 ms + 2.28 ms = 15.09 ms $\leq$ 16 ms, even in the presence of faults. Hence, these results verify the feasibility of the proposed key establishment and refreshment protocol for automotive CPS.

7 Conclusion

In this paper, we have proposed a safe and secure approach for secret key establishment and operation in automotive cyber-physical systems (CPS). The proposed approach realizes authentication and symmetric key establishment using certificates and ECC. To incorporate safety, the proposed approach leverages

multicore ECUs and various FT approaches such as FT-RMT, FT-RMT-QED, and FT-RMT-CP. Results reveal that FT-RMT-CP performs best as compared to other FT approaches. Furthermore, results verify that our proposed approach provides confidentiality, integrity, authentication, and secret key establishment in intra-vehicle networks without violating real-time constraints of the vehicle even in the presence of errors in computation and transmission.

References

1. Beckschulze, E., Salewski, F., Siegbert, T., Kowalewski, S.: Fault handling approaches on dual-core microcontrollers in safety-critical automotive applications. In: International Symposium On Leveraging Applications of Formal Methods, Verification and Validation. pp. 82–92. Springer (2008)
2. Fassak, S., El Idrissi, Y.E.H., Zahid, N., Jedra, M.: A Secure Protocol for Session Keys Establishment between ECUs in the CAN Bus. In: Proc. of IEEE International Conference on Wireless Networks and Mobile Communications (WINCOM). Rabat, Morocco (November 2017)
3. FIPS, P.: 186-4. Digital Signature Standard (DSS) (2013)
4. Huang, T., Zhou, J., Wang, Y., Cheng, A.: On the security of in-vehicle hybrid network: Status and challenges. In: International Conference on Information Security Practice and Experience. pp. 621–637. Springer (2017)
5. Ishtiaq Roufa, R.M., Mustafaa, H., Travis Taylora, S.O., Xua, W., Gruteserb, M., Trappeb, W., Seskarb, I.: Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In: 19th USENIX Security Symposium, Washington DC. pp. 11–13 (2010)
6. ISO: ISO 26262-1:2018: Road vehicles – Functional safety. https://www.iso.org/standard/68383.html (December 2018), Last visited on June 7, 2019
7. Klobedanz, K., Kuznik, C., Thuy, A., Mueller, W.: Timing modeling and analysis for autosar-based software development: a case study. In: Proceedings of the Conference on Design, Automation and Test in Europe. pp. 642–645. European Design and Automation Association (2010)
8. Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., et al.: Experimental security analysis of a modern automobile. In: Security and Privacy (SP), 2010 IEEE Symposium on. pp. 447–462. IEEE (2010)
9. Lin, C.W., Sangiovanni-Vincentelli, A.: Cyber-security for the controller area network (can) communication protocol. In: Cyber Security (CyberSecurity), 2012 International Conference on. pp. 1–7. IEEE (2012)
10. Menezes, A., Hankerson, D., Vanstone, S.A.: Guide to Elliptic Curve Cryptography. Springer (2004)
11. Miller, C., Valasek, C.: Remote exploitation of an unaltered passenger vehicle. Black Hat USA 2015, 91 (2015)
12. Munir, A., Koushanfar, F.: Design and Analysis of Secure and Dependable Automotive CPS: A Steer-by-Wire Case Study. IEEE Transactions on Dependable and Secure Computing (TDSC) (June 2018)
13. NVIDIA: NVIDIA Self-Driving Cars. https://www.nvidia.com/en-us/self-driving-cars/, Last visited on September 5, 2019
14. Poudel, B., Munir, A.: Design and Evaluation of a Reconfigurable ECU Architecture for Secure and Dependable Automotive CPS. IEEE Transactions on Dependable and Secure Computing (TDSC) (November 2018)

15. Vector: ECU Development and Test with CANoe. https://www.vector.com/us/en-us/products/products-a-z/software/canoe/, Last visited on June 3, 2019
16. Wilwert, C., Navet, N., Song, Y.Q., Simonot-Lion, F.: Design of Automotive X-by-Wire Systems. The Industrial Communication Technology Handbook CRC Press (2005)
17. Wolf, M., Gendrullis, T.: Design, implementation, and evaluation of a vehicular hardware security module. In: International Conference on Information Security and Cryptology. pp. 302–318. Springer (2011)