Design and Analysis of Secure and Dependable Automotive CPS: A Steer-by-Wire Case Study

Arslan Munir, *Senior Member, IEEE* and Farinaz Koushanfar, *Senior Member, IEEE*

Abstract—The next generation of automobiles (also known as cybercars) will increasingly incorporate electronic control units (ECUs) in novel automotive control applications. Recent work has demonstrated the vulnerability of modern car control systems to security attacks that directly impacts the cybercar's physical safety and dependability. In this paper, we provide an integrated approach for the design of secure and dependable automotive cyber-physical systems (CPS) using a case study: a steer-by-wire (SBW) application over controller area network (CAN). The challenge is to embed both security and dependability over CAN while ensuring that the real-time constraints of the automotive CPS are not violated. Our approach enables early design feasibility analysis of automotive CPS by embedding essential security primitives (i.e., confidentiality, integrity, and authentication) over CAN subject to the real-time constraints imposed by the desired quality of service and behavioral reliability. Our method leverages multicore ECUs for providing fault tolerance by redundant multi-threading (RMT) and also further enhances RMT for quick error detection and correction. We quantify the error resilience of our approach and evaluate the interplay of performance, fault tolerance, security, and scalability for our SBW case study.

Index Terms—Automotive, cyber-physical systems, multicore, security, fault tolerance, controller area network, behavioral reliability, x-by-wire, steer-by-wire

1 Introduction and Motivation

Modern automobiles may consist of more than 100 electronic control units (ECUs) to implement various distributed control applications. The next generation of automobiles (also known as cybercars) will further escalate the proliferation of ECUs to enable new and exciting control and infotainment applications. The most prevalent protocol for communication among the ECUs in these distributed automotive cyber-physical systems (CPS) applications is controller area network (CAN), which has already been implemented in billions of devices and long-lived transportation systems, and thus likely to stay for many years to come. Emerging automotive CPS applications include *x-by-wire*, for example, break-by-wire, steer-by-wire (SBW), where the electronic controllers substitute the traditional mechanical and/or hydraulic systems.

The realization of emerging automotive CPS applications require addressing dependability and security issues. The automotive CPS applications have stringent dependability requirements as stipulated by ISO 26262 [1], which require tolerance of at least one critical fault without loss of functionality [2]. Meeting these automotive CPS dependability requirements poses various challenges. The miniaturization of electronic devices results in a significant increase in manufacturing variability and runtime errors creating dependability issues. Harsh operating environments coupled with external noise and radiation render automotive electronic systems vulnerable to *permanent*, *transient*, and *intermittent* faults. While permanent faults can impair or stop the correct functionality of the system, *soft errors* induced by transient faults can remarkably reduce the system availability and correct functionality [3]. The intermittent faults, on the other hand, oscillate between quiescent and active states. When the fault is quiescent, the component functions correctly, and when the fault is active, the component malfunctions. An example for an intermittent fault is a loose electrical connection [3]. Furthermore, automotive electronic components are susceptible to security vulnerabilities. Since CAN messages are transmitted in plain-text format, intruders may be able to gain access and even alter the CAN messages creating security threats. These security threats are exacerbated by the increasing integration of cybercar applications with external entities such as consumer electronics, other vehicles, and networks.

The cyber-physical attributes of modern automotive systems [4] directly couple security vulnerabilities to the automobile's physical safety and dependability: an attacker who is able to infiltrate any ECU can potentially circumvent many safety-critical systems while completely ignoring the driver input [5]. Simultaneous integration of security and dependability in automotive CPS is challenging. One of 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. Timeliness (meeting timing constraints) is perceived as the system's quality of service (QoS), and is commonly considered a performance measure. We emphasize that the automotive CPS application's QoS must also be considered as a dependability measure that can impact the system's availability and safety, beyond

- *A. Munir is with the Department of Computer Science, Kansas State University, Manhattan, KS 66506. E-mail: amunir@ksu.edu.*
- *F. Koushanfar is with the Department of Electrical and Computer Engineering, University of California, San Diego, CA 92093. E-mail: fkoushanfar@ucsd.edu.*

Manuscript received 8 Dec. 2016; revised 5 June 2018; accepted 9 June 2018. Date of publication 13 June 2018; date of current version 9 July 2020. (Corresponding author: Arslan Munir.)
Digital Object Identifier no. 10.1109/TDSC.2018.2846741

a certain *critical threshold* as the driver can totally lose the control of his/her car beyond that critical threshold [6]. This temporal performance impact on safety introduces the notion of *behavioral reliability*, which is defined as the probability that the system's worst-case response time is less than the critical threshold [6]. Limited resources (memory, processing), limited bandwidth, cost, and flexibility (ability of a solution to adapt to different performance, cost, and fault tolerance (FT) requirements) constraints further exacerbates the challenges to accommodate security and dependability in automotive CPS design.

The security vulnerabilities and dependability requirements of automotive CPS warrant inclusion of security and dependability approaches in the design. Although earlier work in this area has addressed certain aspects of security and dependability (e.g., [6], [7]), the interplay between performance, security, and safety for SBW applications using CAN has not yet been explored. To overcome the limitations of earlier work, we provide a new comprehensive approach to the design and analysis of secure and dependable automotive CPS with SBW application as a case study. Our main technical contributions are as follows:

- We propose a novel integrated approach that enables early design phase feasibility and performance analysis for devising secure and dependable automotive CPS. The approach is demonstrated on a SBW application case study.
- Embedding and analyzing security primitives over CAN while adhering to the stringent real-time constraints for safety-critical automotive CPS applications. Our approach uses advanced encryption standard (AES) for providing message confidentiality and hash-based message authentication code (HMAC) for ensuring message authenticity and integrity.
- Proposal to include multicore ECUs (dual-core and triple-core ECUs instead of conventional single-core ECUs) to meet the FT requirements (tolerating one permanent fault, multiple soft-errors, and intermittent faults) under stringent cost and real-time constraints by redundant multi-threading (RMT). We denote this approach as *FT-RMT*. We further enhance RMT by exploiting quick error detection (QED) [8], and denote this approach as *FT-RMT-QED*.
- Extension of the FT-RMT-QED on a triple modular redundant (TMR) architecture to enable quick error detection and correction (QEDC). We denote this enhanced FT approach as *FT-RMT-TMR-QED*).[1]
- Quantification of error resilience of our proposed FT approaches by calculating the number of tolerable computational and transmission errors that are manifested as response time variations under the stringent QoS constraints.
- Analyzing the scaling properties of our secure and dependable approach for automotive CPS design using SBW system as a case study for different CAN bus load conditions and message priority assignments.

1. Fig. 1 depicts our proposed dependable and secure approach and also rementions all the relevant acronyms for reference.

The scope of this work is to ensure that simultaneous integration of security and dependability primitives in automotive CPS does not violate stringent real-time constraints imposed by the desired QoS. Although our proposed approach leverages existing security and dependability techniques, the novelty of our work lies in the simultaneously integration of both dependability and security, and the performance and feasibility analysis of our integrated approach over CAN. The evaluation metric for our feasibility analysis is response time, which is constrained by the desired QoS. The remainder of this paper is organized as follows. Section 2 provides a summary of related work. Section 3 presents our proposed secure and dependable approach for cybercars design. Section 4 elaborates our SBW system case study in relation to QoS and behavioral reliability. Evaluation results are presented in Section 5. Finally, Section 6 concludes our study.

2 Related Work

There exists works in literature that address certain aspects of automotive security and dependability. This section discusses key previous works related to security, dependability, and analysis of automotive x-by-wire systems.

2.1 Dependability

Several earlier works explored dependability for automotive embedded systems. Beckschulze et al. [9] investigated FT approaches based on dual-core microcontrollers. Baleani et al. [10] discussed various FT architectures for automotive applications including lock-step dual processor architecture, loosely-synchronized dual processor architecture, and triple modular redundant architecture. Although, the work compared various FT architectures' costs based on the area estimates, the study did not quantify the architectures' FT capabilities subject to real-time constraints. Rebaudengo et al. [11] studied soft-error detection through software-based FT techniques. The authors described an approach to detect soft-errors by automatically introducing data and code redundancy into an existing program written in a high-level language. The proposed approach, however, incurred an average performance penalty of 5×, which may not be acceptable for automotive CPS.

Some previous work explored dependability benchmarking for automotive systems. Ruiz et al. [12] proposed a dependability benchmark for engine control applications. The authors compared the system behavior in the absence of faults (golden run) with the system behavior in the presence of faults. Although dependability benchmarks quantify an application's dependability, the benchmarks do not enhance the application's FT. Hence, the use of dependability benchmarks to assess system dependability is complementary to our work whose focus is to provide FT.

2.2 Security

Security for automotive embedded systems has been studied in literature. Checkoway et al. [13] analyzed the external attack surface of a modern automobile. The authors discovered that remote exploitation is possible via a broad range of attack vectors such as mechanics tools, CD players, bluetooth, and cellular radio. Hoppe et al. [14] explored security

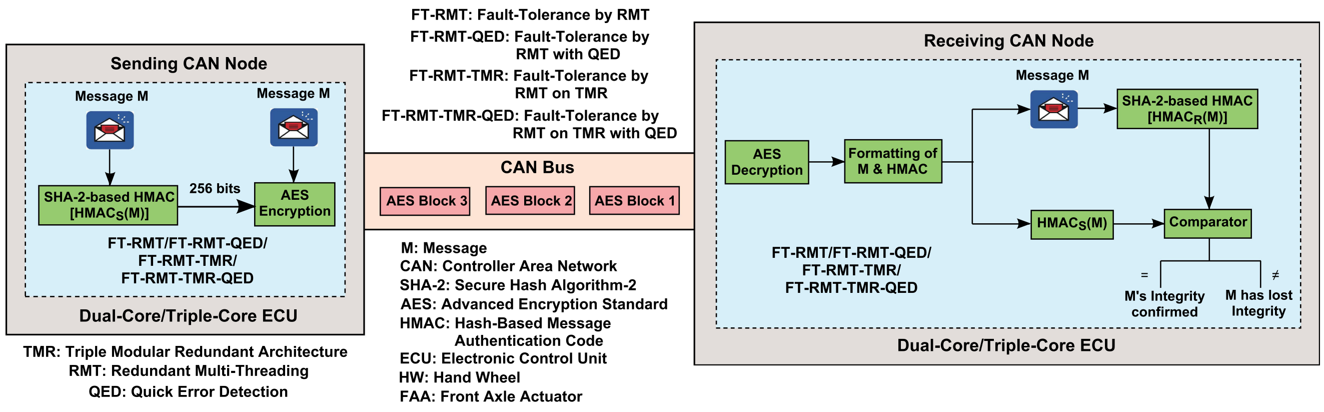
FT-RMT: Fault-Tolerance by RMT
FT-RMT-QED: Fault-Tolerance by RMT with QED
FT-RMT-TMR: Fault-Tolerance by RMT on TMR
FT-RMT-TMR-QED: Fault-Tolerance by RMT on TMR with QED
Sending CAN Node
Message M
Message M
SHA-2-based HMAC [HMAC$_S$(M)]
256 bits
AES Encryption
FT-RMT/FT-RMT-QED/ FT-RMT-TMR/ FT-RMT-TMR-QED
Dual-Core/Triple-Core ECU
CAN Bus
AES Block 3
AES Block 2
AES Block 1
Receiving CAN Node
Message M
SHA-2-based HMAC [HMAC$_R$(M)]
AES Decryption
Formatting of M & HMAC
HMAC$_S$(M)
Comparator
=
≠
M's Integrity confirmed
M has lost Integrity
FT-RMT/FT-RMT-QED/ FT-RMT-TMR/ FT-RMT-TMR-QED
Dual-Core/Triple-Core ECU
TMR: Triple Modular Redundant Architecture
RMT: Redundant Multi-Threading
QED: Quick Error Detection
M: Message
CAN: Controller Area Network
SHA-2: Secure Hash Algorithm-2
AES: Advanced Encryption Standard
HMAC: Hash-Based Message Authentication Code
ECU: Electronic Control Unit
HW: Hand Wheel
FAA: Front Axle Actuator

Fig. 1. A dependable and secure approach for automotive CPS design.

and privacy issues in future automotive systems. The authors presented an attack on a gateway ECU (the gateway ECU is an ECU that connects various automotive subnetworks, such as powertrain, body, and chassis) that enabled an attacker to sniff internal communication even beyond subnetwork borders. Rouf et al. [15] investigated security and privacy vulnerabilities of modern automotive systems with wireless tire pressure monitoring system (TPMS) as a case study. Results indicated that eavesdropping was possible at a distance of around 40 meters from a passing vehicle exploiting TPMS messages. These previous works identified security vulnerabilities in automotive systems, but did not present an integrated approach for designing secure and dependable automotive systems.

Chávez et al. [7] incorporated confidentiality service in CAN based on RC4. The authors measured the clock cycles required for encrypting different CAN frame data sizes. However, the authors did not consider FT aspects or CAN message priorities while analyzing the security over CAN. Groll et al. [16] studied authenticity and confidentiality issues in CAN for safety-critical applications. Results revealed that asymmetric cryptography, owing to excessive time overhead, could only be feasible for initial key distribution process whereas symmetric cryptography could enable secure communication between CAN nodes.

Various prior works [17], [18], [19] studied integration of message authentication codes (MACs) in CAN data frames to secure in-vehicle communication. In [17], the authors used a compound MAC added to the CAN payload which could be used to detect and possibly recover from injection and modification attacks in the in-vehicle networks. The proposed scheme calculated MAC over four consecutive CAN messages and the resulting MAC was divided into four 16-bit blocks and transmitted in the cyclic redundancy code (CRC) field of the next four CAN messages. The protocol required a total of eight CAN messages for verification, which introduced a delay in the data integrity and data authentication verification. Furthermore, the scheme did not provide any protection against reply attack and in case of MAC verification failure, the incorrect message could not be identified individually.

The work in [18], [19] proposed CAN security mechanisms based on MACs and counters that could prevent both masquerade and replay attacks. Kang et al. [20] proposed an authentication protocol for ECUs based on one-way hash chain. To mitigate the hash collision problem, the proposed work used an attack-resilient tree structured algorithm in the authentication protocol. However, these works did not consider message confidentiality and FT.

Our prior work [21] was the first, to the best of our knowledge, to propose simultaneous integration of security primitives (confidentiality, integrity, and authentication) using AES-128 and SHA-2-based HMAC as well as dependability primitives leveraging multicore ECUs. Furthermore, our prior work demonstrated the feasibility of the proposed approach by implementation on a multicore processor and Vector CANoe simulations. Later on, Wu et al. [22] proposed a security protocol for CAN system based on AES-128 encryption and HMAC function. The proposed protocol further compressed CAN messages with an encryption algorithm. Experimental results obtained using Vector CANoe demonstrated the feasibility of the proposed approach. However, the work did not elaborate on the HMAC algorithm being used by the approach nor presented any analytical model on QoS and behavioral reliability. Additionally, the implementation of the proposed approach on a real processor was missing in the work. Moreover, the work did not consider FT of messages during computation and transmission.

In order to overcome the deficiencies of existing work on automotive security and dependability, this work builds on our prior work [21] and makes significant enhancements on concepts, modeling, experimental results and analysis as outlined in Section 1. In particular, this work proposes a TMR architecture to enable QEDC, includes enhanced analytical modeling of our proposed approach, QoS, and behavioral reliability, and extended experimental results and analysis.

2.3 Analysis of X-by-Wire Systems

Few previous works investigated automotive x-by-wire systems. Fredriksson [23] advocated the use of CAN for safety-critical systems including x-by-wire. The work, however, did not present any case study or evaluation results for verifying the feasibility of x-by-wire systems over CAN. Schweppe et al. [24] studied an active braking system for automobiles assisted by vehicle-2-X communication. The active braking system deduced brake recommendations based on relative position, speed, and acceleration of the vehicles. The authors proposed a hardware security module (HSM) for hardware acceleration of signature generation and verification for vehicle-2-X communication. Although the work compared the performance of the HSM building blocks for automotive systems in pure software and hardware accelerated versions, the work did not consider the interplay between performance, security, FT, message priorities, and scalability. Ringler et al. [25] studied a brake-by-wire system over a time-triggered protocol in which all the activities were initiated with reference to a globally synchronized time. The work, however, did not consider security issues of the brake-by-wire system nor the work evaluated the end-to-end response time.

Wilwert et al. [6] presented an approach for evaluating the temporal performance and behavioral reliability of an SBW system considering the delay variation introduced by network transmission errors. The authors evaluated the maximum tolerable system response time and the impact of this response time on an automotive system's QoS for a time division multiplex access (TDMA) communication protocol. However, the work considered only communication errors on the bus and not the computational errors in ECUs. Furthermore, the authors did not consider the system vulnerability to security attacks. In our work, we also consider security overhead while analyzing the system response time. Moreover, the previous works did not analyze the scalability aspects whereas our work analyzes the SBW system response time under different load conditions and message priority assignments.

Statistical methods for analyzing CAN message response times have been studied in literature. Zeng et al. [26] computed the probability distribution of CAN message response times using statistical analysis when only partial information about the functionality and architecture of a vehicle was available. Although the statistical model could predict CAN message response times, however, many parameters required by the model, such as queueing lengths and the time between consecutive higher priority message bursts, are difficult to determine. Results indicated that the response time estimation quality of statistical models for high-priority messages was in general worse than the low-priority messages. The limitation of statistical methods for high-priority messages motivated us for rigorous simulations of SBW systems to provide better response time estimates than the statistical methods.

3 A Secure and Dependable Approach for Automotive CPS Design

The design of secure and dependable automotive CPS is challenging because of limited resource budgets (e.g., memory and processing, bandwidth, cost) and real-time constraints [27]. In particular, the inclusion of dependability and security primitives, and protocols must not violate the real-time constraints. The scope of our approach is confined to error detection and correction for dependability, and to provide confidentiality, integrity, and authentication for security. Fig. 1 provides an overview of our dependable and secure approach for automotive CPS design. The figure shows the operations involved at both the sending and receiving CAN nodes to ensure dependability and security. This section elaborates the dependability and security primitives adopted in our approach.

3.1 Dependability

To assist the design and production of safe automotive systems, International Organization for Standardization (ISO) has developed a functional safety standard, viz., ISO 26262 [1]. Many of the current automotive systems consist of single-core ECUs that have difficulty meeting both performance and dependability requirements of automotive CPS simultaneously as well as present challenges running both safety-critical and non-safety-critical software on the same ECU without interference. To exploit the technological advancements in silicon and accompanied low-cost of single-chip solutions, our approach leverages multicore ECUs to provide FT. Our approach is applicable to both dual-core and triple-core ECUs. We note that our multicore-based FT approach does not provide resilience against common mode failures or single point of failure (e.g., a power supply failure), however, these common mode failures can be compensated by having redundant or backup modules (e.g., a backup power supply). We consider RMT-based FT because the approach does not incur high cost required in other custom hardware-based FT approaches, such as lock-step FT architectures, albeit at the expense of some increased code size and reduced performance. We consider generic dual-core/triple-core architectures that meet the low cost as well as flexibility requirements since the architectures can adapt to various performance and FT requirements for an application (the dependability approaches based on specific hardware architectures, such as lock-step processor architectures, offer limited flexibility). Our approach utilizing dual-core ECUs is suitable for applications with stringent cost constraints whereas triple-core ECUs are more befitting for applications with relatively less stringent cost constraints.

For dual-core ECUs, the FT configuration can be either FT-RMT or FT-RMT-QED. The FT-RMT executes safety-critical computations on redundant threads and detects an error at the end of computation if there is a mismatch between the two threads' output. The FT-RMT-QED enhances FT-RMT with QED [8]. In the 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. Error detection latencies in the FT-RMT-QED approach are configurable and can range from a few cycles to a few thousand cycles depending on the desired tradeoff between error detection latency and complexity (i.e., additional software modifications to incorporate QED checks).

Our FT approach on dual-core ECUs can detect one permanent fault, and tolerate multiple soft-errors and intermittent faults. Multiple soft-errors and/or intermittent faults are tolerated as our approach recomputes the result on any soft-error and/or intermittent fault detection at any point in the program, and repeats this recomputation process till an error-free result is obtained. Our approach distinguishes permanent faults from transient faults by setting a *threshold* on the number of recomputations: if an error is not recovered by the recomputation threshold, the error is designated as a permanent fault, and then *acceptance tests* are utilized to select the result from the correctly functioning core. The recomputation threshold value is set based on the real-time constraints and the available *slack time*. The slack time is determined from the difference between the real-time deadline and the worst-case execution time. Section 4.2 elaborates further on meeting real-time deadlines in the presence of recomputations to recover from transient faults. We note that both permanent and intermittent faults can be eliminated by vehicle service and repair, however, our proposed approach provides resilience against these faults while driving on-road where instantaneous vehicle service and repair are not feasible. A major advantage of both the FT-RMT and the FT-RMT-QED approaches is that the methods are *self-checking*, i.e., they do not require a separate golden response created through simulation.

Our FT approach utilizing dual-core ECUs essentially detects errors whereas error correction is enabled by recomputations. To permit single error (permanent or transient) detection as well as correction on-the-fly without recomputations, we extend our FT approach to utilize triple-core ECUs that can be configured as either FT-RMT-TMR or FT-RMT-TMR-QED. The FT-RMT-TMR executes the computations on three redundant threads and the correct output is determined by *majority voting* of the threads' outputs. The FT-RMT-TMR-QED inserts check instructions at different points in the program/computation to enable early detection of errors via majority voting. The FT-RMT-TMR-QED copies the correct output, which is determined by majority voting, to an additional buffer. Then, the output of the faulty thread is replaced by the correct output so that the approach can provide single error detection and correction for the remaining computation as well. Hence, by using additional comparison and copy instructions, the FT-RMT-TMR-QED can detect and correct multiple single errors at different points in the program (the number of errors tolerated depends upon the granularity of check instructions) albeit at the expense of additional performance overhead.

3.2 Security Threat Model

In order to better elucidate our security approach, we characterize the likely capabilities of an adversary aiming to infiltrate an automobile's internal networks (e.g., CAN, FlexRay). Modern automobiles provide several physical interfaces, such as on-board diagnostics (OBD-II) port (typically located under the car's dashboard) and entertainment systems (e.g., CD, USB, iPod), that directly provide access to an automobile's internal networks. An adversary can also connect to an automobile's internal networks indirectly via short range wireless access (e.g., bluetooth, remote keyless entry, wireless tire pressure sensors) or long range wireless access (e.g., telematics systems, such as General Motor's OnStar, and broadcast channels, such as global positioning systems, satellite radio, digital radio) [13]. Hence, in order to ensure the safety, security, and privacy of vehicles and passengers, security primitives, in particular, confidentiality, integrity, and authentication, needs to be integrated in in-vehicle networks. Assuming that an adversary has gained access to an automobile's internal networks either directly or indirectly, this section summarizes briefly the associated security threat model against which our proposed approach provides resilience.

Threat 1—Passive Eavesdropping & Traffic Analysis → Need for Confidentiality: Modern automotive in-vehicle networks carry a mix of operational and personally identifiable information, such as current location, previous destinations, navigation history, call history, microphone recordings, and financial transactions, etc. An adversary invading an in-vehicle network could perform passive eavesdropping (i.e., sniff and store all the traffic in an automobile's internal network) and traffic analysis, thus, obtaining critical information about the driver and the vehicle. In addition, for the x-by-wire systems, if an adversary knows the initial location of the vehicle, then, by eavesdropping and traffic analysis of the steering angle, accelerator, and braking values, the adversary could track the car which might put the driver and passengers at risk. Even for encrypted messages, it is important to consider whether an adversary could obtain partial or complete information from the messages. Additionally, an adversary can have the capability to request generation of encrypted messages. Recorded packets and/or knowledge of the plain-text can be leveraged to reveal the encryption key, decrypt complete packets, or gather other useful information through traffic analysis. Hence, the confidentiality of messages and data over in-vehicle networks is critical for operational security, privacy, and consumer trust.

Threat 2—Active Eavesdropping & Message Injection → Need for Authentication and Integrity: An attacker may perform spoofing attacks by actively injecting and/or modifying messages in the in-vehicle network. For example, an adversary may attach his/her own device or compromise a valid user device (e.g., a cell phone attached to the infotainment system) in order to send fraudulent (or malicious) requests (commands, codes, or messages) into the system. Similarly, an adversary might inject malicious messages by encoding the messages on a CD as a song file and convincing the user to play the CD using social engineering. The adversary can also insert a replayed packet if there is no replay protection or the adversary can circumvent the replay protection. By inserting some well targeted messages, the adversary might be able to gain more information from the system reaction through active eavesdropping. Furthermore, the attacker's device may impersonate a valid ECU or gateway for malicious activities that may jeopardize safety of the driver and the vehicle. Thus, entity authentication and message integrity verification are required in in-vehicle networks to defend against these vulnerabilities.

Threats 1 and *2* are possible in the absence or possible breaking of data confidentiality and integrity in in-vehicle networks. These threats put the safety, security, and privacy of driver and passengers of the vehicle at serious risk. These threats expose the automobile to severe vulnerabilities as the adversary can potentially control many safety-critical systems (e.g., brakes, engine, lights, locks) while completely ignoring the driver input [5]. We reemphasize that the basic CAN protocol does not incorporate any security primitives to countermeasure the above mentioned threats. Some recent works in literature have proposed protocols for integrating security primitives over CAN, however, most of the proposed protocols either provide authentication and integrity or confidentiality but not confidentiality, integrity, and authentication simultaneously. Furthermore, the proposed works do not consider FT aspects nor the feasibility of simultaneously integration of security and dependability primitives in terms of adhering to real-time constraints of automotive distributed control functions.

3.3 Security

To countermeasure the threats presented in Section 3.2, our approach (Fig. 1) provides confidentiality, integrity, and authentication for automotive CPS with CAN as the vehicular network. To minimize the encryption overhead while providing adequate security for CAN message lifetimes, we leverage AES-128 (128-bit) encryption to provide confidentiality and an HMAC based on SHA-2/SHA-256 (Secure Hash Algorithm-2) to render integrity and authenticity [28]. Fig. 1 shows the operation of our approach at both the sending and receiving CAN nodes using generic dual-core/triple-core architectures. The figure shows that the sending CAN node consists of an SHA-2-based HMAC module and an AES encryption module. The SHA-2-based HMAC module implements SHA-256 algorithm that takes the message M as input and outputs 256 bits, which is known as the *message digest*. The 256-bit HMAC $HMAC_S(M)$ and the message M are given as input to the AES encryption module, which encrypts the message and $HMAC_S(M)$. Our approach assumes that initial AES and HMAC keys are stored in secure tamper resistant memories of participating ECUs by original equipment manufacturers (OEMs). Furthermore, to provide *key freshness*, the AES and HMAC keys are updated/refereshed deterministically over time by participating ECUs as is done in the transport layer security (TLS) [29]. Whenever car engine starts, one of the ECUs (designated by OEM) in each safety-critical functional domain broadcasts a nonce and AES and HMAC keys are updated deterministically based on that nonce for remainder of the encrypted messages on the CAN bus. This continuous refreshing of AES and HMAC keys prevents replay attacks. Our approach for providing confidentiality, integrity, and authentication is inspired by the secure sockets layer (SSL) [29]. In our approach, confidentiality, integrity, and authentication can be added to a message M as

$$\begin{aligned}\mathcal{O}_{M,S} &= \text{calc}[HMAC_S(M)]\\ &\quad + \text{ Encrypt}_{AES}[M + HMAC_S(M)],\end{aligned} \tag{1}$$

where `calc()` denotes calculation of $HMAC_S(M)$ of the message M by the sending node S, and $O_{M,S}$ denotes operations at the sending CAN node S that include encryption of the message M and the $HMAC_S(M)$ using AES algorithm.

To determine the storage bits required for the $\mathcal{O}_{M,S}$ operation, let us consider an 8-byte CAN message. Hence, $M + HMAC(M)$ = 64 + 256 bits = 320 bits. The encryption of these 320 bits require three 128-bit AES blocks. The first two AES blocks encrypt the first 256 bits whereas the third AES block encrypts the remaining 64 bits padded by a 1 bit followed by 0 bits to make the block length of 128 bits (padding in hexadecimal looks like 0x80,0x00,. . .,0x00) [28]. The transfer of 384 bits (3 AES blocks) of the encrypted message requires 384/64 = 6 CAN message frames. Hence, an unencrypted 8-byte CAN message requires six 8-byte CAN messages on encryption. We point out that public key cryptography can be used to exchange and update symmetric keys (both for AES and HMAC) at regular intervals for enhanced security, however, this key establishment process is not the focus of our current work.

Eq. (1) summarizes the operations at the sending CAN node, however, additional comparison and copy instructions are required to implement our FT approaches. For instance, the FT-RMT-QED requires comparison instructions to compare the threads' output for error detection. Similarly, the majority voting on threads' outputs, in case of FT-RMT-TMR-QED, requires additional comparison instructions as compared to the FT-RMT-QED. Moreover, error correction for FT-RMT-TMR-QED approach requires copying the correct output determined by majority voting to an additional buffer, and then copying the correct output to the faulty thread's output to reinstate single error detection and correction capability for the remaining computation. The operations at the sending CAN node for the FT-RMT-TMR-QED can be given as

$$\begin{aligned}\mathcal{O}_{M,S}^{FT-RMT-TMR-QED} &= \text{calc}^{\forall i=1,2,3}\Big[HMAC_S^{T_i}(M)\Big]\\ &+ \text{copy}\Big[HMAC_S^{c}(M), \mathcal{V}_{mj}^{i=1,2,3}\big\{HMAC_S^{T_i}(M)\big\}\Big]\\ &+ \text{copy}\Big[HMAC_S^{T_f}(M), HMAC_S^{c}(M)\Big]\\ &+ \text{Encrypt}_{\text{AES block}_j}^{\forall j=1,2,3}\Big[M + HMAC_S^{T_i\ \forall\ i=1,2,3}(M)\Big]\\ &+ \text{copy}^{\forall j=1,2,3}\Big[\text{AES block}_j^{c}, \mathcal{V}_{mj}^{i=1,2,3}\big\{\text{AES block}_j^{T_i}\big\}\Big]\\ &+ \text{copy}^{\forall j=1,2,3}\Big[\text{AES block}_j^{T_f}, \text{AES block}_j^{c}\Big],\end{aligned} \tag{2}$$

where $\mathcal{V}_{mj}$ denotes the majority voting operation, $HMAC_S^c(M)$ denotes the correct HMAC output determined by majority voting on the threads' outputs, $HMAC_S^{T_i}(M)$ denotes the HMAC calculated by the thread $T_i\ \forall\ i = 1, 2, 3$ at the sending CAN node, $HMAC_S^{T_f}(M)$ denotes the faulty HMAC output from the thread T_f ($f \in \{1, 2, 3\}$) determined by majority voting comparisons, AES block_j^c denotes the correct encryption of AES block j determined by majority voting, and AES $\text{block}_j^{T_f}$ denotes the faulty AES encryption of block j from thread T_f determined by majority voting. Similar equations can be written for the operations at the sending CAN node for FT-RMT, FT-RMT-QED, and FT-RMT-TMR.

Fig. 1 shows that the receiving CAN node consists of the following modules: AES decryption module, formatting module, SHA-2-based HMAC module, and a comparator module. The AES decryption module decrypts the received CAN frames. The formatting module operates on the decrypted CAN frames to retrieve the message M and $HMAC_S(M)$. The SHA-2-based HMAC module calculates the HMAC of the received message $HMAC_R(M)$. To verify the integrity and authenticity of the received message, the comparator module at the receiving CAN node compares $HMAC_S(M)$ with $HMAC_R(M)$. If $HMAC_S(M)$ is equal to $HMAC_R(M)$, then the received message is authentic otherwise the message has lost its integrity. The operations at the receiving CAN node can be summarized as

$$\begin{aligned}\mathcal{O}_{M,R} &= \text{Decrypt}_{AES}[M + HMAC_S(M)]\\ &\quad + \text{format}[M + HMAC_S(M)] + HMAC_R(M)\\ &\quad + \text{comp}[HMAC_S, HMAC_R],\end{aligned} \tag{3}$$

where $\mathcal{O}_{M,R}$ denotes the message decryption along with the associated operations performed at the receiving CAN node to verify the received message's integrity. The function `format()` denotes the formatting/extraction of message M and $HMAC_S(M)$ from the received CAN frames, and the function `comp()` denotes the comparison of $HMAC_S(M)$ with $HMAC_R(M)$.

The actual implementation of our FT approaches at the receiving CAN node requires additional comparison and copy instructions. For instance, the operations at the receiving CAN node for FT-RMT-TMR-QED can be given as

$$\begin{aligned}
&\mathcal{O}_{M,R}^{FT-RMT-TMR-QED}\\
&= \text{Decrypt}_{\text{AES block}_j}^{\forall j=1,2,3}\Big[M + HMAC_S^{T_i\ \forall\ i=1,2,3}(M)\Big]\\
&\quad + \text{copy}^{\forall j=1,2,3}\Big[\text{AES block}_j^c, \mathcal{V}_{mj}^{i=1,2,3}\{\text{AES block}_j^{T_i}\}\Big]\\
&\quad + \text{copy}^{\forall j=1,2,3}\Big[\text{AES block}_j^{T_f}, \text{AES block}_j^c\Big]\\
&\quad + \text{format}^{i=1,2,3}\Big[M + HMAC_S^{T_i}(M)\Big]\\
&\quad + \text{calc}^{\forall i=1,2,3}\Big[HMAC_R^{T_i}(M)\Big]\\
&\quad + \text{copy}\Big[HMAC_R^c(M), \mathcal{V}_{mj}^{i=1,2,3}\{HMAC_R^{T_i}(M)\}\Big]\\
&\quad + \text{comp}\big[HMAC_S(M), HMAC_R^c(M)\big].
\end{aligned} \tag{4}$$

Similar equations can be written for the operations at the receiving CAN node for FT-RMT, FT-RMT-QED, and FT-RMT-TMR.

Our security approach provides resilience against the security threat model described in Section 3.2. We clarify that our approach provides security for all the CAN frame fields (e.g., data field, control field) except for the channel arbitration field and the start and end of frame bits as the leakage of these arbitration fields is not considered harmful in most scenarios. It is reasonable to believe that once security primitives (confidentiality, integrity, and authentication) are implemented using our approach, the adversary cannot break the message confidentiality and integrity without knowing the secret key (please refer to Section 5.1 for cryptanalysis). Furthermore, an adversary cannot obtain useful information about the key via analyzing the ciphertext even if the corresponding plaintext is known. With reference to the *threat 1*, an adversary may eavesdrop on the vehicular network traffic but the adversary cannot decrypt the packets without knowing the secret key. Our approach completely eliminates the *threat 2* because SHA-2-based HMAC not only prevents insertion of forged messages but also prohibits message modifications. Refreshing of AES and HMAC keys over time by ECUs in our approach prevents replay attacks.

3.4 Advantages of the Proposed Approach

Other than the simultaneous integration of security and dependability primitives, our proposed approach offers additional benefits including *cost-effectiveness*, *flexibility*, *performance*, and *energy efficiency*. Since automotive domain has stringent cost constraints, our proposed multicore-based approach provides a cost-effective way to incorporate security and dependability primitives as a single multicore ECU can implement the functionality of multiple single-core ECUs. This potential reduction in the number of single-core ECUs and the associated wiring harnesses results in cost efficiency. Furthermore, the proposed multicore-based approach offers a cost-effective and flexible approach as compared to the dependability and security approaches based on specific hardware architectures, such as lock-step dual processor architecture, Secure Hardware Extension (SHE) [30]. In particular, since new vulnerabilities and corresponding countermeasures are continuously discovered, our multicore-based solution provides the flexibility of relatively easy upgrade to newer security and dependability protocols. Moreover, the contemporary dedicated solutions provide either security or dependability but not the two simultaneously. Furthermore, the additional computing power furnished by multicore ECUs makes them a suitable platform for implementing real-time constrained compute-intensive tasks of modern cars (including autonomous vehicles), such as audio, video, image processing, and machine learning algorithms (e.g., for autonomous driving). The multiple cores in multicore ECUs impart enhanced performance through parallelism, and thus permit running the cores at a lower clock frequency than single-core ECUs, which results in reduced energy consumption. Additionally, the potential reduction in the number of single-core ECUs and associated wiring harnesses, and conversion from mechanical/hydraulic systems to electronic ones reduce the vehicle weight, and thus fuel/energy consumption.

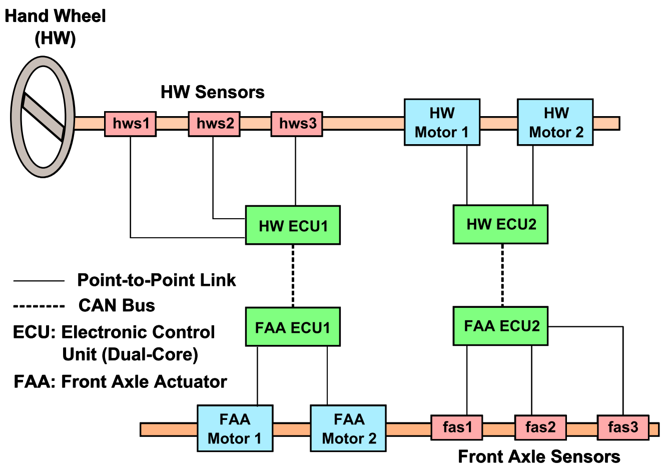

Fig. 2. Steer-by-wire operational architecture.

4 Steer-by-Wire System

An SBW system replaces mechanical steering system with ECUs, sensors, and actuators, which interact via a communication bus, such as CAN or FlexRay. An SBW system provides various advantages over mechanical steering systems that motivate the adoption of x-by-wire systems for future automobiles. An SBW system eliminates the risk of steering column entering the cockpit in the event of a frontal crash. Since steering column is one of the heaviest components in the vehicle, removing the steering column reduces the weight of the vehicle and therefore reduces fuel consumption. An SBW system enhances driver comfort by providing a variable steering ratio, that is, the steering ratio between the handwheel and the road wheels can be adapted according to the driving conditions (e.g., smaller steering ratio in parking and urban driving as compared to the driving on freeways). This section elaborates our SBW system that leverages multicore ECUs to incorporate dependability and security primitives.

4.1 Steer-by-Wire Operational Architecture

Fig. 2 depicts the SBW architecture that we consider for our case study. The architecture provides FT by redundancy at ECU-level, sensor-level, and actuator level. The architecture consists of two multicore hand wheel ECUs (HW ECU1 and HW ECU2) and two multicore front axle actuator ECUs (FAA ECU1 and FAA ECU2). The multicore ECUs (HW ECU1, HW ECU2, FAA ECU1, and FAA ECU2) for the SBW system can implement either of our FT approaches (FT-RMT, FT-RMT-QED, FT-RMT-TMR, FT-RMT-TMR-QED) depending upon the dependability and performance requirements. Each of the ECUs is connected to the CAN bus. Our SBW architecture consists of three hand wheel sensors (hws1, hws2, and hws3) that are placed near the hand wheel to measure the driver's requests in terms of hand wheel angle, hand wheel torque, and the hand wheel speed. Similarly, three front axle sensors (fas1, fas2, and fas3) measure the front axle position. The hand wheel sensors (the front axle sensors) are connected to the HW ECUs (FAA ECUs) by point-to-point links. Two front axle actuator (FAA) motors (FAA motor 1 and FAA motor 2) operate in active redundancy on the front axle while two hand wheel (HW) motors (HW motor 1 and HW motor 2) operate in active redundancy on the hand wheel.

A SBW system aims to provide two main services [2]: (1) Front axle control (FAC) that controls the wheel direction in accordance with the driver's request; and (2) Hand wheel force feedback (HWF) that provides a mechanical-like force feedback to the hand wheel. In our implementation of the SBW function, the FAC function is implemented by HW ECU1 and FAA ECU1, and the HWF function is implemented by FAA ECU2 and HW ECU2. In the following, we further elaborate FAC and HWF functions' implementations for the SBW application.

4.1.1 Front Axle Control Function

The FAC function computes the orders that are given to the front axle motor(s) according to the state of the front axle and the driver's requests obtained through the hand wheel. The three HW sensors (hws1, hws2, and hws3) measure the driver's requests and send these measurements to the HW ECU1. The HW ECU1 performs the necessary filtering and computations on the sensed input. After the necessary computations, HW ECU1 formats the computed orders/signals in CAN message format, calculates the message's HMAC to provide integrity, and performs encryption of the message and the associated HMAC to provide confidentiality. These computations are required to be completed under a few milliseconds due to stringent real-time constraints of the SBW system. The encrypted messages are sent by the HW ECU1 CAN controllers to the FAA ECU1 via the CAN bus. The FAA ECU1, which is placed behind the front axle, uses the data sent by the HW ECU1 as well as the last wheel position to determine the commands for the FAA motor 1 and 2.

4.1.2 Hand Wheel Force Feedback Function

The HWF function computes the orders that are given to the hand wheel motor(s) according to the speed of the vehicle, the front axle position, and the front tie rod force. The three front axle sensors (fas1, fas2, and fas3) send the sensed front axle position to the FAA ECU2. The FAA ECU2 performs the necessary computations, formats the computed signals/values in CAN message format, calculates HMAC, and performs encryption of the message and the associated HMAC. The FAA ECU2 CAN controller sends the encrypted messages on the CAN bus, which are then consumed by the HW ECU2 to compute the commands for HW motor 1 and 2 to provide the necessary force feedback to the HW.

4.2 QoS and Behavioral Reliability

A SBW system is sensitive to the delay from the driver's request at the hand wheel to the reception of the request by the front axle actuators. This end-to-end delay/response time T_{res} is perceived as the QoS, and can impact availability and safety in the worst case if exceeded beyond a *critical threshold* T_{max}. The *behavioral reliability* is defined as the probability that the worst-case response time is less than the critical threshold [6]. Automotive OEMs determine T_{max} by various means, such as Matlab/Simulink simulations, vehicular network simulations, vehicle system simulations, and vehicle tests, etc. The impact of T_{res} variation on vehicle's performance and stability can be evaluated in terms of a *QoS score*, denoted by S, determined by the time required to reach the desired position and stability. Table 1 shows the relation between S and the perturbation time (pure delay) for an instantaneous rotation of handwheel from 0 to 45 degree at 100 km/h (Table 1 is derived by extrapolating the data in [2]). We observe from the table that with a minimum tolerable score of 11.19, the critical limit T_{max} for the response time is 6 ms beyond which the vehicle becomes unstable and the safety of the driver can be at risk.

In the following, we analytically model the response time for the SBW functions. We also model the error resilience provided by our adopted FT approaches (Section 3) for the SBW system subject to the implicit timing constraints imposed by behavioral reliability.

The T_{res} for the FAC function (and the HWF function) comprises of the *pure delay* T_P, the *mechatronic delay* T_{mech}, and the *sensing delay* T_{sens}, i.e.,

$$T_{res} = T_P + T_{mech} + T_{sens}. \tag{5}$$

The pure delay comprises of the ECUs' computational delay for processing the driver's command, producing the command for the actuator, and the transmission delay including the bus arbitration. The pure delay for our secure and dependable SBW system also consists of processing delay for the incorporated security and dependability primitives. The mechatronic delay is the delay introduced by actuators such as electric motors. The sensing delay corresponds to the delay involved in sensing the driver's command (front axle position) and storing the sensed information in a memory location accessible by HW ECUs (FAA ECUs). Since T_{mech} and T_{sens} can be bounded by a constant and can be different for different kind of actuators and sensors, we focus on T_P for our analysis [2]. Systems that cannot guarantee a T_P lower than a tolerable upper bound T_{max} are considered unstable. The behavioral reliability P_{BR} evaluation is based on the worst-case T_P and not its nominal value because of the safety-critical nature of the system, that is, $P_{BR} = P[T_P^{WC} < T_{max}]$, where T_P^{WC} denotes the worst-case T_P.

The nominal T_P for the FAC function, T_P^{FAC}, is given as

$$T_P^{FAC} = T_{ecu-hw1} + T_{ecu-faa1} + T_{channel}^{hw1-faa1}, \tag{6}$$

TABLE 1
QoS Score S versus Pure Delay T_P for a Steering System [2]

Steering System Configuration	T_P (ms)	S
Mechanical steering system	0	11.23
Steer-by-wire	3.6	11.21
Steer-by-wire	5.0	11.20
Steer-by-wire	6.0	11.19
Steer-by-wire	8.0	11.17
Steer-by-wire	9.6	11.15
Steer-by-wire	10.0	11.146
Steer-by-wire	11.5	11.13

where $T_{ecu-hw1}$ and $T_{ecu-faa1}$ denote the computation time at HW ECU1 and FAA ECU1, respectively, and $T^{hw1-faa1}_{channel}$ denotes the channel time to transmit secure messages from HW ECU1 to FAA ECU1.

The worst-case T_P for the FAC function, T^{FAC}_{P-WC}, is given as

$$\begin{aligned} T^{FAC}_{P-WC} &= n_1 \cdot T_{ecu-hw1} + n_2 \cdot T_{ecu-faa1} + n_3 \cdot T^{hw1-faa1}_{channel}, \\ & n_1, n_2 \in \{1, 2, \ldots\},\ n_3 \in \{1, 1.167, 1.33, 1.5, \ldots\}, \end{aligned} \tag{7}$$

where n_1 and n_2 denote the number of computations required to yield an error-free result at HW ECU1 and FAA ECU1, respectively, and n_3 denotes the number of transmissions required for error-free sending of secure messages over CAN. In Eq. (7), n_3 values follow a fractional variation as six encrypted CAN frames are required for one unencrypted message in our secure and FT SBW system, and errors can occur in transmission of any or all of the frames.

For ensuring the QoS dictated by behavioral reliability, the T_{P-WC} must be less than or equal to T_{max}. Hence, for the FAC function

$$n_1 \cdot T_{ecu-hw1} + n_2 \cdot T_{ecu-faa1} + n_3 \cdot T^{hw1-faa1}_{channel} \leq T_{max}. \tag{8}$$

Eq. (8) helps in analyzing the number of computational errors (i.e., errors occurring in the computation due to permanent, transient and/or intermittent faults) and transmission errors tolerated by a secure and dependable SBW system to attain the desired QoS and behavioral reliability corresponding to T_{max}. Eq. (8) determines exactly the maximum number of tolerable errors for one component (HW ECU1, FAA ECU1, or channel) given the parameters for the other two components are fixed for FT-RMT. Eq. (8) indicates that early detection of errors in the program, which could be caused by permanent, transient and/or intermittent faults, can provide room for more computations to yield the error-free result within the time constraint dictated by the required QoS. We reemphasize that the dependability focus of this paper is on the detection and correction of permanent and transient, and intermittent faults. The program errors resulting from other sources (e.g., specification faults, control flow, and timing errors, etc.) need to be vetted through various hardware and software verification and assurance techniques during the design and test phase. For FT-RMT-QED, Eq. (8) gives an estimate for the maximum number of tolerable errors. The exact number of maximum tolerable errors for FT-RMT-QED is determined by inserting the computation time value for the error detected at a particular point in the program. For example, using FT-RMT-QED and keeping n_2 and n_3 fixed, actual $n_1 \cdot T_{ecu-hw1}$ is determined as

$$n_1 \cdot T_{ecu-hw1} = \begin{cases} T^{C}_{ecu-hw1}, & n_1 = 1 \\ (n_1 - 1) \cdot T^{QED}_{ecu-hw1} + T^{C}_{ecu-hw1}, & n_1 \geq 2, \end{cases} \tag{9}$$

where $T^{C}_{ecu-hw1}$ denotes the time required for the complete computation at HW ECU1 whereas $T^{QED}_{ecu-hw1}$ denotes the error detection time for an erroneous computation at HW ECU1. Eq. (9) assumes that all errors are detected at the same point in the program using QED, however, if errors are detected at different points, error detection times for the errors detected at different points in the program are to be added accordingly. Since FT-RMT-TMR-QED can also correct single errors in the program, Eq. (9) is applicable for FT-RMT-TMR-QED only for the case when there are multiple (≥ 2) errors in the program between the inserted check points so that the majority voting is unable to reach at a consensus output.

The nominal and worst-case pure delay, and the number of maximum tolerable errors for the HWF function can be derived similarly.

5 Evaluation Results

Our secure and FT automotive CPS design provides confidentiality, integrity, and authenticity for CAN messages by AES-128 encryption/decryption and SHA-2-based HMAC. We implement these security primitives on an Intel core 2 quad processor Q9450, with symmetric multiprocessor architecture, operating at 2.66 GHz. We measure the clock cycles required for execution when dependability is furnished by dual-cores (FT-RMT and FT-RMT-QED) and triple-cores (FT-RMT-TMR and FT-RMT-TMR-QED). The Intel core 2 quad processor runs `GNU/Linux 2.6.18-308.24.1.el5PAE` version #1 `SMP`. We point out that 32-bit symmetric multiprocessor architecture is prevalent in embedded applications, and hence we choose this architecture for obtaining clock cycles. Using the clock cycles, we estimate the security primitives execution time on a 32-bit multicore ECU (dual-core/triple-core) for safety-critical automotive CPS applications. We adopt OpenMP [31] to provide RMT-based FT on a multicore architecture. For our case study, we assume the steering wheel sensor sampling rate to be fixed at 420 Hz, that is, T_{sens} = 2.38 ms [32]. We simulate our SBW system in Vector CANoe [33] with CAN baud rate set to 1 Mbps. We use CAPL (Vector CANoe programming language) to implement the SBW functions on ECUs. This section first presents the theoretical cryptanalysis of our proposed security approach. Then this section discusses timing analysis for implementing security and dependability primitives on a 32-bit multicore ECU. We then quantify the number of computational faults tolerated by ECUs for the SBW system with given QoS and behavioral reliability constraints. Finally, the section presents scalability analysis for the SBW system as the CAN bus load varies.

5.1 Cryptanalysis

Our proposed security approach (Section 3.3) provides resilience against the security threat model described in Section 3.2. The security of our proposed scheme relies upon the security of AES encryption and SHA-2 hash

TABLE 2
$\mathcal{O}_{M,S}$ Timing Results for a 32-Bit ECU Operating at 200 MHz

Operational Mode	Error Detection Point	ECU Architecture	Clock Cycles	Time (μs)
Without FT	N/A	single-core	163,218	816.09
FT-RMT	@ end of computation	dual-core	222,820	1,114.1
FT-RMT-QED	@ end of computation	dual-core	230,776	1,153.9
FT-RMT-QED	@ HMAC calculation	dual-core	158,015	790.1
FT-RMT-QED	@ AES expand key operation	dual-core	173,843	869.22
FT-RMT-QED	@ AES encryption (block 1)	dual-core	197,524	987.6
FT-RMT-QED	@ AES encryption (block 2)	dual-core	215,540	1,077.7
FT-RMT-TMR	@ end of computation	triple-core	271,437	1,357.18
FT-RMT-TMR-QED	@ any	triple-core	372,826	1,864.13

FT-RMT and FT-RMT-QED denote fault-tolerance by RMT, and fault-tolerance by RMT with QED, respectively.

function. There is currently no known analytical attack against AES and a brute-force attack leveraging a supercomputer (10.51 petaFLOPS) would require 1 billion billion (10^{18}) years to crack the 128-bit AES key [34]. There are also currently no known collisions or attacks against full round SHA-2 [35]. This cryptanalysis discussion assumes that side-channel attacks can be thwarted. Cryptanalysis using side-channel attacks is beyond the scope of this paper.

5.2 Timing Analysis

For timing analysis of our FT approaches, we inject soft errors at different points in the program (security primitives implementation). Our software-based fault injection emulates bit flipping in the program/memory due to external noise and/or radiation. We note that for fault tolerance timing analysis of our approach, fault locations can assume any distribution (e.g., uniform, Gaussian), can be time or space triggered, and can be injected manually or through the use of dedicated fault injection tools. For precise control of error locations at different phases of the program, we have manually inserted/triggered errors in the program by code manipulation.

Table 2 presents the timing results with FT operational modes (FT-RMT, FT-RMT-QED, FT-RMT-TMR, and FT-RMT-TMR-QED) for a 32-bit ECU operating at 200 MHz for the $\mathcal{O}_{M,S}$ computations at the sending CAN node given by Eq. (1). We measure the clock cycles (averaged over 10 runs to smooth any discrepancies due to operating system overheads) using `rdtsc()` [36] at the start and end of computations. Table 2 indicates that incorporating FT (in any configuration: FT-RMT, FT-RMT-QED, FT-RMT-TMR, or FT-RMT-TMR-QED) incurs performance overhead as compared to the single-core implementation with no FT. For example, FT-RMT, FT-RMT-QED, FT-RMT-TMR, and FT-RMT-TMR-QED incur performance overheads of 36, 41, 66, and 128 percent, respectively, at the sending CAN node. The FT techniques incur performance penalty due to inherent multi-threading overhead, and additional instructions for comparison and copy operations. Results verify that the FT-RMT-QED (FT-RMT-TMR-QED) enables earlier detection (and/or correction) of errors for $O_{M,S}$ computations as compared to the FT-RMT (FT-RMT-TMR) depending on the error point in the program. For example, the FT-RMT-QED detects an error 41 percent clock cycles earlier as compared to the FT-RMT when the error occurs at HMAC calculation.

Table 3 presents the timing results for a 32-bit ECU operating at 200 MHz for the $\mathcal{O}_{M,R}$ computations at the receiving CAN node given by Eq. (3). The FT approaches at the receiving CAN node also incur performance overhead as compared to the single-core implementation with no FT. Results reveal that FT-RMT, FT-RMT-QED, FT-RMT-TMR, and FT-RMT-TMR-QED incur performance overheads of 36, 41, 75, and 130 percent, respectively, at the receiving CAN node. Results verify that FT-RMT-QED enables early detection of errors as compared to FT-RMT for the $\mathcal{O}_{M,R}$ computations. For example, FT-RMT-QED detects an error 158 percent clock cycles earlier than FT-RMT when the error occurs at the AES expand key operation.

Results in Tables 2 and 3 indicate that FT-RMT-TMR-QED incurs more overhead than FT-RMT-QED because of an additional redundant thread as well as additional comparison and copy instructions. The comparison of the three threads outputs for majority voting in case of FT-RMT-TMR-QED

TABLE 3
$\mathcal{O}_{M,R}$ Timing Results for a 32-Bit ECU Operating at 200 MHz

Operational Mode	Error Detection Point	ECU Architecture	Clock Cycles	Time (μs)
Without FT	N/A	single-core	169,761	848.8
FT-RMT	@ end of computation	dual-core	230,044	1,150.22
FT-RMT-QED	@ end of computation	dual-core	238,796	1,193.98
FT-RMT-QED	@ AES expand key operation	dual-core	89,214	446.07
FT-RMT-QED	@ AES decryption (block 1)	dual-core	111,238	556.19
FT-RMT-QED	@ AES decryption (block 2)	dual-core	134,101	670.5
FT-RMT-QED	@ AES decryption (block 3)	dual-core	164,482	822.41
FT-RMT-QED	@ formatting received HMAC	dual-core	182,466	912.33
FT-RMT-QED	@ HMAC calculation	dual-core	230,564	1,152.82
FT-RMT-TMR	@ end of computation	triple-core	296,339	1,481.7
FT-RMT-TMR-QED	@ any	triple-core	391,099	1,955.5

FT-RMT and FT-RMT-QED denote fault-tolerance by RMT, and fault-tolerance by RMT with QED, respectively.

TABLE 4
Performance Advantage ($\mathcal{I}_{\%}$) of FT-RMT-TMR-QED (Denoted as $\mathbf{FT}_{\mathbf{TMR}}^{\mathbf{QED}}$) over FT-RMT-QED for QEDC

Error @	FT-RMT-QED (ms)	$\mathbf{FT}_{\mathbf{TMR}}^{\mathbf{QED}}$ (ms)	$\mathcal{I}_{\%}$
e_1	0.79 + 1.15 = 1.94	1.86	4.3
e_2	0.869 + 1.15 = 2.02	1.86	8.6
e_3	0.988 + 1.15 = 2.138	1.86	14.9
e_4	1.08 + 1.15 = 2.23	1.86	19.9
$e_1 + e_2 + e_3 + e_4$	0.79 + 0.869 + 0.988 + 1.08 + 1.15 = 4.88	1.86	162

requires more instructions than the comparison of the two threads outputs for FT-RMT-QED. Furthermore, FT-RMT-TMR-QED requires additional copy instructions to copy the correct output determined by majority voting to an additional buffer as well as to copy the determined correct output to the faulty thread output so that the future computations can proceed from an error-free state, which permits error detection and correction during the course of computation.

Although FT-RMT-TMR-QED incurs additional overhead as compared to FT-RMT-QED for an error-free computation, FT-RMT-TMR-QED can provide performance improvement over FT-RMT-QED in case errors are detected in the computation. This better performance of FT-RMT-TMR-QED over FT-RMT-QED in the presence of errors can be vital for time-constrained safety-critical applications. Our experiments indicate that FT-RMT-TMR-QED is amenable for QEDC in error-prone environments (e.g., high electromagnetic interference (EMI), and radiations) because of on-the-fly error correction capability furnished by majority voting. The FT-RMT-QED, on the contrary, requires recomputation from start of the program in case of error detection, which can take longer to obtain error-free result as compared to FT-RMT-TMR-QED. The time required to obtain an error-free result from FT-RMT-QED includes the time to detect error(s) plus the time for a complete error-free computation. Table 4 shows the performance advantage of FT-RMT-TMR-QED for QEDC at the sending CAN node in an error-prone environment with errors occurring at different points (computational phases) in the program. In Table 4, e_x denotes error detection points in the program: $e_1 \rightarrow$ HMAC calculation; $e_2 \rightarrow$ AES expand key operation; $e_3 \rightarrow$ AES block 1 encryption; and $e_4 \rightarrow$ AES block 2 encryption. The last row in Table 4 represents the case where errors occur at each major computational phase in the program. Results show that FT-RMT-TMR-QED can provide 162 percent performance improvement over FT-RMT-QED in the presence of multiple soft errors.

5.3 QoS and Behavioral Reliability

An SBW system is sensitive to the delay from the driver's request at the hand wheel to the corresponding response from the front axle actuators. This delay is perceived as the QoS and can impact availability and safety if exceeded beyond a certain critical threshold T_{max}. The pure delay for a stable SBW system must be less than the critical delay T_{max} despite recomputations (permitted by FT approaches such as FT-RMT and FT-RMT-QED on error detection) and retransmissions to mask off computation and transmission errors, respectively.

TABLE 5
The Maximum Number of Allowed Computational Runs at HW ECU1 n_1 to Yield Correct Result for the FAC Function with T_{max} = 6 ms

n_2 & n_3	n_1 FT_a	n_1 FT_b	n_1 FT_c	n_1 FT_d	n_1 FT_e	n_1 FT_f
$n_2 = 1, n_3 = 1$	3	3	4	4	3	3
$n_2 = 1, n_3 = 1.167$	3	3	4	4	3	3
$n_2 = 1, n_3 = 1.33$	3	3	4	4	3	3
$n_2 = 1, n_3 = 1.5$	3	3	4	3	3	3
$n_2 = 1, n_3 = 1.667$	3	3	4	3	3	3
$n_2 = 1, n_3 = 1.833$	3	3	3	3	3	3
$n_2 = 1, n_3 = 2$	3	2	3	3	3	3

We conduct experiments to determine the maximum number of allowed recomputations at SBW ECUs to yield error-free results subject to a critical delay T_{max} of 6 ms. For brevity, we present results for FT-RMT and FT-RMT-QED only, however, FT-RMT-TMR and FT-RMT-TMR-QED depicts similar trends. The number of faults tolerated at HW ECU1 is given by $(n_1 - 1)$ since n_1 is the total number of computations required to obtain an error-free result. Table 5 depicts the maximum number of allowed recomputations at HW ECU1 to yield correct result for the FAC function when T_{max} = 6 ms and $T_{channel}^{hw1-faa1}$ = 0.737 ms. n_2 denotes the number of computational runs at FAA ECU1 and n_3 denotes the number of transmissions required for error-free transmission of the encrypted message. FT_a denotes FT-RMT, FT_b denotes FT-RMT-QED with error detected at the end of computation, FT_c denotes FT-RMT-QED with error detected at HMAC calculation, FT_d denotes FT-RMT-QED with error detected at AES expand key operation, FT_e denotes FT-RMT-QED with error detected at AES encryption of block 1, and FT_f denotes FT-RMT-QED with error detected at AES encryption of block 2.

Results (Table 5) indicate that FT-RMT-QED permits more (or at least equal) recomputations than FT-RMT, depending on the error detection point at HW ECU1, to yield correct result within the time constraints imposed by the desired QoS. For example, when $n_2 = 1$, $n_3 = 1$, and T_{max} = 6, FT-RMT-QED tolerates 50 percent more faults $(n_1 - 1)$ than FT-RMT (3 allowed recomputations for FT-RMT-QED as compared to 2 for FT-RMT) when FT-RMT-QED detects error at HMAC calculation or expand key operation. Results show that in the presence of transmission errors $(n_3 > 1)$, FT-RMT-QED can permit slightly less computations than FT-RMT when error is detected at the end of computation due to the overhead of QED checks insertions.

Table 6 depicts the maximum number of allowed recomputations at FAA ECU1 to yield correct result for the FAC function when T_{max} = 6 ms and $T_{channel}^{hw1-faa1}$ = 0.737 ms. The number of faults tolerated at FAA ECU1 is given by $(n_2 - 1)$. In the table, n_1 denotes the number of computational runs at HW ECU1 and n_3 denotes the number of transmissions required for error-free transmission of the encrypted message. FT_a denotes FT-RMT, FT_b denotes FT-RMT-QED with error detected at the end of computation, FT_g denotes FT-RMT-QED with error detected at AES expand key operation, FT_h denotes FT-RMT-QED with error detected at AES decryption of block 1, FT_i denotes FT-RMT-QED with error detected at AES decryption of block 2,

TABLE 6
The Maximum Number of Allowed Computational Runs at FAA ECU1 n_2 to Yield Correct Result for the FAC Function with T_{max} = 6 ms

n_1 & n_3	n_2 FT_a	n_2 FT_b	n_2 FT_g	n_2 FT_h	n_2 FT_i	n_2 FT_j	n_2 FT_k	n_2 FT_l
$n_1 = 1, n_3 = 1$	3	3	7	6	5	4	4	3
$n_1 = 1, n_3 = 1.167$	3	3	7	6	5	4	4	3
$n_1 = 1, n_3 = 1.33$	3	3	7	5	5	4	3	3
$n_1 = 1, n_3 = 1.5$	3	3	6	5	4	4	3	3
$n_1 = 1, n_3 = 1.667$	3	3	6	5	4	3	3	3
$n_1 = 1, n_3 = 1.833$	3	2	6	5	4	3	3	3
$n_1 = 1, n_3 = 2$	2	2	5	4	4	3	3	3

FT_j denotes FT-RMT-QED with error detected at AES decryption of block 3, FT_k denotes FT-RMT-QED with error detected at formatting received HMAC, and FT_l denotes FT-RMT-QED with error detected at HMAC calculation at the receiving node.

Table 6 indicates that FT-RMT-QED permits more (or at least equal) recomputations than FT-RMT, depending on the error detection point at FAA ECU1, to yield correct result within the time constraints imposed by the desired QoS. For example, when $n_1 = 1$, $n_3 = 1$, and $T_{max} = 6$ ms, FT-RMT-QED tolerates 200 percent more faults $(n_2 - 1)$ than FT-RMT (6 allowed recomputations for the FT-RMT-QED as compared to 2 for FT-RMT) when FT-RMT-QED detects error at AES expand key operation.

5.4 Scalability Analysis

Scalability analysis assesses system's performance for a projected addition of new components/messages later in the design process. Our scalability analysis assists automotive CPS early design phases where tactical decisions, such as message priority assignments, are to be made in the presence of incomplete and estimated information such as bus load. To investigate the scalability of our secure and dependable SBW system over CAN, we measure T_P and T_{res} both for the FAC function and the HWF function (T_P^{FAC} and T_P^{HWF} denote the pure delay for the FAC function and the HWF function, respectively). First, we analyze the system feasibility without any additional messages on the CAN bus. After feasibility analysis, we study the effect of additional messages on the CAN bus. The study of the additional CAN messages' effect is important as a CAN bus can need to carry messages for multiple distributed control functions. Furthermore, since the high computing power imparted by multicore ECUs can enable implementation of multiple control functions on a single multicore ECU (Section 3.4), additional messages will need to be transmitted on the CAN bus making the scalability analysis imperative. For comprehensive scalability study of the SBW system, we add messages both with higher priority as well as lower priority than the SBW application's messages on the CAN bus (in the rest of this paper, we denote these messages as high-priority and low-priority for brevity). We investigate the jitter induced in the pure delay of the SBW system due to additional load on the CAN bus. We also analyze the effect of higher priority messages' payload sizes on the SBW system's pure delay.

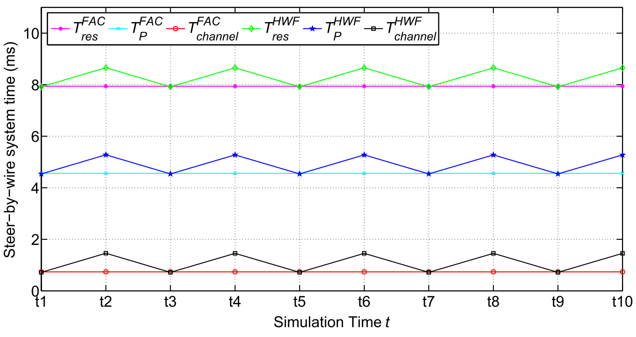

Fig. 3. Response time, pure delay, and channel time for the SBW system with FT-RMT-TMR-QED.

5.4.1 *Feasibility Analysis*

To investigate the feasibility of CAN for safety-critical real-time constrained cybercar applications, we simulate our SBW system without any additional messages on the CAN bus carrying the SBW application's messages. We obtain results for the SBW system with FT-RMT-QED as well as FT-RMT-TMR-QED for computations at SBW ECUs. For SBW system with FT-RMT-QED, we take $T_{ecu-hw1}$ and $T_{ecu-faa2}$ to be 1.15 ms (Table 2), and $T_{ecu-faa1}$ and $T_{ecu-hw2}$ to be 1.19 ms (Table 3). For SBW system with FT-RMT-TMR-QED, we take $T_{ecu-hw1}$ and $T_{ecu-faa2}$ to be 1.86 ms (Table 2), and $T_{ecu-faa1}$ and $T_{ecu-hw2}$ to be 1.96 ms (Table 3). To clarify, we point out that these time values are selected based on the computation time of sending and receiving ECUs where the FAC function is implemented by HW ECU1 and FAA ECU1 and the HWF function is implemented by FAA ECU2 and HW ECU2 (Section 4.1).

Results for the SBW system with FT-RMT-QED reveal that the response time and pure delay for the HWF function displays moderate jitters caused by the small channel time fluctuations. However, these pure delay jitters are still well within T_{max}. For example, the pure delay jitter for the HWF function is 24 percent on average over the SBW application run. The variation in channel time for the HWF function as compared to no variations for the FAC function is due to the lower priority of the HWF function's messages than the FAC function's messages.

Fig. 3 depicts response times, pure delays, and channel times both for the FAC function and the HWF function for the SBW system with FT-RMT-TMR-QED. Time on x-axis represents simulation times at different instants over the run of the SBW application to capture the overall timing behavior of the SBW system. Comparison with the corresponding results for FT-RMT-QED reveals that the response time and pure delay both for the FAC function and the HWF function for the SBW system with FT-RMT-TMR-QED is slightly greater than the corresponding response time and pure delay for the SBW system with FT-RMT-QED due to additional computational time required for FT-RMT-TMR-QED in case of no errors. The channel times, for both the FAC function and the HWF function, for the SBW system with FT-RMT-TMR-QED are equal to the corresponding times for the SBW system with FT-RMT-QED. Results in Fig. 3 assume an error-free run of the computations. We point out that the timing results for the SBW system with FT-RMT-QED would be shifted upward (implying additional required time) for computations with errors at

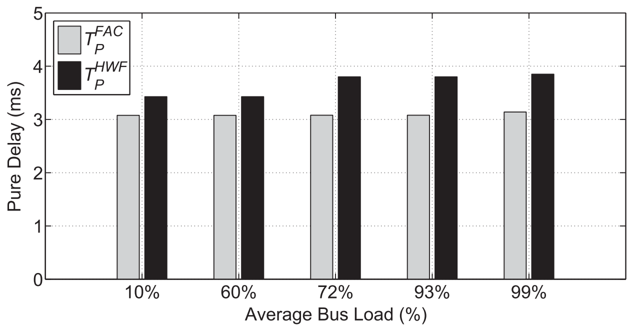

Fig. 4. Effect of increasing bus load due to low-priority messages on pure delay of the SBW application.

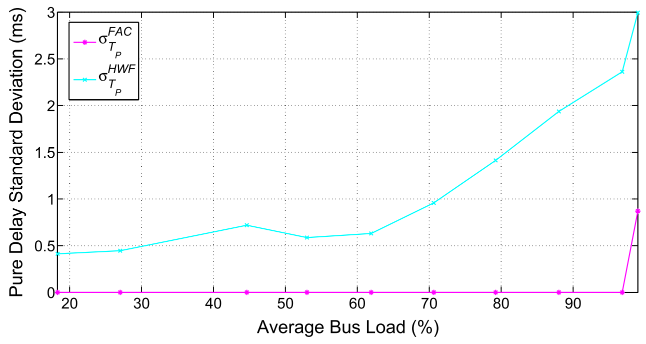

Fig. 6. Effect of increasing bus load due to high-priority messages on the pure delay's standard deviation σ of the SBW application.

multiple points (locations) in the program whereas the SBW system with FT-RMT-TMR-QED would be able to maintain the timing behavior in the presence of single transient errors occurring at multiple points in the program because of on-the-fly error detection and correction capabilities of FT-RMT-TMR-QED (Table 4).

Results verify that the pure delay for both the FAC and the HWF functions are much less than the typical critical delay for a SBW system (of the order of few milliseconds to tens of milliseconds). These results establish the feasibility of our secure and dependable SBW system implementation over the CAN bus.

5.4.2 Effect of Additional Messages & Bus Load

To investigate the scalability of CAN for safety-critical real-time constrained cybercar applications, we simulate our SBW system with both low-priority and high-priority additional messages on the CAN bus carrying the SBW application's messages. We studied the effect of low-priority and high-priority message sizes on the pure delay and delay jitter of the SBW function. We further examined the effect of high-priority message sizes on the pure delay of the SBW function.

Effect of Additional Low-Priority Messages on Pure Delay: Our experiments with high-priority and low-priority messages on the CAN bus reveal that the additional lower priority messages than the SBW function's (e.g., FAC) messages have negligible effect on pure delay of the SBW function. Fig. 4 depicts the effect of increasing bus load due to low-priority messages on the average pure delay for the SBW application. For our experiments, the SBW application messages (including both FAC and HWF functions) with no additional messages on the CAN bus correspond to the average bus load of 10 percent. Results show that the increasing bus load due to low-priority messages has negligible impact on the pure delay for both the FAC and the HWF function. For example, T_P^{FAC} and T_P^{HWF} increases by 2 and 12 percent, respectively, as the additional low-priority messages increases the average bus load from 10 to 99 percent.

Effect of Additional Low-Priority Messages on Delay Jitter: Experimental results reveal that high bus load due to low-priority messages does not induce large jitter in pure delay for both the FAC and the HWF function. For example, the standard deviation (σ) for T_P^{FAC} and T_P^{HWF} is 0.035 and 0.34 ms, respectively, for an average bus load of 99 percent due to additional low-priority messages. The comparatively high σ for T_P^{HWF} than T_P^{FAC} is due to lower priority of HWF function's messages than the FAC function's messages in our SBW application design.

Effect of Additional High-Priority Messages on Pure Delay: Experiments reveal that additional high-priority messages can drastically impact pure delay of the SBW application. Fig. 5 depicts the effect of increasing bus load due to high-priority messages on the average pure delay for the SBW application. Results show that both the average T_P^{FAC} and T_P^{HWF} increase as the bus load grows. Results indicate that T_P^{FAC} and T_P^{HWF} increases by 58.4 and 78 percent, respectively, as the average bus load due to high-priority messages increases from 45 to 88 percent. The figure shows that this delay increase is not linear and shoots up drastically above the average bus load of 97 percent.

Effect of Additional High-Priority Messages on Delay Jitter: Experiments indicate that high-priority messages can induce large jitters in pure delay for both the FAC and the HWF function especially at high bus loads. Fig. 6 depicts the effect of increasing bus load due to high-priority messages on the average pure delay's standard deviation σ for the SBW application. $\sigma_{T_P}^{FAC}$ and $\sigma_{T_P}^{HWF}$ denote σ for FAC and HWF functions, respectively. Results show that the $\sigma_{T_P}^{HWF}$ increases piecewise linearly with the increasing average bus load. $\sigma_{T_P}^{FAC}$ remains zero implying no variations in pure delay for the FAC function up to an average bus load of 97 percent beyond which the $\sigma_{T_P}^{FAC}$ increases. We point out that although $\sigma_{T_P}^{FAC}$ is zero for average bus loads less than 97 percent, pure delay for the FAC function increases linearly as the bus load increases. The comparatively high σ for T_P^{HWF} than T_P^{FAC} is due to lower priority of HWF function's messages than the FAC function's messages in our SBW application design. This jitter in pure delay due to additional high-priority messages can render the SBW system unstable. Further experiments with high-priority

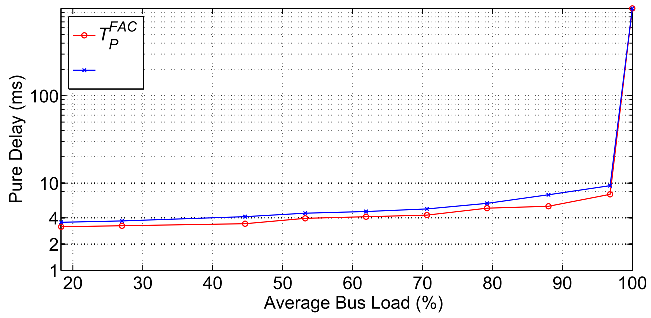

Fig. 5. Effect of increasing bus load due to high-priority messages on pure delay of the SBW application.

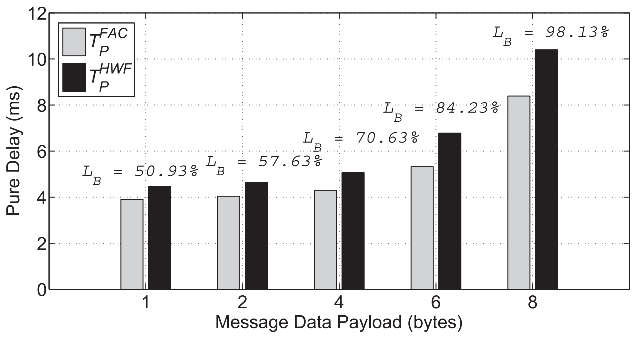

Fig. 7. Effect of increasing high-priority messages' payload on pure delay of the SBW application. L_B denotes the average bus load.

messages reveal that the SBW system can become completely unavailable if the CAN bus load due to high-priority messages reaches 99 percent.

Effect of Additional High-Priority Message Sizes on Pure Delay: To study the impact of high-priority message sizes on pure delay, we conduct simulations with different message payloads. Our simulation setup consists of 7 additional high-priority messages that are sent every 1 ms while the messages' payload varies. Fig. 7 depicts the effect of additional high-priority messages' payload on pure delay for both the FAC and the HWF functions. The figure also shows the resulting average bus load as payload for additional high-priority CAN messages varies. Results reveal that the average pure delay increases for both the FAC function and the HWF function as the payload increases. For example, the average T_P^{FAC} and the average T_P^{HWF} increases by 115 and 133 percent, respectively, as the size of additional high-priority messages' payload increases from 1 byte to 8 bytes. The average T_P^{HWF} increases more than the average T_P^{FAC} because our SBW system design assigns higher priority to the FAC function's messages than the HWF function's messages.

Scalability analysis suggests that safety-critical and time-constrained automotive CPS that incorporate security and dependability primitives can be implemented over CAN with careful selection of security and FT approaches as well as prudent priority assignment of CPS application's messages over CAN. Furthermore, for a stable automotive CPS over CAN, the bus load due to additional higher priority messages needs to be monitored and controlled. One way of message monitoring and control is to implement firewall and authentication services in gateway nodes that connect different automotive subnetworks.

6 Conclusions

In this paper, we provide an integrated approach for the design of secure and dependable automotive CPS with SBW application over controller area network as a case study. The challenge is to embed both security primitives (confidentiality, integrity, and authentication) and dependability primitives over CAN while ensuring that the real-time constraints of the automotive CPS applications are not violated. Our design leverages multicore (dual-core/triple-core) electronic control units to provide fault tolerance to the system by redundant multi-threading (FT-RMT) and FT-RMT with quick error detection (FT-RMT-QED).

Results reveal that FT-RMT-QED can detect errors 158 percent clock cycles earlier as compared to FT-RMT. We quantify the number of computational errors permitted by FT-RMT and FT-RMT-QED within the SBW system's worst-case response time threshold imposed by the desired QoS and behavioral reliability. Results show that FT-RMT-QED can tolerate 200 percent more faults than FT-RMT within the time constraints imposed by the desired QoS. To permit single error detection as well as correction on-the-fly without recomputations, we extend our FT approach to utilize triple-core ECUs that can be configured as either FT-RMT-TMR (an extension of FT-RMT for TMR) or FT-RMT-TMR-QED (an extension of FT-RMT-QED for TMR). Results reveal that FT-RMT-TMR-QED can provide 162 percent performance improvement over FT-RMT-QED in presence of multiple transient errors. Results verify the feasibility of our secure and dependable SBW system implementation over CAN. Scalability analysis reveals that the SBW system is able to maintain the desired QoS even in the presence of additional low-priority messages on the bus. Scalability analysis suggests that for a stable SBW system over CAN, the bus load due to additional high-priority messages must be restricted to less than 97 percent.

Our proposed approach can be adopted for early investigations of security, dependability, and performance interplay for other in-vehicle network protocols, such as CAN with flexible data-rate (CAD FD) or FlexRay. Our future work plans to investigate cost, performance, FT, and energy tradeoffs of hardware acceleration of security and dependability primitives for automotive CPS.

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) (NSF CNS Grant #1743490 and NSF CNS Grant #1649423) and the Office of Naval Research (ONR) (Award #N00014-17-1-2500). Any opinions, 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 and the ONR.

References

[1] ISO26262, "Road vehicles—Functional safety," Jan. 2013. [Online]. Available: http://www.iso.org/iso/catalogue_detail?csnumber=43464

[2] C. Wilwert, N. Navet, Y.-Q. Song, and F. Simonot-Lion, "Design of automotive X-by-wire systems," in *The Industrial Communication Technology Handbook*. Boca Raton, FL, USA: CRC Press, 2005.

[3] I. Koren and C. M. Krishna, *Fault-Tolerant Systems*. San Mateo, CA, USA: Morgan Kaufmann, 2007.

[4] A. Vinel, N. Lyamin, and P. Isachenkov, "Modeling of V2V communications for C-ITS safety applications: A CPS perspective," *IEEE Commun. Lett.*, May 2018, doi: 10.1109/LCOMM.2018.2835484.

[5] K. Koscher, A. Czeskis, F. Roesner, S. Patel, and T. Kohno, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 447–462.

[6] C. Wilwert, Y.-Q. Song, F. Simonot-Lion, Loria-Trio, and T. Clément, "Evaluating quality of service and behavioral reliability of steer-by-wire systems," in *Proc. IEEE Conf. Emerging Technol. Factory Autom.*, Sep. 2003, pp. 193–200.

[7] M. L. Chávez, C. H. Rosete, and F. R. Henríquez, "Achieving confidentiality security service for CAN," in *Proc. IEEE Int. Conf. Electron. Commun. Comput.*, Mar. 2005, pp. 166–170.

[8] T. Hong, Y. Li, S.-B. Park, D. Mui, D. Lin, Z. A. Kaleq, N. Hakim, H. Naeimi, D. S. Gardner, and S. Mitra, "QED: Quick error detection tests for effective post-silicon validation," in *Proc. IEEE Int. Test Conf.*, Nov. 2010, pp. 1–10.

[9] E. Beckschulze, F. Salewski, T. Siegbert, and S. Kowalewski, "Fault handling approaches on dual-core microcontrollers in safety-critical automotive applications," in *Proc. Int. Symp. Leveraging Appl. Formal Methods Verification Validation*, Oct. 2008, pp. 82–92.

[10] M. Baleani, A. Ferrari, L. Mangeruca, A. Sangiovanni-Vincentelli, M. Peri, and S. Pezzini, "Fault-tolerant platforms for automotive safety-critical applications," in *Proc. ACM Int. Conf. Compilers Archit. Synthesis Embedded Syst.*, Oct./Nov. 2003, pp. 170–177.

[11] M. Rebaudengo, M. S. Reorda, M. Torchiano, and M. Violante, "Soft-error detection through software fault-tolerance techniques," in *Proc. 14th Int. Symp. Defect Fault-Tolerance VLSI Syst.*, 1999, pp. 210–218.

[12] J.-C. Ruiz, P. Yuste, P. Gil, and L. Lemus, "On benchmarking the dependability of automotive engine control applications," in *Proc. Int. Conf. Dependable Syst. Netw.*, Jun./Jul. 2004, pp. 857–866.

[13] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. 20th USENIX Conf. Secur.*, Aug. 2011, p. 6.

[14] T. Hoppe, S. Kiltz, and J. Dittmann, "Automotive IT-security as a challenge: Basic attacks from the black box perspective on the example of privacy threats," in *Proc. Int. Conf. Comput. Safety Rel. Secur.*, Sep. 2009, pp. 145–158.

[15] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, and I. Seskar, "Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study," in *Proc. 19th USENIX Conf. Secur.*, Aug. 2010, pp. 21–21.

[16] A. Groll and C. Ruland, "Secure and authentic communication on existing in-vehicle networks," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2009, pp. 1093–1097.

[17] D. K. Nilsson, U. E. Larson, and E. Jonsson, "Efficient in-vehicle delayed data authentication based on compound message authentication codes," in *Proc. IEEE 68th Veh. Technol. Conf.*, Sep. 2008, pp. 1–5.

[18] B. Groza, S. Murvay, A. van Herrewege, and I. Verbauwhede, *LiBrA-CAN: A Lightweight Broadcast Authentication Protocol for Controller Area Networks*. Darmstadt, Germany: Springer, 2012, pp. 185–200.

[19] C.-W. Lin and A. Sangiovanni-Vincentelli, "Cyber-security for the controller area network (CAN) communication protocol," in *Proc. Int. Conf. Cyber Secur.*, Dec. 2012, pp. 1–7.

[20] K.-D. Kang, Y. Baek, S. Lee, and S. H. Son, "An attack-resilient source authentication protocol in controller area network," in *Proc. ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, May 2017, pp. 109–118.

[21] A. Munir and F. Koushanfar, "Design and performance analysis of secure and dependable cybercars: A steer-by-wire case study," in *Proc. IEEE Consum. Commun. Netw. Conf.*, Jan. 2016, pp. 1066–1073.

[22] Y. Wu, Y.-J. Kim, Z. Piao, J.-G. Chung, and Y.-E. Kim, "Security protocol for controller area network using ECANDC compression algorithm," in *Proc. IEEE Int. Conf. Signal Process. Commun. Comput.*, Aug. 2016, pp. 1–4.

[23] L.-B. Fredriksson, "CAN for critical embedded automotive networks," *IEEE Micro*, vol. 22, no. 4, pp. 28–35, Aug. 2002.

[24] H. Schweppe, T. Gendrullis, M. S. Idress, Y. Roudier, B. Weyl, and M. Wolf, "Securing Car2X applications with effective hardware-software co-design for vehicular on-board networks," in *Proc. Joint VDI/VW Automotive Secur. Conf.*, Oct. 2011, http://www.eurecom.fr/en/publication/3433/detail/securing-car2x-applications-with-effective-hardware-software-co-design-for-vehicular-on-board-networks

[25] T. Ringler, J. Steiner, R. Belschner, and B. Hedenetz, "Increasing system safety for by-wire applications in vehicles by using a time triggered architecture," in *Proc. 17th Int. Conf. Comput. Safety Rel. Secur.*, 1998, pp. 243–253.

[26] H. Zeng, M. D. Natale, P. Giusto, and A. Sangiovanni-Vincentelli, "Using statistical methods to compute the probability distribution of message response time in controller area network," *IEEE Trans. Ind. Informat.*, vol. 6, no. 4, pp. 678–691, Nov. 2010.

[27] F. Koushanfar, A.-R. Sadeghi, and H. Seudie, "EDA for secure and dependable cybercars: Challenges and opportunities," in *Proc. 49th IEEE/ACM Des. Autom. Conf.*, Jun. 2012, pp. 220–228.

[28] C. Paar and J. Pelzl, *Understanding Cryptography*. Berlin, Germany: Springer, 2010.

[29] R. Oppliger, *SSL and TLS: Theory and Practice*. Norwood, MA, USA: Artech House, 2009.

[30] Fujitsu. Secure hardware extension. Feb. 2012. [Online]. Available: https://www.escrypt.com/fileadmin/escrypt/pdf/WEB_Secure_Hardware_Extension_Wiewesiek.pdf

[31] OpenMP, "The OpenMP API specification for parallel programming," Nov. 2012. [Online]. Available: http://openmp.org/wp/

[32] K. Klobedanz, C. Kuznik, A. Thuy, and W. Mueller, "Timing modeling and analysis for AUTOSAR-based software development - a case study," in *Proc. Des. Autom. Test Eur. Conf. Exhib.*, Mar. 2010, pp. 642–645.

[33] Vector, "ECU development and test with CANoe," Nov. 2012. [Online]. Available: http://www.vector.com/vi_canoe_en.html

[34] EETimes, "How secure is AES against brute force attacks?" 2013. [Online]. Available: http://en.wikipedia.org/wiki/Brute-force_attack

[35] C. Dobraunig, M. Eichlseder, and F. Mendel, "Analysis of SHA-512/224 and SHA-512/256," in *Proc. Int. Conf. Advances Cryptology*, Nov./Dec. 2015, pp. 612–630.

[36] Time-stamp counter. Nov. 2012. [Online]. Available: http://www.mcs.anl.gov/kazutomo/rdtsc.html

Arslan Munir (M'09, SM'17) received the MASc degree in electrical and computer engineering from the University of British Columbia (UBC), Vancouver, Canada, in 2007 and the PhD degree in electrical and computer engineering from the University of Florida (UF), Gainesville, Florida, in 2012. He is currently an assistant professor with the Department of Computer Science (CS), Kansas State University (K-State). He holds a Michelle Munson-Serban Simu Keystone Research Faculty Scholarship from the College of Engineering. He was a postdoctoral research associate with the Electrical and Computer Engineering (ECE) Department, Rice University, Houston, Texas from May 2012 to June 2014. From 2007 to 2008, he worked as a software development engineer with Mentor Graphics, Embedded Systems Division. His current research interests include embedded and cyber-physical systems, secure and trustworthy systems, hardware-based security, computer architecture, multicore, parallel computing, distributed computing, reconfigurable computing, artificial intelligence (AI) safety and security, data analytics, and fault tolerance. He received many academic awards including the doctoral fellowship from Natural Sciences and Engineering Research Council (NSERC) of Canada. He earned gold medals for best performance in electrical engineering, gold medals and academic roll of honor for securing rank one in pre-engineering provincial examinations (out of approximately 300,000 candidates). He is a senior member of the IEEE.

Farinaz Koushanfar (M'03, SM'14) received the MA degree in statistics, in 2005, and the PhD degree in electrical and computer engineering, both from the University of California, Berkeley, California. She is a professor and Henry Booker faculty scholar of Electrical and Computer Engineering (ECE), University of California San Diego (UCSD), California. She is co-founder and co-director of the Center for Machine-Integrated Computing and Security (MICS), UCSD. Her research interests include embedded systems security, automated customization of machine learning algorithms, and adaptive embedded systems design. She is a recipient of several awards and honors, including the Presidential Early Career Award for scientists and engineers, the ACM SIGDA Outstanding New Faculty Award, the NAS Kavli Foundation Fellowship, and the Young Faculty (or CAREER) Awards from the Army Research Office (ARO), Office of Naval Research (ONR), Defense Advanced Research Projects Agency (DARPA), and National Science Foundation (NSF). She is a senior member of the IEEE.