Network and System Level Security in Connected Vehicle Applications

(Invited Paper)

Hengyi Liang
Northwestern University
HengyiLiang2018@u.northwestern.edu

Matthew Jagielski
Northeastern University
jagielski.m@husky.neu.edu

Bowen Zheng
University of California, Riverside
bowen.zheng@email.ucr.edu

Chung-Wei Lin
National Taiwan University
cwlin@csie.ntu.edu.tw

Eunsuk Kang
Carnegie Mellon University
eskang@cmu.edu

Shinichi Shiraishi
Toyota InfoTechonology Center, Japan
sshiraishi@jp.toyota-itc.com

Cristina Nita-Rotaru
Northeastern University
c.nitarotaru@neu.edu

Qi Zhu
Northwestern University
qzhu@northwestern.edu

ABSTRACT

Connected vehicle applications such as autonomous intersections and intelligent traffic signals have shown great promises in improving transportation safety and efficiency. However, security is a major concern in these systems, as vehicles and surrounding infrastructures communicate through ad-hoc networks. In this paper, we will first review security vulnerabilities in connected vehicle applications. We will then introduce and discuss some of the defense mechanisms at network and system levels, including (1) the Security Credential Management System (SCMS) proposed by the United States Department of Transportation, (2) an intrusion detection system (IDS) that we are developing and its application on collaborative adaptive cruise control, and (3) a partial consensus mechanism and its application on lane merging. These mechanisms can assist to improve the security of connected vehicle applications.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**; **Systems security**; • **Computer systems organization → Embedded and cyber-physical systems**;

KEYWORDS

Vehicular Network, Security, SCMS, IDS, consensus

ACM Reference Format:
Hengyi Liang, Matthew Jagielski, Bowen Zheng, Chung-Wei Lin, Eunsuk Kang, Shinichi Shiraishi, Cristina Nita-Rotaru, and Qi Zhu. 2018. Network and System Level Security in Connected Vehicle Applications: *(Invited Paper)*. In *IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN (ICCAD '18), November 5–8, 2018, San Diego, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3240765.3243488

ICCAD '18, November 5–8, 2018, San Diego, CA, USA

ACM ISBN 978-1-4503-5950-4/18/11…$15.00
https://doi.org/10.1145/3240765.3243488

1 INTRODUCTION

With the rapid advancement of automotive functionality and architecture, autonomous driving and vehicular ad-hoc networks are likely to become a reality in the near future. These highly autonomous vehicles will be equipped with a number of multi-modal sensors for perceiving the surrounding environment and wireless communication modules for communicating with other vehicles and infrastructures nearby. However, these features for autonomy and connectivity also expose the vehicular systems to cyber-physical attacks via various interfaces, including Bluetooth, Wi-Fi, remote key access, etc. Several such attacks have been demonstrated through concrete experiments on individual vehicles [6, 12, 15].

For connected vehicles in a vehicular network, there are also major concerns on malicious attacks, such as network jamming and flooding that may result in significant packet delays and losses [23], message replay, masquerade attack, and insider attack from compromised vehicles or road side units [24]. As vehicles are working within a dynamic physical world and interacting with a complex external environment, security issue should be considered from an architecture perspective as shown in Figure 1 and detailed below.

- The external network between vehicles should have security mechanisms that are compatible with the existing vehicular communication protocols such as the Dedicated Short Range Communications (DSRC) [10].
- The gateway of a vehicle should have firewalls or intrusion detection systems.
- The in-vehicular network of a vehicle should have light-weight authentication and encryption mechanisms for messages on it.
- The components of a vehicle should have secure storage and manage cryptographic keys.

The Secure Credential Management System (SCMS) [1, 2, 21] has been proposed in recent years to establish trust between distinct entities (vehicles and infrastructures) and provide security and

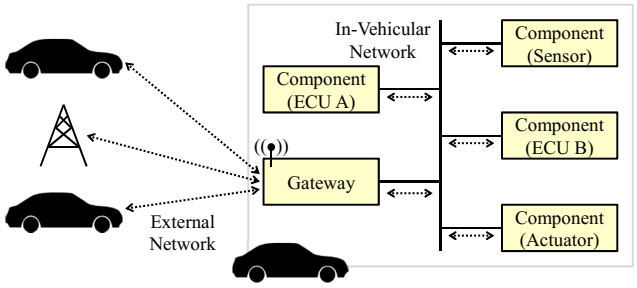


Figure 1: Illustration of in-vehicle architecture and vehicular network communication.

privacy for vehicular networks. Whyte et al. introduced the design of SCMS in [21]. SCMS is more than a traditional public key infrastructure (PKI) as it provides **i)** scalability to support millions of vehicles and **ii)** trade-offs between security, privacy, and efficiency. SCMS provides support to enforce that a message originates from a trusted and legitimate entity.

However, the security mechanisms enabled by SCMS are not designed to protect against insider attacks. An insider is an entity that has been authenticated (e.g., by SCMS or PKI), but compromised later. Several examples include: a sensor is tempered, a hardware or software implementation flaw is discovered, a secret key is leaked, or a legitimate user wants to take advantage and does not follow the protocol correctly. To protect against insider attacks, two techniques can be used, each with a different goal: intrusion detection – to detect anomalies, and Byzantine-resilient consensus algorithms – to ensure a consistent view of the state of the system regardless of the presence of compromised participants. Such algorithms are a good fit for defending against insider attacks and can work in conjunction with SCMS to provide insider-resilient security services.

Intrusion detection has been used before to detect in-car attacks. Following a widely publicized take over of the breaking and steering system of a Jeep [15] by injecting controller area network (CAN) messages at higher frequency, an intrusion detection system (IDS) [19, 20] was proposed to monitor tasks and detect timing violation of periodic CAN bus messages. Cho et al. [7] also proposed an IDS by fingerprinting ECUs.

Byzantine-resilient algorithms [14, 16] and [3, 4] have been traditionally used to agree on a common perception of the system state, in spite of compromised participants. Thus, they are not concerned with detection, but with operating through attacks. To the best of our knowledge, such algorithms have not been used in the context of vehicular networks.

In this paper, we will discuss security issues related to SCMS, and present two defense mechanisms that are work-in-progress. The rest of the paper is organized as follows. Section 2 introduces a credential management system to authenticate vehicular network messages and entities. Section 3 discusses an intrusion detection system and its application on collaborative adaptive cruise control. Section 4 studies the importance and complexity of consensus for connected vehicle applications, and presents a partial consensus mechanism and its application on lane merging.

2 SECURITY CREDENTIAL MANAGEMENT SYSTEM

All networked systems, need a secure form of identification, and all security services rely on public keys. Thus, credential systems and PKIs are fundamental building blocks for secure systems. Considering for example a low level communication protocols like DSRC, authentication and non-reputation of messages are needed, which in turn require digital signatures, and public keys respectively. More detailed credentials might be needed to ensure authorized access to different components or to perform certain operations by entities involved in a connected car architecture. Credentials systems might rely on PKI to achieve their own functionality.

In this section, we overview one of the proposals for a credential management systems for connected cars, and examine potential attacks against it and the impact against such attacks.

2.1 SCMS Overview

We studied three documents to understand SCMS design: a research paper [21], a white paper [1], and the proof-of-concept implementation requirements and specifications [2]. According to [1], SCMS is a framework to issue and manage digital certificates that form the basis of secure communication between connected cars, or cars and vehicular infrastructure. The main goal is to provide public keys used to digitally sign all messages exchanged. It uses an architecture where several certificate authorities (CAs) establish a chain of trust and issue several certificates with different lifetime validity.

There are two type of entities that must be registered in the system and obtain certificates: OBE — onboard equipment, basically identifying a vehicle, and RSE — road side entities, identifying other entities communicating with the vehicles.

There are several types of CAs and several types of certificates, with different goals and validity. In terms of CAs, the following CAs are planned: Enrollment CA, Root CA, Intermediate CA, Pseudonym CA. From the available document it is not clear what is the hierarchy between these CAs. The Enrollment CA is most likely in charge of issuing OBE Enrollment Certificates (see below). The role of the others is not very clear, however it does seem that the Root CA might be in charge of the short-term certificates.

In the case of an OBE, the system distinguishes between network layer authentication, achieved through Pseudonym certificates, and application authentication, achieved through Identification certificates. Specifically, according to [1], type of certificates for an OBE are: (1) OBE Enrollment Certificate — long-term certificate used to request other certificates (pseudonym and identification certificates). (2) Pseudonym Certificate — short term and used primarily for basic safety message (BSM) authentication and misbehavior reporting. A device is given multiple certificates that are valid simultaneously, so that it can change them frequently. (3) Identification Certificate — short term and used for authorization in V2I applications. An OBE has only one identification certificate valid at a time for a given application.

For an RSE, the same type of mechanisms where a pair of long-term certificate and short term certificates are used. Specifically, an RSE uses (1) Enrollment Certificate — long term and used to request application certificates. A certification process will provide authorization for RSEs to interface with the SCMS and request an

enrollment certificate during the bootstrap process. (2) Application Certificate — Application certificates are used by an RSE to sign any over-the-air messages transmitted, such as signal phase and timing or traveler information message. There is only one application certificate valid at a time for a given application.

The architecture also specifies a misbehaving detection entity that is supposed to communicate with the Root CA.

2.2 Comments on the SCMS Architecture

Below we list several comments regarding the SCMS architecture and questions that require further study.

1. Reliance on National Institute of Standards and Technology (NIST) authenticated Network Time Protocol (NTP). It appears that clock synchronization is a requirement for SCMS, and the proof of concept requirements specify the use of the NIST authenticated NTP servers. Given the recent attacks shown against NTP, using an authenticated service is a step in the right direction.

2. Size of Certificate Revocation Lists (CRLs). One of the major problems for PKIs is how to deal with compromised certificates. SCMS uses CRLs to address this issue. These lists are distributed periodically, and they are digitally signed. The Transport Layer Security (TLS)/web security community has been moving away from CRLs to Online Certificate Status Protocol (OCSP), mainly for scalability reasons. The size of a CRL can get very big as the number of participants grows, and thus revoked certificates can exist. Also, it requires participants to actively retrieve them, resulting often in-and-out of data information. OCSP itself has limitations — it adds delays, it requires the server to be present, and it leaves the client wondering in case of failures. OCSP with stapling addresses some of these issues. Recent work [11] has been looking at reducing the size of these revocation lists.

3. Use of TLS and interaction of TLS with the SCMS. It looks like the proposed proof-of-concept relies on TLS for the devices to communicate with the SCMS components. Better understanding is needed on how TLS certificates are generated, used, deployed, and revoked.

4. Provable security for the used cryptographic constructions. All the cryptographic constructions should be subjected to security analysis with the goals of creating models for provable security. This will not apply for example for TLS communication subjected to TLS security proofs, but for any other form of secure communication for which such proofs do not exist.

5. Synchronization attacks on the OBE. It was not clear what are the time synchronization requirements with respect to OBE. For example, attackers can influence the local clock.

6. Availability of the enrollment CA and the other CAs. Given that the white paper talks about accessing these CAs over the Internet, operating when they are not available should be considered. Centralization makes for easy management but does not operates well with failures or under denial of service attacks.

7. Containment and contingency plan if any of the CAs is compromised. Compromise of the main root CA is a very serious problem and unfortunately in the last few years several such incidents occurred. The architecture should take into account such a worst case scenario.

Relation between SCMS and IDS. As mentioned above, the SCMS architecture specifies a misbehaving detection entity that is assumed to communicate with the Root CA and inform detected misbehavior with respect to the content of BSMs. An IDS can be integrated with the misbehaving authority to inform it about misbehaving entities. Reports of this misbehavior can include proofs in the form of messages signed and contain the malicious information (thus justifying the need for non-repudiation). Note that some of these messages may be merely a reflection of attack propagation from along the platoon and not a malicious attack in itself. Thus, integration of such information, say into a reputation system, must be done very carefully not to punish honest participants.

3 INTRUSION DETECTION FOR ATTACKS AGAINST COLLABORATIVE ADAPTIVE CRUISE CONTROL

In this section we describe an intrusion detection approach to detecting anomalies in applications for vehicular networks. Specifically, we focus on adaptive cruise control as a representative application.

3.1 Attacks against Collaborative Adaptive Cruise Control

Collaborative adaptive cruise control (CACC) extends traditional adaptive cruise control (ACC) by involving the preceding car into the acceleration computation. Specifically, in addition to measurement from on-board sensors like RADAR or LIDAR, the new acceleration computation also leverages the acceleration information of the preceding car, obtained through DSRC communication. In such application, messages containing safety-critical information (e.g.,, vehicle acceleration rate) are exchanged among vehicles via DSRC BSMs. Each vehicle not only sends and receives messages, but also works as a router for forwarding messages.

For CACC, we consider the attacks identified in [9]. Specifically, we focus on the POS attack and VEL attack. The POS attack occurs when the attacker has the ability to modify LIDAR (position) sensor values. It operates by slowly increasing the distance measured to the direct leader so that the follower will overestimate the gap and follow too closely. Such attack is able to reduce the safety of the algorithm and increase the likelihood of a crash. The VEL attack takes place when the attacker can modify RADAR (velocity) sensor values, and works similarly.

3.2 Intrusion Detection System

We propose an anomaly detection scheme based on the Principal Component Analysis (PCA). PCA fits a Gaussian model to the data, and uses this representation of normal behavior to detect anomalies. This allows the model to construct a rich representation of the data. PCA uses an eigen decomposition of the data's variance-covariance matrix in order to produce a few new features which explain as much of the variance in the original data as possible. If the eigenvalue-eigenvector pairs of this matrix are $(\lambda_1, x_1), (\lambda_2, x_2), \ldots, (\lambda_n, x_n)$, then the proportion of the variance explained by the ith eigenvector is

$$\frac{\lambda_i}{\sum_j \lambda_j}.$$

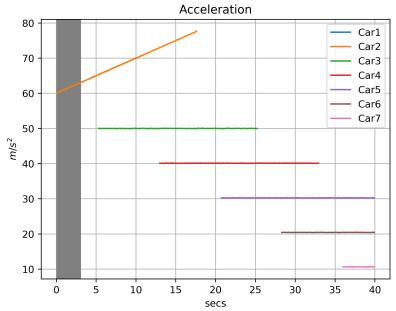


Figure 2: PCA defense with no attack. Note the large number of false positives.

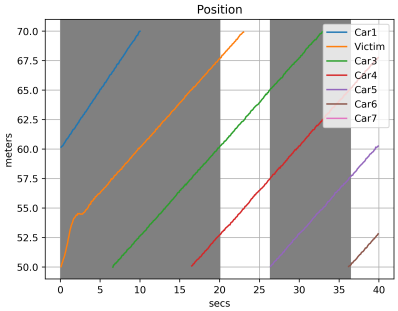


Figure 3: PCA defense against VEL attack with $c_v = 1$. Detected attacks are shown with a gray line. PCA is quite effective at detecting this attack.

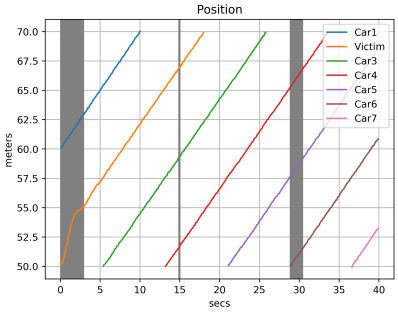


Figure 4: PCA defense against VEL attack with $c_v = 0.1$. Detected attacks are shown with a gray line.

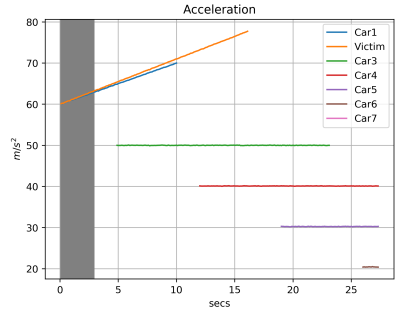


Figure 5: PCA defense against POS attack. Detected attacks are shown with a gray line. PCA is effective in the beginning but stops being effective over time.

Note that, because the variance-covariance matrix is positive semi-definite, all eigenvalues are nonnegative. In order to compute the probability of a sample s, we write the sample as a linear combination of the eigenvectors

$$s = s_1 x_1 + s_2 x_2 + \cdots s_n x_n,$$

and compute the Mahalanobis distance from the mean as

$$d(s) = \sum_i \frac{s_i^2}{\lambda_i}.$$

A simple thresholding of this distance can be used to classify a point as an anomaly. For more information on the computation of PCA and its use for anomaly detection, we refer the interested reader to Section 2.2 of [18].

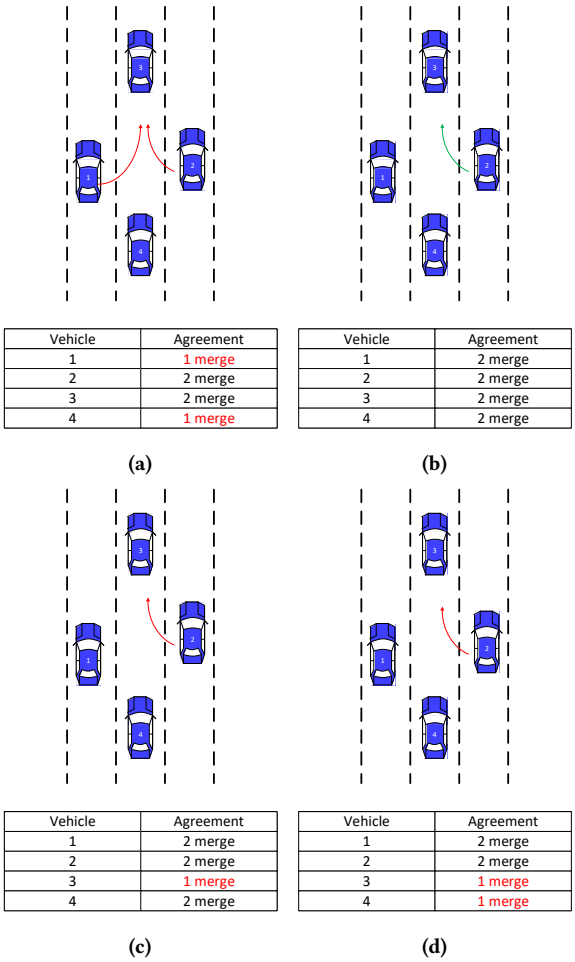
In our work, the features we supply are $p_{old}, p_{new}, v_{old}, v_{new}, a_{old}, a_{new}, p_{lead}, v_{lead}, a_{lead}$, where p_{old} and p_{new} are the old and new position values of the ego vehicle, v_{old} and v_{new} are the old and new velocities of the ego vehicle, a_{old} and a_{new} are the old and new acceleration of the ego vehicle, and $p_{lead}, v_{lead}, a_{lead}$ are DSRC-transmitted position, velocity, and acceleration of the leading vehicle. We select the first two principal components as we find this to model the original data and its interactions between variables well.

Note that with these features, a model is represented by up to 9 vectors of dimension 9; anomaly detection is also fast — up to 9 projections onto these dimension 9 vectors. Data storage is only 6 points per observation — position, velocity, and acceleration of both the car and its leader. This gives a detection technique with very low storage and computation overhead.

3.3 Simulation Results

We focus on VEL and POS attacks, i.e., modifying the values of velocity and position, respectively. We experiment with the number of principal components and the distance threshold in order to minimize false positives.

PCA is very effective at detecting the VEL attack with the lying magnitude $c_v = 1$, as seen in Figure 3. Even with the lying magnitude $c_v = 0.1$, as seen in Figure 4, PCA is fairly effective at detecting the attack. However, it has two failings, both stemming from the Gaussian assumption. The first is its high false positive rate, which detects benign points as malicious frequently, especially in the starting period of CACC, where velocities change frequently. This is visible when there is no attack as shown in Figure 2. The second is that it is unable to detect the POS attack as effectively as the VEL attack (see Figure 5). This is because that the POS attack is comparatively more effective, and so works with a lie of less magnitude. Due to these weaknesses, we have also looked to Hidden Markov Models (HMMs) to fill these gaps.



Vehicle	Agreement
1	1 merge
2	2 merge
3	2 merge
4	1 merge

(a)

Vehicle	Agreement
1	2 merge
2	2 merge
3	2 merge
4	2 merge

(b)

Vehicle	Agreement
1	2 merge
2	2 merge
3	1 merge
4	2 merge

(c)

Vehicle	Agreement
1	2 merge
2	2 merge
3	1 merge
4	1 merge

(d)

Figure 6: (a) Only partial agreement reached. Unsafe to merge as vehicles 1 and 2 may collide in the middle lane. (b) Global consensus/agreement reached. Safe to merge. (c) Partial agreement reached among vehicles 1, 2 and 4. Vehicle 2 may merge into the middle lane based on its own sensors and the communication with 4, but the case is not as safe as in case (b). (d) Partial agreement reached between vehicles 1 and 2, and between 3 and 4. Vehicle 2 may still be able to merge into the middle lane based on its own sensors, but the case is not as safe as the ones in (b) and (c).

4 VEHICULAR NETWORK CONSENSUS

In this section, we investigate the importance of consensus in a connected vehicle environment, where the participants (vehicles and infrastructures) have to reach *certain level of agreement* to ensure the desired system properties, such as safety, liveness and deadlock-free.

Plenty of work have presented various solutions to the classical consensus problems with respect to different assumptions on synchrony or asynchrony of the system, failure types, and requirements of termination. Early techniques [14, 17] that tolerate Byzantine faults [14] rely on the synchrony of system, and thus are not suitable for vehicular networks. Lamport et al. [13] provide a fault-tolerant solution for asynchronous distributed system. However, it only assumes fail-stop faults. Feng et al. [22] address the asynchronous consensus challenge in a multi-agent system. Their approach overcomes the difficulties of unreliable communication networks and is promising for connected vehicle applications that work in continuous-state domain. However, the failure of agents is not explicitly considered in their control strategy.

Next, we will first use a motivating example to demonstrate various scenarios of consensus in connected vehicle applications. We will then discuss the difficulties of reaching global consensus in vehicular networks, especially when under malicious attacks, and present a security-aware consensus protocol that is based on reaching required *partial consensus*.

4.1 Motivating Example

A motivating example is shown in Figure 6, where four vehicles are communicating with each other and try to reach an agreement on lane merging choice, i.e., whether vehicle 1 or 2 should merge into the middle lane. In Figure 6 (a), no global consensus is reached. Only partial agreement is reached between vehicles 1 and 4 (that vehicle 1 will merge into the middle lane), and between vehicles 2 and 3 (that vehicle 2 will merge into the middle lane), respectively. In this case, it would be unsafe to perform the lane merging, as vehicles 1 and 2 could merge into the middle lane at the same time. In Figure 6 (b), all four vehicles reach a global consensus that vehicle 2 could merge into the middle lane. In this case, the lane merging could be safely performed, assuming the vehicles use their sensors and possibly additional communication via DSRC to maintain a safe distance between each other during the merging.

There are more complex scenarios with different levels of agreement. In Figure 6 (c), partial agreement is reached between vehicles 1, 2 and 4 to let vehicle 2 merge. It should be mostly safe to perform the lane merging in this case. However, as vehicle 3 thinks vehicle 1 will merge into the space behind it, possibly with slower speed and larger initial distance between them than vehicle 2 merging (or vehicle 3 could be unaware of any merging at all), this case is not as safe as the one in Figure 6 (b). In Figure 6 (d), partial agreement is reached between vehicles 1 and 2 to let vehicle 2 merge; while vehicles 3 and 4 think vehicle 1 will merge. It may still be possible to perform the lane merging, but the case is not as safe as the ones in Figure 6 (b) and 6 (c) — vehicles 3 and 4 could accelerate or brake thinking it should be safe for vehicle 1, while vehicle 2 is the one that is actually merging.

4.2 Consensus Challenges in Vehicular Networks and Partial Consensus

The example in Figure 6 shows the complexity and importance of addressing consensus in connected vehicle applications. On one hand, these applications are built on distributed systems and are often asynchronous. The vehicular communication network is ad-hoc and its topology changes dynamically. There could be Byzantine faults caused by equipment failure, packet losses, or security attacks (e.g., jamming and flooding attacks on communication channels, or insider attacks). With these challenges, it may take significant amount of time (if ever) to reach global consensus in connected

vehicle applications. Note that in traditional computing systems , it has been long shown that consensus without sacrificing liveness is impossible for asynchronous distributed systems, known as the FLP Impossibility [8]. In *time-critical* connected vehicle applications, waiting for a long time to reach global consensus not only affects liveness property, it may significantly worsen system performance and even cause incorrect functionality — as the physical environment and system dynamics evolve over time.

On the other hand, for many connected vehicle applications, partial agreement among participants may already be sufficient to achieve the desired functionality and performance. As shown in Figure 6 (c) and Figure 6 (d), the lane merging could be safely performed with partial agreement *and* additional constraints on how vehicles 3 and 4 may operate. Next, we will address such notion of partial consensus/agreement in lane merging application. We will leverage the consensus strategy as discussed in [3] and [5], to coordinate a local group of n autonomous vehicles with the consideration of Byzantine faults in an asynchronous system.

4.3 Partial Consensus for Lane Merging

System Model. We assume that vehicles are free to propose lane-change (lane merging) maneuvers that are allowed in traffic. Let $c_k = (l_s, l_d, p_i)$ denote a lane-change maneuver for vehicle p_i to go from a starting lane l_s to a destination lane l_d. Let $C = \{c_1, \ldots, c_m, \phi\}$ denote the set of allowed lane-change maneuvers, including the special case of no lane-change, denoted by ϕ. Every time a vehicle p_i intends to make a lane-change, it has to first propose an $x_i \in C$ and broadcast x_i to all other vehicles. Let $X = \{x_i | 1 \leq i \leq n\} \subseteq C$ be the collective set of all the initial proposed lane changes. Finally, we define *k-set consensus* as: each vehicle p_i has to decide on a single choice $y_i \in X$ and the size of the collective set of the decided values $F = \{y_i\}$ is at most k.

Broadcast Primitive. We leverage the broadcast primitive presented in [3] to facilitates all correct vehicles to communicate with each other and validate/accept values (an abstract notion representing any concrete information in practice), as shown in Algorithm 1. Three types of messages are used during the broadcast primitive procedure: *Initial*, *Echo* and *Ready*. We require vehicle p_i to first broadcast its value v_i through an initial message *Initial*(v_i). When any vehicle p_j receives *Initial*(v_i) or enough number of *Echo* or *Ready* messages from other vehicles, p_j broadcasts a message *Echo*(v_i,p_j) to inform all other vehicles. When p_j knows that enough number of vehicles have received messages about v_i, it broadcasts a message *Ready*(v_i,p_j). Finally, once vehicle p_j has received enough *Ready*(v_i,p_j) messages, it validates and accepts v_i.

The k-set Consensus Protocol. We integrate the above broadcast primitive procedure into the k-set agreement protocol proposed in [5], and apply it to our lane changing application, as shown in Algorithm 2. Each vehicle p_i keeps a state vector T_i to record the state (in this case the lane-change proposals) of all vehicles. In Step 1, if vehicle p_i proposes a lane-change choice $x_i \in C$, it sets corresponding entry $T_i[i] = x_i$, otherwise $T_i[i] = \phi$. An partial order $\leq$ is defined on these state vectors. $T_i \leq T_j$ if $\forall k \leq n, T_i[k] = \phi \vee T_i[k] = T_j[k]$. Moreover, $T_i < T_j$ if $T_i \leq T_j \wedge T_i \neq T_j$.

In Step 3, the *Update*(T_i, T_j) function updates the state vector of a vehicle p_i, following two rules: **a)** if $T_i[k] = \phi \wedge T_j[k] = \phi$ then $T[k] = \phi$, and **b)** if $T_i[k] = x_i \vee T_j[k] = x_i$ then $T[k] = x_i$. We use a *Timeout()* function to help terminate the process in case of long communication delay or persistent packet losses (either due to unreliable communication channels or malicious attacks such as jamming or flooding). In Step 4, the *Decide*(T_i) function makes a lane-changing decision $y_i \in X$ from the proposals in T_i. The decision could be based on the priority among vehicles, the urgency of the merging, the estimated time for merging, etc. If the *Timeout()* function is not evoked, the protocol ensures that there are at most k decisions at the end of the protocol.

Algorithm 1: Broadcast Primitive: BroadcastPrimitive(v_i, p_i)

```
Vehicle p_i broadcasts Initial(v_i) to all other vehicles;
for each vehicle p_j do
    Step 1: wait until the receipt of Initial(v_i), or (n + t)/2
      Echo(v_i,p_k), or (t + 1) Ready(v_i,p_k) messages from other
      vehicles (with various k indices), p_j broadcasts an
      Echo(v_i,p_j) to all other vehicles;
    Step 2: wait until the receipt of (n + t)/2 Echo(v_i,p_k) or
      (t + 1) Ready(v_i,p_k) messages, p_j broadcasts a
      Ready(v_i,p_j) to all other vehicles;
    Step 3: wait until the receipt of (2t + 1) Ready(v_i,p_k)
      messages, p_j validates and accepts value v_i;
```

Algorithm 2: Asynchronous k-set Consensus Protocol

```
for each vehicle i do
    Step 1: Construct an initial vector T_i:
        T_i[i] = x_i if p_i decide to change lane or φ otherwise
        ∀ j ≠ i, T_i[j] = φ
    Step 2: BroadcastPrimitive(T_i, p_i); set r = 1
    Step 3:
    while not Timeout() do
        for each received vector T_j do
            if received vector T_j is not a decision vector then
                if T_j < T_i then
                    continue ;
                else if T_j == T_i then
                    r = r + 1
                    if r < n-k+1 then
                        continue ;
                    else
                        break ;              // go to step 4
                else
                    T_i = Update(T_i, T_j), go to step 2 ;
            else
                set T_i = T_j ;
    Step 4: y_i = Decide(T_i); broadcast T_i as a decision vector
```

System Safety Objective Evaluation. Note that the partial consensus protocol in Algorithm 2 ends whenever there are no more than k different decisions. It does not distinguish between different

partial consensus scenarios that have the same number of decisions. As we can see from Figure 6 (a), (c) and (d), which all have two decisions, these scenarios may exhibit different levels of safety. Thus, even though knowing the number of decisions provides useful information on system safety (e.g., global consensus with one single decision is the best scenario), we need to further evaluate the various partial consensus scenarios individually (a limitation of Algorithm 2). Next, we will discuss how this may be done for the lane merging application.

Upon the termination of Algorithm 2, we assume the size of the collective decision set is $k' \leq k$. Let P_i denote the set of vehicles that decide on the same lane-change choice y_i. We consider the following scenarios:

- **Scenario 1**: All vehicles agree on the same choice y, i.e, $k' = 1$. In this case, the system has the highest level of safety, similar to the case in Figure 6 (b).
- **Scenario 2**: There exist two lane-change decisions y_i and y_j with the same destination lane, initially proposed by vehicles p_u and p_v, respectively. Furthermore, $p_u \in P_i$, and $p_v \in P_j$. That is, p_u and p_v both think they can perform the lane change, and the destination lane is the same. This case has the lowest level of safety, similar to the case in Figure 6 (a).
- **Scenario 3**: First, the condition for Scenario 2 is not true. Furthermore, there exist two lane-change decisions y_i and y_j with the same destination lane. This means that there are vehicle(s) that do not propose to change lane themselves, but have a wrong understanding of (or not aware of) which vehicle(s) will perform the lane change. This is similar to the cases in Figure 6 (c) and (d).

The quantitative evaluation of different scenarios, in particular Scenario 3, could be quite complicated. It will depend on the involved vehicles' current positions, speeds, accelerations, and very importantly, their corresponding decision y_i (i.e., their understanding of which vehicle(s) may perform lane change). We plan to investigate this in our future work.

5 SUMMARY

In this paper, we addressed some of the security issues in connected vehicle applications, with consideration of attacks on communication channels and insider attacks. We presented and discussed three defense mechanisms targeting safety, authenticity, and security. Possible future work includes development of more thorough and general defense mechanisms and the integration of multi-dimension security services.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation Grants 1646641, 1834324, and 1834701.

REFERENCES

[1] 2-17. SECURITY CREDENTIAL MANAGEMENT SYSTEM (SCMS) PROOF OF CONCEPT (POC). https://www.its.dot.gov/factsheets/pdf/CV_SCMS.pdf
[2] 2016. Security Credential Management System Proof-of-Concept Implementation EE Requirements and Specifications Supporting SCMS Software Release 1.0.
[3] Gabriel Bracha. 1987. Asynchronous Byzantine agreement protocols. *Information and Computation* 75, 2 (1987), 130–143.
[4] Christian Cachin, Klaus Kursawe, and Victor Shoup. 2005. Random Oracles in Constantinople: Practical Asynchronous Byzantine Agreement Using Cryptography. *Journal of Cryptology* 18, 3 (01 Jul 2005), 219–246. https://doi.org/10.1007/s00145-005-0318-0
[5] Soma Chaudhuri. 1993. More choices allow more faults: Set consensus problems in totally asynchronous systems. *Information and Computation* 105, 1 (1993), 132–158.
[6] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. 2011. Comprehensive experimental analyses of automotive attack surfaces.. In *USENIX Security Symposium*. San Francisco, 77–92.
[7] Kyong-Tak Cho and Kang G. Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 911–927. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/cho
[8] Michael J Fischer, Nancy A Lynch, and Michael S Paterson. 1985. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)* 32, 2 (1985), 374–382.
[9] Matthew Jagielski, Nicholas Jones, Chung-Wei Lin, Cristina Nita-Rotaru, and Shinichi Shiraishi. 2018. Threat Detection for Collaborative Adaptive Cruise Control in Connected Cars. In *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '18)*. ACM, New York, NY, USA, 184–189. https://doi.org/10.1145/3212480.3212492
[10] John B Kenney. 2011. Dedicated short-range communications (DSRC) standards in the United States. *Proc. IEEE* 99, 7 (2011), 1162–1182.
[11] Mohammad Khodaei and Panos Papadimitratos. 2018. Efficient, Scalable, and Resilient Vehicle-Centric Certificate Revocation List Distribution in VANETs. In *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '18)*. ACM, New York, NY, USA, 172–183. https://doi.org/10.1145/3212480.3212481
[12] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. 2010. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 447–462.
[13] Leslie Lamport et al. 2001. Paxos made simple. *ACM Sigact News* 32, 4 (2001), 18–25.
[14] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4, 3 (1982), 382–401.
[15] Charlie Miller and Chris Valasek. 2015. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA* 2015 (2015), 91.
[16] M. Pease, R. Shostak, and L. Lamport. 1980. Reaching Agreement in the Presence of Faults. *J. ACM* 27, 2 (April 1980), 228–234. https://doi.org/10.1145/322186.322188
[17] Marshall Pease, Robert Shostak, and Leslie Lamport. 1980. Reaching agreement in the presence of faults. *Journal of the ACM (JACM)* 27, 2 (1980), 228–234.
[18] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. 2003. *A novel anomaly detection scheme based on principal component classifier*. Technical Report. MIAMI UNIV CORAL GABLES FL DEPT OF ELECTRICAL AND COMPUTER ENGINEERING.
[19] H. M. Song, H. R. Kim, and H. K. Kim. 2016. Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In *2016 International Conference on Information Networking (ICOIN)*, Vol. 00. 63–68. https://doi.org/10.1109/ICOIN.2016.7427089
[20] P. Waszecki, P. Mundhenk, S. Steinhorst, M. Lukasiewycz, R. Karri, and S. Chakraborty. 2017. Automotive Electrical and Electronic Architecture Security via Distributed In-Vehicle Traffic Monitoring. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36, 11 (Nov 2017), 1790–1803. https://doi.org/10.1109/TCAD.2017.2666605
[21] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn. 2013. A security credential management system for V2V communications. In *2013 IEEE Vehicular Networking Conference*. 1–8. https://doi.org/10.1109/VNC.2013.6737583
[22] Feng Xiao and Long Wang. 2008. Asynchronous consensus in continuous-time multi-agent systems with switching topology and time-varying delays. *IEEE Trans. Automat. Control* 53, 8 (2008), 1804–1816.
[23] Bowen Zheng, Chung-Wei Lin, Huafeng Yu, Hengyi Liang, and Qi Zhu. November 2016. CONVINCE: A Cross-Layer Modeling, Exploration and Validation Framework for Next-generation Connected Vehicles. In *Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on*. Article 37, 8 pages. https://doi.org/10.1145/2966986.2980078
[24] B. Zheng, M. O. Sayin, C. W. Lin, S. Shiraishi, and Q. Zhu. 2017. Timing and security analysis of VANET-based intelligent transportation systems: (Invited paper). In *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 984–991. https://doi.org/10.1109/ICCAD.2017.8203888