Security Vulnerabilities of Connected Vehicles Streams and their Impact on Cooperative Driving

Mani Amoozadeh[1], Arun Raghuramu[2], Chen-Nee Chuah[1], Dipak Ghosal[2]
H. Michael Zhang[3], Jeff Rowe[2], Karl Levitt[2]

[1]Department of Electrical & Computer Engineering, University of California, Davis, CA, USA
[2]Department of Computer Science, University of California, Davis, CA, USA
[3]Department of Civil and Environmental Engineering, University of California, Davis, CA, USA
{maniam, araghuramu, chuah, dghosal, hmzhang}@ucdavis.edu, {rowe, levitt}@cs.ucdavis.edu

***Abstract*—Autonomous vehicle capable of navigating unpredictable real-world environments with little human feedback are a reality today. Such systems rely heavily on on-board sensors such as cameras, radar/LIDAR, and GPS as well as capabilities such as 3G/4G connectivity and V2V/V2I communication to make real time maneuvering decisions. Autonomous vehicle control imposes very strict requirements on the security of the communication channels used by the vehicle to exchange information as well as the control logic that performs complex driving tasks, e.g., adapting vehicle velocity, or changing lanes. This study presents a first look at the effects of security attacks on the communication channel as well as sensor tampering of a connected vehicle stream equipped to achieve Cooperative Adaptive Cruise Control (CACC). Our simulation results show that an insider attack can cause significant instability in the CACC vehicle stream. We also illustrate how different countermeasures, such as downgrading to ACC mode, could potentially be used to improve security and safety of the connected vehicle streams.**

I. Introduction

Autonomous driverless cars have recently received much publicity with successful demonstrations by Google, whose self-driving car has completed over 700,000 autonomous-driving miles across cities in the U.S.[1]. Levinson et al. [1] present algorithms used in "Junior" –Stanford's autonomous vehicle that is capable of performing complex driving tasks in real time in unpredictable traffic conditions. While these ambitious forward-looking projects are currently in the proof-of-concept stage, lower levels of *function-specific automation* like Cruise Control (CC), and *combined function automation*[2] such as ACC (Adaptive Cruise Control) are already finding their way into deployment in certain high-end automobiles. CACC (Cooperative Adaptive Cruise Control) is an extension to the ACC that leverages inter-vehicle communications to create tightly-coupled vehicle stream [2].

To achieve automated cooperative driving, vehicles need to have access to each other's information. Such information enhances the ability of the autonomous vehicle to plan ahead and make better decisions to improve the overall safety and performance of the vehicle. One possible way of achieving inter-vehicle communication is through Vehicular Ad-hoc Networks (VANET). We refer to vehicles that participate in information-sharing and cooperative driving through VANET communications as *connected vehicle stream*. Although VANET technologies have matured over time, they are still riddled with security issues. Engoulou et al. [3] present the most recent survey of various security issues in VANETs including security requirements, attacks, and privacy protection. Faults in software components can potentially lead to devastating effects for autonomous vehicles and other vehicles sharing the roadway. Thus, it is important to design the system to be robust, secure against malicious attacks, privacy preserving and fault tolerant.

The *Security Credential Management System (SCMS)*, developed under a cooperative agreement with the United States Department of Transportation (USDOT), is currently the leading candidate for V2V security backend design in the U.S. [4]. These efforts focus on securing the V2X communication channels over *Wireless Access in Vehicular Environment (WAVE)* standard. Relatively, little attention has been given to the safety and security of VANET control protocols in autonomous vehicles, and the impact of different security attacks in the presence of an adversary and bad-faith participants. Compromising sensor input or control protocols can lead to incorrect decisions made by autonomous vehicles, leading to dynamic interactions between vehicles that threaten the stability and safety of autonomous vehicle streams.

Blum et al. [5] argue that jamming attacks can cause collisions in a vehicle platoon, leading to serious multicar pile-up. Guette et al. [6] present a security model based on Trusted Platform Module (TPM), and describe an application of cooperative driving and its associated threat model. These prior works only discuss a limited number of attacks and none of them consider the actual vehicle longitudinal control in CACC. As a result, these studies ignore other impacts that a security attack might have on a platoon such as string instability. Moreover, none of these studies have carried out

This work is partly supported by NSF CMMI-1301496 grant and Intel Science and Technology Center for Secure Computing.

[1]Google Official Blog, The latest chapter for the self-driving car: mastering city street driving

[2]U.S. Department of Transportation, Policy on Automated Vehicle Development

any quantitative analysis of the impact of the attacks. Our main contributions in this work are:

Analysis of security risks in autonomous vehicle stream: we perform a study of the security vulnerabilities and risks associated with deploying VANET communication in connected vehicle streams to achieve automated sensing and control such as CACC. We consider various types of what-if scenarios when communications between autonomous vehicles participating in CACC are compromised. In addition, we examine existing countermeasures, explore limitations of these methods and possible ways to alleviate negative effects.

Quantitative analysis of security attacks: VENTOS (VEhicular NeTwork Open Simulator) is an integrated open-source simulator we have developed to study VANET-based traffic applications. CACC car-following model based on one-vehicle look-ahead communication utilizing IEEE 802.11p were implemented in VENTOS in order to study various what-if scenarios involving security attacks on CACC vehicle stream. In particular, we present simulation results on the impact of application and network layer attacks by a malicious insider on the performance of a CACC vehicle stream.

II. Cooperative Adaptive Cruise Control (CACC)

CACC is an enhancement of ACC, and we consider it as a case study of cooperative driving of autonomous vehicles. CACC incorporates wireless V2V communication to access rich preview information about the surrounding vehicles. This leads to tighter following gaps and faster response to changes than ACC, and makes collaborative driving such as platooning feasible. In this study, parameter exchange in CACC vehicle stream is done through *beacons* that are periodic single-hop messages broadcasted by each vehicle.

We consider a CACC vehicle stream that is moving on a straight single-lane highway. The vehicles utilize a simple one-vehicle look-ahead communication scheme as shown in Fig. 1a. Each CACC vehicle is listening to beacon messages sent wirelessly using IEEE 802.11p from its immediate preceding vehicle. The vehicles then utilize the speed, position, acceleration and other information embedded in these beacon messages to achieve a distributed longitudinal control. Details on the CACC control design are discussed in [2].

We assume that the platoon of vehicles is already formed and is traveling on a straight single-lane highway with no need to change the platoon size or perform maneuvers (split, merge, leave, etc). There will be no need for a *platoon management protocol* and the platoon leader acts only as the first vehicle in the platoon and makes disturbance by slowing down or speeding up. Thus, the only active communication between CACC vehicles is beaconing that is used to exchange necessary parameters for longitudinal controller.

III. Security Attacks on CACC Vehicle Stream

We group the security attacks on a CACC vehicle stream as application layer, network layer, system level, and privacy leakage attacks. All these attacks can potentially impact the string stability of the system and compromise the safety and privacy of the passengers of the CACC vehicle stream. Such attacks can be launched either by an outsider or insider adversary. While leveraging state-of-the-art security architectures can potentially limit the capabilities of outsider attacks, there can still be disruptive insider attacks. In the following subsections, we discuss the various categories of security attacks in more depth.

A. Application Layer Attacks

Application layer attacks affect the functionality of a particular application such as CACC beaconing, or message exchange in the platoon management protocol. The adversary can use message falsification (modification), spoofing (masquerading), or replay attacks to maliciously affect the vehicle stream. The impact of such attacks can be a temporary instability in the vehicle stream and in severe cases it can lead to rear-end collision.

In the *message falsification attack*, the adversary starts listening to the wireless medium, and upon receiving each beacon, manipulates the content meaningfully and rebroadcasts it as depicted in Fig. 2a. Changing the value of different fields in a beacon might have different effects on the system depending on the implementation of the vehicle's longitudinal control system. For instance, changing the acceleration field might have a more significant effect than changing the velocity.

In the *spoofing attack*, the adversary impersonates another vehicle in the stream in order to inject fraudulent information into a specific vehicle. In one-vehicle look-ahead communication, the adversary can impersonate the vehicle preceding the target vehicle, even when the attacker is physically distant from the target vehicle. Note that '*in-transit traffic tampering attack*' in order to modify, delay, or drop messages is not applicable in CACC vehicle stream. This is due to the fact that all communications are single-hop and no vehicle is acting as a relay node in the system.

In a *replay attack*, the adversary receives and stores a beacon sent by a member of the stream and tries to replay it at a later point of time with malicious intent. The replayed beacon contains old information which can lead to hazardous effects. For instance, consider a scenario where a CACC vehicle stream is moving forward with speed of 30 m/s (108 km/h). The adversary captures a beacon, and stores it for later use. When the leading vehicle slows down, the adversary injects the old beacon into the system periodically. Following vehicles still think that the leading vehicle is driving with 30 m/s and they will not slow down, potentially leading to a collision.

State-of-the-art security architectures employing a strong cryptographic system have the potential to effectively thwart application layer attacks in the case where the adversary is an untrusted outsider. Digital signatures provide *data integrity* for beacon messages, and protect them from unauthorized change. In addition to data integrity, digital signatures also provide *authentication* (both peer entity, and data-origin authentication), as well as *non-repudiation* (with the help of a trusted third party) services. Moreover, using *nonce* in the messages which

is an arbitrary number (chosen in a pseudo-random process) used only once in communication, is a technique to prevent replay attacks.

Although the above methods are well known, there are practical challenges involved in deployment, implementation and standardization of such security architectures in VANET. This is due to the scale of the proposed VANET network and varying interpretations of what constitutes "security and privacy" in various areas of the world [3]. Furthermore, in the case where the adversary is a trusted insider such as a compromised vehicle with a valid certificate, the problem is much harder to solve. Typical approaches to handling this issue is via misbehavior/anomaly detection techniques such as [7] which require multiple sources of data which may not always be available. In addition, such techniques do not guarantee perfect detection in all circumstances and are usually associated with finite false negative and false positive rates. There is much left to be done in the anomaly detection field to ensure acceptable performance of these algorithms to ensure the safety of passengers in the fully autonomous driving scenario.

B. Network Layer Attacks

Unlike application layer attacks, network layer attacks have the potential to affect the functioning of multiple user applications. For instance, the adversary can attempt a Denial-of-Service (DoS) or Distributed Denial-of-Service (DDoS) attack to overwhelm the communication capability of a vehicle or a group of vehicles, and make them unable to participate properly in a CACC vehicle stream.

A known method to realize DoS in the VANET scenario is by using a vehicular botnet. Mevlut et al. [8] demonstrate the problems vehicular botnets introduce to the autonomous car setting via a simulation study. In their work, they focus on producing physical congestion on a given road segment and do not discuss the effects of network level congestion through botnets of compromised cars. It is envisioned that autonomous cars are equipped with a tamper-proof *Hardware Security Module (HSM)* which is responsible for storing digital keys as well as performing all cryptographic operations, such as message signing/verification, encryption, and hashing [9]. These cryptographic operations are complex and CPU-intensive, hence, there is an upper-bound in the number of cryptographic operations a HSM can perform at a time. A DoS/DDoS attack can target this limitation to overwhelm an autonomous vehicle and make its HSM unavailable.

Radio jamming to deliberately disrupt communications over small or wide geographic areas as depicted in Fig. 2c is another possible network layer DoS attack. IEEE 802.11p standard uses one *control channel (CCH)* with multiple *service channels (SCH)*. The adversary can use different jamming techniques like one-channel jamming or swiping between all channels and trying to jam them all. We would need a system such as [10] to help detect and mitigate such attacks. Other countermeasures for DoS attack in CACC vehicle stream involve traditional solutions such as channel switching, technology switching, frequency hopping, or utilizing multiple radio transceivers. If none of these techniques are feasible, a CACC vehicle may need to downgrade to the ACC system to help avoid a rear-end collision.

C. System Level Attacks

All presented attacks so far were centered around exploiting V2V communication in a CACC vehicle stream. Another type of attack is tampering with vehicle hardware or software that can be done by a malicious insider in the manufacturing level or by an outsider in an unattended vehicle (for instance by replacing or altering certain vehicle sensors). Even if the communication channel is secure, and a state-of-the-art security architecture is deployed in VANET, if the on-board hardware/software are tampered with or faulty, then the input information to the system will not be accurate. This will affect the operation of the high-level protocols as illustrated in Fig. 2d. Hence, the risk of tampering should not be neglected.

One possible solution is to use tamper-proof sensors. If a tamper-proof version is not available or too expensive to be deployed in large scale, then misbehavior detection techniques discussed in Section V can be useful. It is worth mentioning that in general, the term "tamper-proof" hardware means 100% secure against tampering which is mostly used by marketing specialists and does not exist in practice. There are different technologies to make a device less vulnerable against tampering, and *tamper resistance*, *tamper evidence*, *tamper detection*, and *tamper response* are more accurate terms used to characterize such technologies [11], but these are out of scope of this paper.

D. Privacy Leakage Attacks

CACC vehicles periodically broadcast beacons that contains various information such as the vehicle identity, current vehicle position, speed, acceleration, etc. The availability of this information can comprise the privacy. The adversary can carry out an *eavesdropping attack* as shown in Fig. 2b to extract valuable information about the vehicle stream such as their trace by linking position data, and use them for her own benefit. The presence of signatures in the beacon messages can worsen the situation, and allow the adversary to easily identify the participating vehicles in the CACC stream.

Eavesdropping is a type of passive attack, and hence difficult to detect, especially in broadcast wireless communication. However, it is possible to prevent the success of eavesdropping by using encryption to achieve data privacy or using anonymity techniques to achieve identity and location privacy. Anonymity is typically implemented using *group signatures* [12] or *short-term certificates (pseudonyms)* [13]. Many privacy-preserving security architectures in VANET are using pseudonym-based schemes to keep the information private, but their applicability in a platoon requires more detailed investigation, and is out of the scope of this paper.

IV. Simulation Study

A. Simulation Setting

In order to study some of the discussed attacks on a CACC vehicle system, we utilize the VENTOS (Vehicular Network Open Simulator) platform[3]. VENTOS is an integrated simulator, and is made up of many different modules, including Simulation of Urban Mobility (SUMO) and OMNET++/Veins. SUMO[4] is adopted as our traffic simulator and while OMNET++[5] is used to simulate the wireless communication. Our ACC and CACC car-following models are implemented to replace the default car-following model in SUMO. Moreover, Traffic Control Interface (TraCI) which is responsible for data/command exchange between SUMO and OMNET++ is extended with new set of commands to gain necessary control over parameters exchange for ACC/CACC vehicles. Detailed packet-level simulation is performed in OMNET++, and IEEE 802.11p protocol, the standard protocol adopted for V2V communication in Veins (Vehicles in Network Simulation) framework[6], is used for wireless communication between CACC vehicles.

VENTOS allows us to study situations where the driverless vehicles utilize V2V communication to achieve CACC, and analyze the local and string stability of the system under different speed profiles. Stability is a critical requirement for both ACC and CACC control system design and is achieved by dampening traffic flow disturbances [14]. *Local stability* concerns with one vehicle following a preceding vehicle. A system is said to be local stable if the magnitude of disturbance decreases with time. *String stability*, also referred to as *platoon stability* and *asymptotic stability*, concerns with the propagation of disturbance in a string of vehicles. String stable means disturbance damps out when propagating to upstream vehicles.

We present simulation results of performing application and network layer security attacks with a malicious insider adversary. The insider adversary is on the side of the road with fixed position and is equipped with a radio to communicate with other vehicles in the network. In the application layer attack, we consider message falsification that the adversary modifies the CACC beacon messages in order to affect the stream. In the network layer attack, we consider radio jamming that all wireless communications are disrupted.

B. Results

Fig. 3 shows the speed profile of ten autonomous/driveless CACC vehicles that are moving on a straight single-lane highway when no insider adversary is present. The first vehicle (platoon leader) is referred to as 'Veh 1' and its speed profile is given and shown with a dashed blue line. As Veh 1 accelerates and decelerates, the disturbance propagates to the following vehicles. At $t = 120\,\text{s}$, Veh 1 accelerates at $1.5\,\text{m/s}^2$ to a cruise speed of $20\,\text{m/s}$. Then it decelerates to speed of $5\,\text{m/s}$ with deceleration of $-2\,\text{m/s}^2$. The other nine vehicles follow with uniform time gap setting. As can be seen, the system is stable, and vehicles follow each other smoothly.

Fig. 4 presents the speed profile of the same vehicle stream in the presence of a malicious insider vehicle that is using message falsification attack in order to compromise the system. At $t = 147\,\text{s}$, CACC vehicle stream enters into the radio range of the adversary and gets affected by it. The adversary tries to manipulate the acceleration field of beacon messages to a fix value of $6\,\text{m/s}^2$. Due to the fact that no security features are implemented in vehicles, they accept the falsified beacons and use them for longitudinal control which leads to string instability in the stream, and this disturbance magnifies through the stream. Here, falsification attack is most effective when a sudden change in acceleration occurs.

We note that since the attacker in this case is an insider, cryptographic security in terms of digital signatures/certificates by itself will not be able to prevent the attack. We will need more elaborate misbehavior detection measures as mentioned earlier to help mitigate this problem. In essence, the speed profile of the vehicle stream will look similar to the one in Fig.4 even in the presence of security measures such as digital signatures/certificates.

Fig. 5 illustrates the space-gap between vehicles before and after of radio jamming attack. Before radio jamming, space-gap is $16\,\text{m}$ when the CACC vehicle stream is traveling with speed of $20\,\text{m/s}$ (mark 1), and $5.5\,\text{m}$ in speed of $5\,\text{m/s}$ (mark 2). The insider adversary disrupts all communications in CCH starting from $t = 308\,\text{s}$ (mark 3). Failure to receive beacons, CACC vehicles downgrade to ACC mode with larger time-gap and delay settings. As a result, the space-gap is increased to $26\,\text{m}$ (mark 4) and reaction of followers to speed changes becomes relatively slower (mark 5). Downgrading to ACC is a simple countermeasure that diminishes the impact of radio jamming attack from a rear-end collision to reduction in CACC performance. We leave designs of more elaborate countermeasures to the insider jamming attack for future work.

V. Potential Countermeasures for Detecting Malicious Behavior

In this section, we explore some possibilities to help secure CACC vehicle streams by detecting the compromised or faulty sensor/vehicle. These could have implications for vehicles with varying levels of automation, from WAVE-enabled vehicles with ACC/CACC capabilities to fully automated driver-less vehicles.

A. Local Plausibility Check

A simple approach in detecting a faulty sensor is to check whether the incoming information is plausible or not [15]. For instance, if a sensor is not reading within its normal range, then the sensor may be faulty or tampered with. The incorrect information can be either discarded or interpolated from the past correct information. Another possibility is deriving the information from other relevant sensors. For instance, if the

[3]http://rubinet.ece.ucdavis.edu/projects/ventos

[4]Simulation of Urban MObility (SUMO), http://goo.gl/Al4w07

[5]OMNet++ Network Simulation, http://www.omnetpp.org/

[6]Veins, http://veins.car2x.org/

wheel speed sensor is compromised and faulty, then the velocity can be derived from the engine speed sensor.

B. Wearables and Mobile Devices

Wearable devices such as the Google Glass and mobile devices such as smartphones and tablets carry a wide array of sensors such as cameras, accelerometer, and GPS along with wireless communication capability. These devices are carried by the driver or passengers of a vehicle. This opens up rich opportunity for developing applications which can potentially improve the security and safety of the system. For instance, the wearable/mobile device of the driver or passenger can act as a verifier for the sensing data generated or received by the vehicle. The wearable device can construct a "belief" from its sensor data about the position of the vehicle, velocity, or acceleration, and cross check this with the belief computed by the vehicle. If there is a discrepancy in the beliefs as seen from the wearable device and the vehicle, then it might be an indicator of a security compromise of the hardware or software in the car, or in the communication channel. If the passenger has multiple devices, it might be possible to fuse the sensor information from these devices to construct a more well formed belief which can then be checked against the vehicle's belief.

C. Voting

The two approaches described above are based on local detection of misbehaving sensor/software or compromised communication. If this fails, collaborative decision-making techniques such as *voting* could be carried out that enable vehicles to collectively shield themselves against the misbehaving vehicle. Voting is most effective in scenarios where there are multiple vehicles in a group that are coordinating with one another. A group of vehicles can be defined as nearby vehicles driving within a geographic region, or members of a vehicle platoon. Vehicles in a group keep track of each other's behavior and check for anomalies in the data received from the members of the group and possibly other vehicles on the road. The vehicles then perform a trust computation and then vote for/against keeping the vehicle in the group. This process needs to be done at regular intervals and therefore incurs communication overheads. More detailed study is needed to understand the trade-offs between performance and cost.

VI. Conclusion

Mainstream car manufacturers such as Tesla, BMW, and Audi are all working on commercial versions of vehicles with varying levels of automation embedded in them. With the increased reliance on sensors and computer software to take driving decisions for passengers, addressing security issues will become very important with time. This work is an attempt towards gaining a better understanding of the security risks involved in connected vehicle streams, from WAVE-enabled vehicles with ACC/CACC capabilities to fully automated driver-less vehicles. We analyze different security attacks on a vehicle stream and discuss the possible security design decisions which will need to be taken to ensure safety of the system. The impact of the security attack such as rear-end collision or instability of the stream is shown with the help of simulation where we implement the ACC/CACC longitudinal control and perform a detailed packet-level simulation.

References

[1] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 163–168.

[2] M. Amoozadeh, H. Deng, C.-N. Chuah, H. M. Zhang, and D. Ghosal, "Platoon Management with Cooperative Adaptive Cruise Control Enabled by VANET," *Vehicular Communications Journal*, 2015.

[3] R. G. Engoulou, M. Bellaïche, S. Pierre, and A. Quintero, "Vanet security surveys," *Computer Communications*, vol. 44, pp. 1–13, 2014.

[4] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn, "A security credential management system for v2v communications," in *Vehicular Networking Conference (VNC), 2013 IEEE*. IEEE, 2013, pp. 1–8.

[5] J. Blum and A. Eskandarian, "The threat of intelligent collisions," *IT professional*, vol. 6, no. 1, pp. 24–29, 2004.

[6] G. Guette and C. Bryce, "Using tpms to secure vehicular ad-hoc networks (vanets)," in *Information Security Theory and Practices. Smart Devices, Convergence and Next Generation Networks*. Springer, 2008, pp. 106–116.

[7] T. H.-J. Kim, A. Studer, R. Dubey, X. Zhang, A. Perrig, F. Bai, B. Bellur, and A. Iyer, "Vanet alert endorsement using multi-source filters," in *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*. ACM, 2010, pp. 51–60.

[8] M. T. Garip, M. E. Gursoy, P. Reiher, and M. Gerla, "Congestion attacks to autonomous cars using vehicular botnets," 2015.

[9] M. Wolf and T. Gendrullis, "Design, implementation, and evaluation of a vehicular hardware security module," in *Information Security and Cryptology-ICISC 2011*. Springer, 2012, pp. 302–318.

[10] A. Hamieh, J. Ben-Othman, and L. Mokdad, "Detection of radio interference attacks in vanet," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*. IEEE, 2009, pp. 1–5.

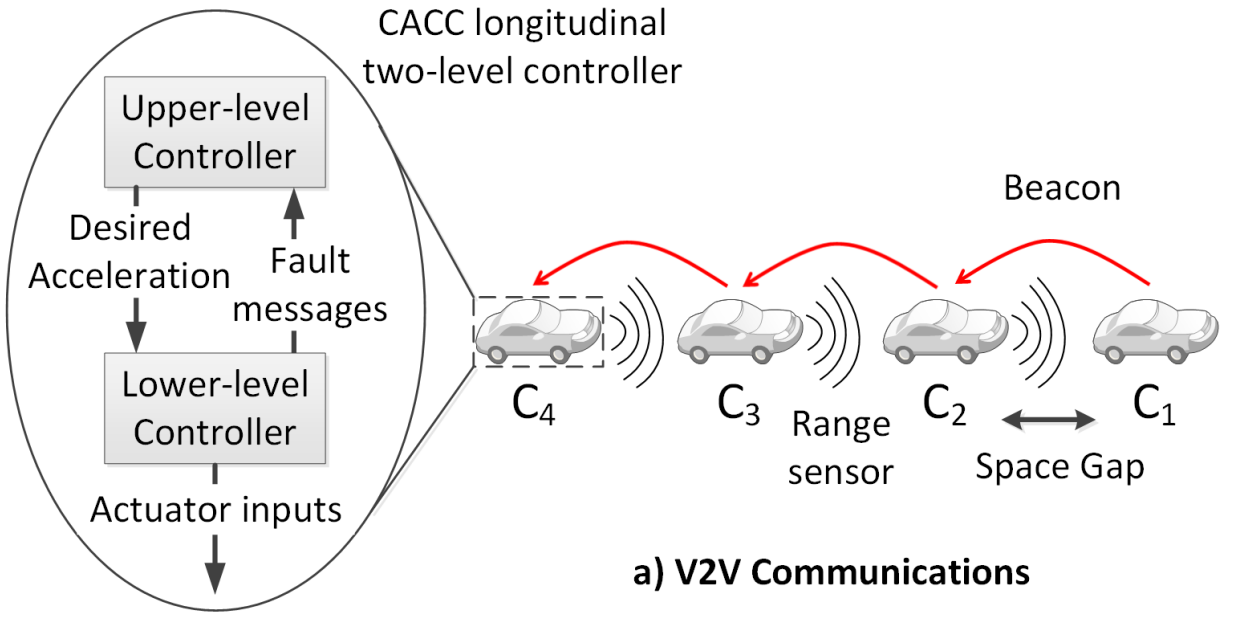
[11] B. Rosenberg, *Chapter 10: Hardware Security Modules, Handbook of financial cryptography and security*. CRC Press, 2010.

[12] X. Lin, X. Sun, P.-H. Ho, and X. Shen, "GSIS: a secure and privacy-preserving protocol for vehicular communications," *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 6, pp. 3442–3456, 2007.

[13] P. Papadimitratos, L. Buttyan, J.-P. Hubaux, F. Kargl, A. Kung, and M. Raya, "Architecture for secure and private vehicular communications," in *Telecommunications, 2007. ITST'07. 7th International Conference on ITS*. IEEE, 2007, pp. 1–6.

[14] R. Pueboobpaphan and B. van Arem, "Driver and vehicle characteristics and platoon and traffic flow stability," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2189, no. 1, pp. 89–97, 2010.

[15] P. Golle, D. Greene, and J. Staddon, "Detecting and correcting malicious data in vanets," in *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*. ACM, 2004, pp. 29–37.



WSMP Header	Sender Address	Receiver Address	Position	Acceleration	Max Decel	Lane ID

b) Beacon Format

Fig. 1. One-vehicle look-ahead communication in CACC vehicle stream. ith vehicle receives information embedded in beacon messages from $(i-1)$th vehicle using V2V wireless communication, and feeds it into the longitudinal control system to maintaining a safe gap from the preceding vehicle. Longitudinal control system in each vehicle is typically designed as a hierarchical two-level controller [2]

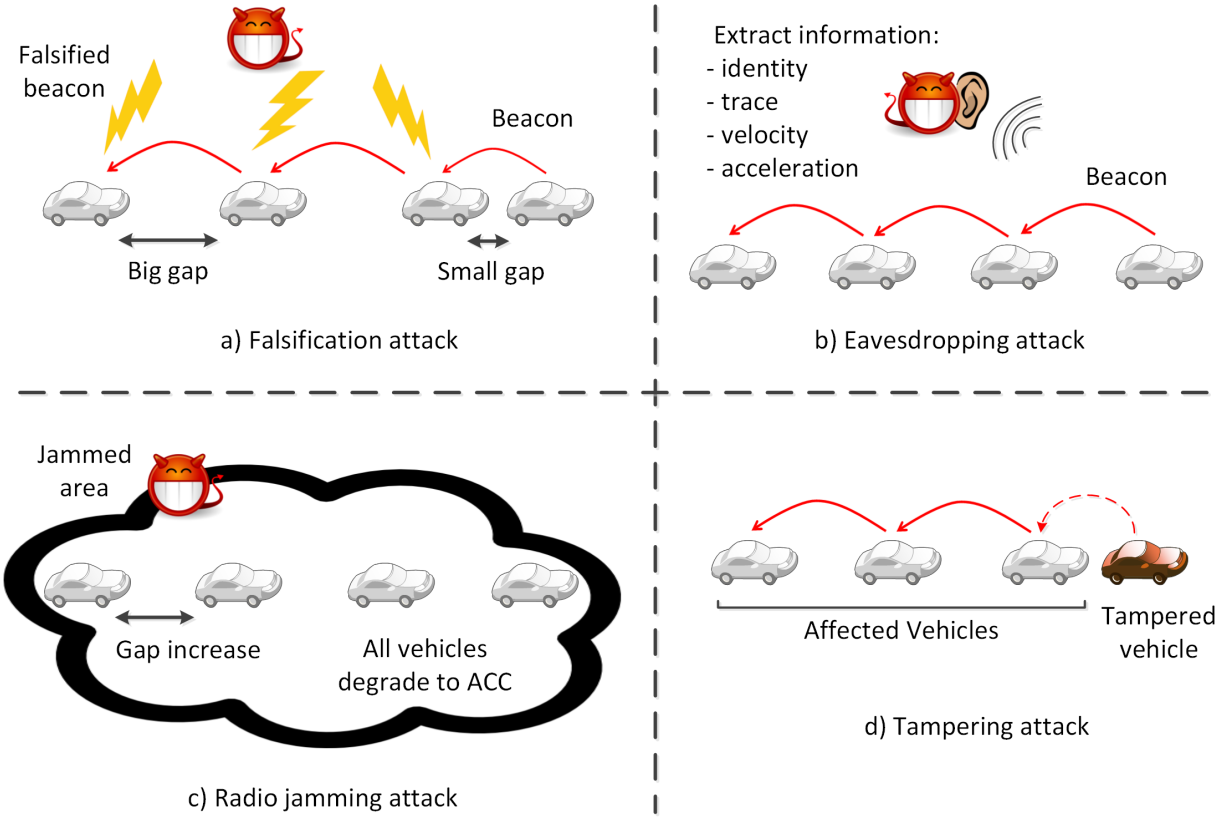


Fig. 2. Security attacks on a CACC vehicle stream.

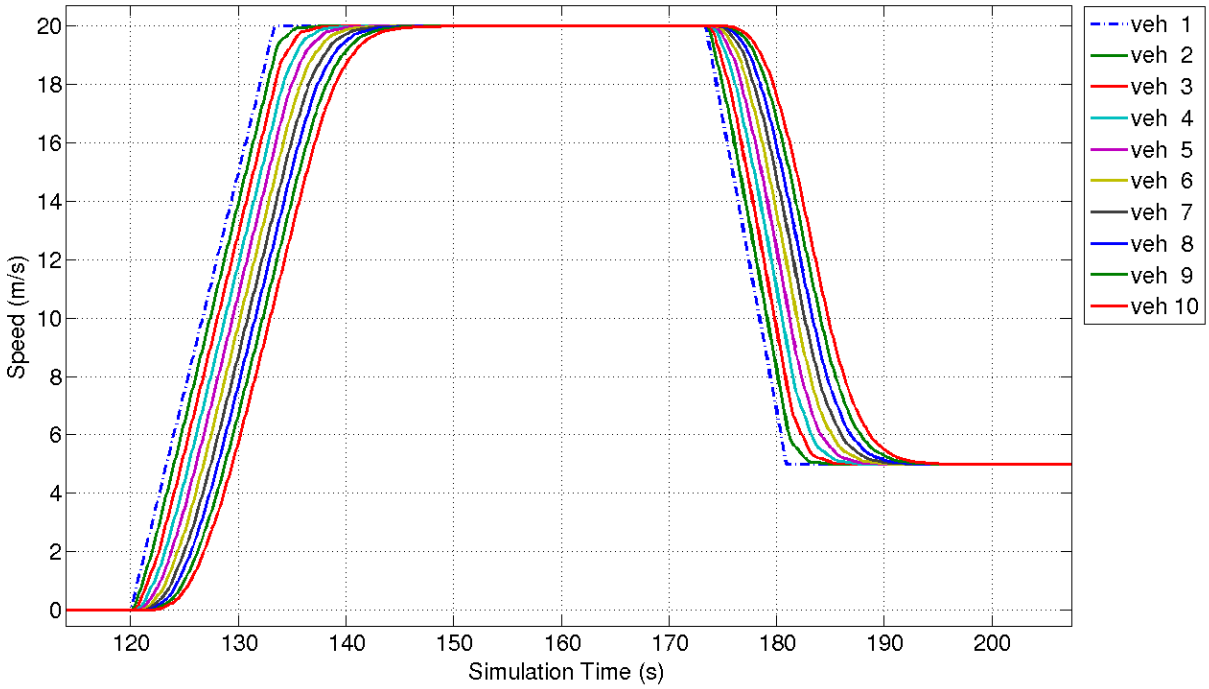


Fig. 3. Speed profile of CACC vehicle stream with no adversary. The system is stable and as 'Veh 1' speeds up and slows down, all vehicles follow each other smoothly.

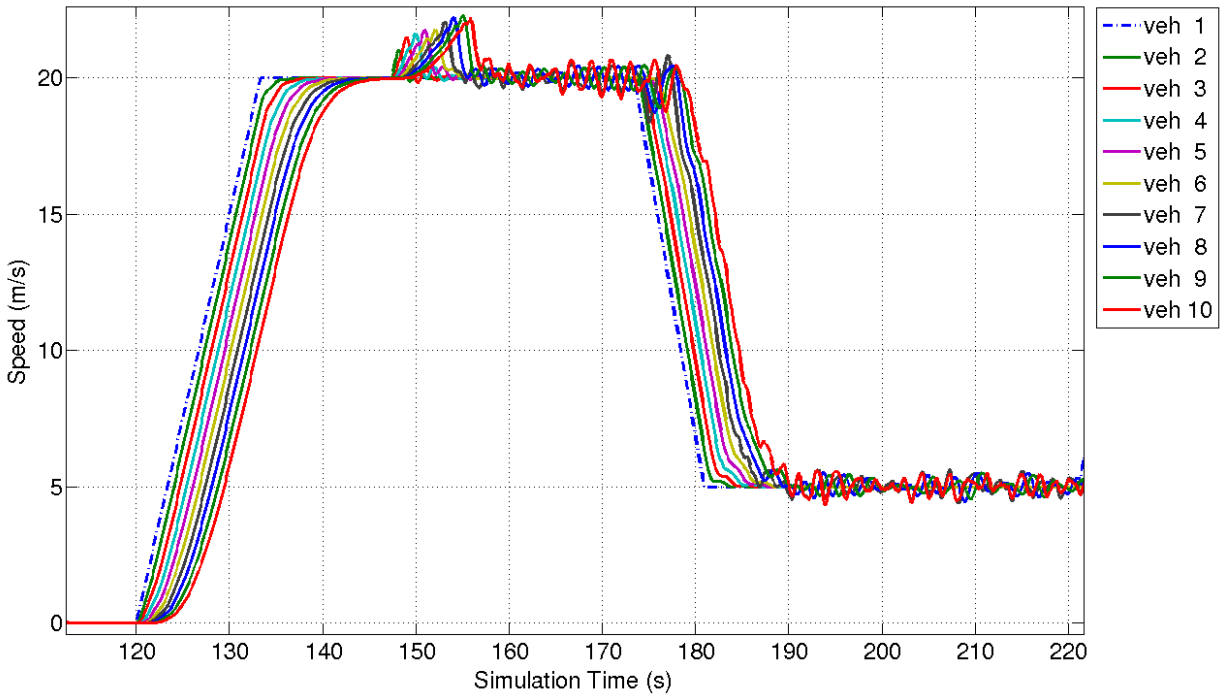


Fig. 4. Effect of message falsification attack on the CACC vehicle stream. String stability is not maintained and the disturbance magnifies through the stream.

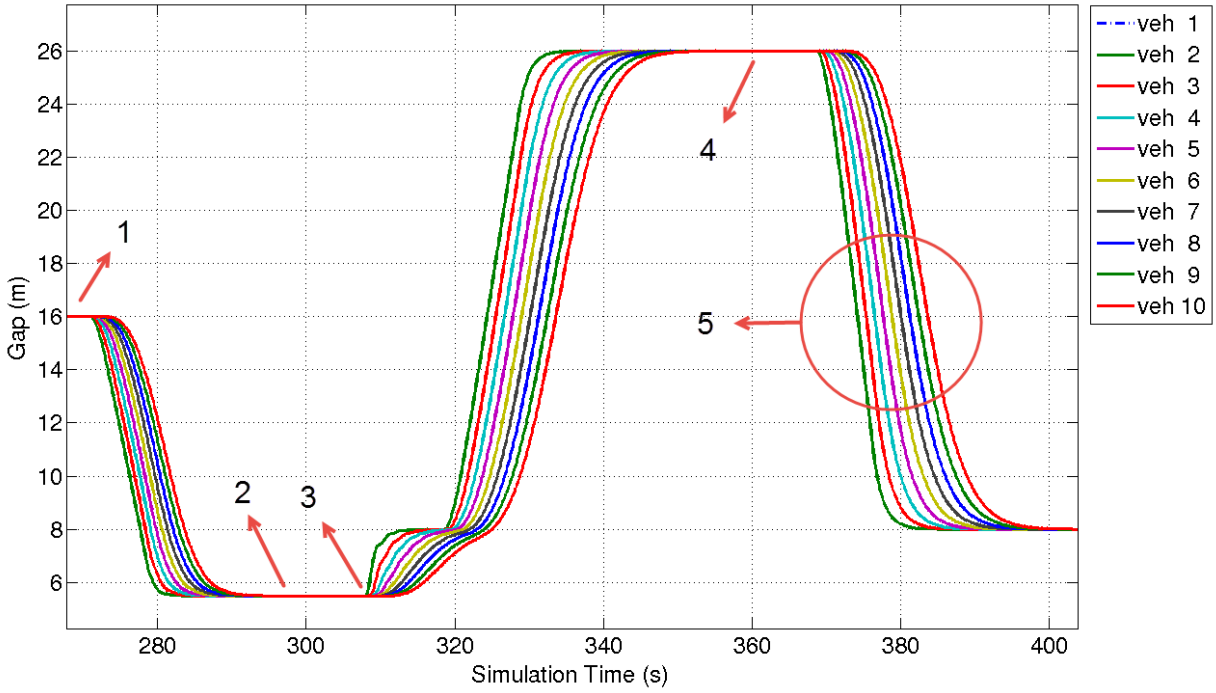


Fig. 5. Effect of radio jamming attack on the CACC vehicle stream. The CACC controller detects communication loss at $t = 308\,\mathrm{s}$ and downgrades to ACC mode.