CommPact: Evaluating the Feasibility of Autonomous Vehicle Contracts

Jeremy Erickson
University of Michigan
jericks@umich.edu

Shibo Chen
University of Michigan
chshibo@umich.edu

Melisa Savich
University of Michigan
msavich@umich.edu

Shengtuo Hu
University of Michigan
shengtuo@umich.edu

Z. Morley Mao
University of Michigan
zmao@umich.edu

ABSTRACT

In Autonomous Vehicle (AV) platooning, vehicles queue up with minimal following distances for improved traffic density and fuel economy. If one vehicle is compromised and suddenly brakes, these AVs will most likely be unable to prevent a collision. In this work, we propose a proactive approach to platooning security: *Autonomous Vehicle contracts*, in which AVs are architected to use secure enclaves to enforce agreed-upon driving rules, such as a restriction not to brake harder than a certain threshold while the contract is in effect. We explore whether AV contracts will be feasible in worst-case emergency situations while simultaneously under attack, when it is imperative to return full autonomy to AVs as soon as possible. Through our prototype contract implementation using Intel SGX enclaves, including measurement from real-world testing of wireless On-Board Units (OBUs), we show that AV contracts can be quickly and safely terminated in the event of an emergency while retaining a false positive rate of under 0.001% per 10 hours of use. We find that individual autonomy can be returned to the vehicles of an 8-vehicle platoon under contract within 1.5 seconds of an attack, including both detection and safe vehicle separation. Smaller platoons are even quicker. Consequently, automobile manufacturers may find the additional safety offered by AV contracts to provide a net benefit.

1 INTRODUCTION

We are currently on the precipice of Autonomous Vehicles (AVs) becoming a reality for the general public. This technology promises improvements to safety, reliability, efficiency, and quality of life. Vehicle-to-Vehicle (V2V) and Vehicle-to-Roadside (V2R) communications, collectively called V2X, are one area of ongoing research in which vehicles communicate with entities around them to better plan their driving paths. V2X can be used to deliver critical safety and positional information to nearby vehicles, as well as help shape traffic patterns on a regional scale. One exciting promise of Connected AVs (CAVs) is Cooperative Adaptive Cruise Control (CACC), or *platooning*, in which several vehicles trail one after another with minimal following distance, behaving as a single unit.

Platooning brings many advantages, particularly for highway driving. Vehicles following closely behind one another can reduce their wind resistance, significantly increasing their fuel economy. This is particularly important for vehicles that are on the road for long periods of time with minimal speed changes, such as trucks, for which testing has shown a 5-10% decrease in fuel usage, including for the leading vehicle [31]. Additionally, by packing vehicles tighter together, traffic congestion can be reduced. Clearly platooning is a desirable construct, generating at least one startup so far [36], and the literature is filled with work exploring the necessary communication between vehicles to make it a reality [14, 24, 44, 46, 50].

Just as humans are taught to drive safely, AVs are taught to identify and avoid dangerous situations, e.g., too little following distance with a preceding vehicle. Some AV models explicitly place safety constraints on the vehicle operation [5], within which the AV can freely navigate. For instance, an AV should retain a safe following distance between itself and any preceding vehicle so that it has time to react to any unexpected behavior. However, in the process of joining a platoon, an AV will find itself compelled to violate this safe following distance constraint, as it must maintain a close following distance for economic benefit. This puts the vehicle occupants at risk that the preceding vehicle could suddenly brake and cause a collision, potentially one involving many vehicles with limited ability to respond. Liu has demonstrated this issue [35] using the PLEXE [47] platooning extension for the popular AV simulator, Veins [49]. Researchers have demonstrated that existing vehicles can be exploited to remotely control the operation of the brakes, even over the network [10, 30, 38]. Given the history of malware in general, it would be foolish to expect future autonomous vehicles to be immune to compromise.

Many researchers have explored the idea of misbehavior detection for autonomous vehicles [7, 12, 13], but these approaches are inherently reactive. A compromised vehicle must first exhibit unexpected behavior before other vehicles can detect an anomaly and respond to the situation. In a platoon, with many vehicles packed closely together, a malicious vehicle could potentially create a devastating pile-up by braking more quickly than following vehicles can react. Based on published third-party stopping distance test results, modern passenger vehicles exhibit varying maximum braking decelerations ranging from approximately 8 m/s^2 to 11 m/s^2 on dry pavement. We show a small selection of these in Table 1. Even with instantaneous detection and response, two vehicles traveling at 100 km/h (62 mph) with the preceding vehicle braking at 1 G and the trailing vehicle braking at 0.9 G will require 4.32 meters of following distance to avoid a collision. This necessary separation increases linearly by another car length (5 meters) for every 173 ms of delayed response by the trailing vehicle. Although reactive measures may be able to reduce the damage and risk of injury from such a collision, they are clearly insufficient to prevent contact at desirable following distances of 1-3 meters.

Model	Category	Weight (lbs | kg)	Deceleration (m/s^2)
2017 Koenigsegg Agera RS	Super	3000 | 1360	11.62 to 12.85
2015 Ford Mustang GT	Sport	3805 | 1726	10.93
2016 Mazda MX-5 (Miata) Club	Sport	2332 | 1058	10.44
2016 Honda Civic Sedan (Touring)	Compact	2923 | 1326	10.09
2016 Honda Civic Sedan (EX)	Compact	2790 | 1266	9.29
2015 Ford F-150	Truck	5160 | 2341	9.15
2017 Toyota Sienna Limited	Minivan	4560 | 2068	8.87
2016 Ford F-150	Truck	4629 | 2100	7.93

Table 1: Mean maximum decelerations calculated from stopping distance tests on dry pavement.

Instead, we propose a proactive solution: autonomous vehicle *contracts*. A contract is an agreement between autonomous vehicles not to violate certain driving parameters. For instance, two or more vehicles traveling close together in the same lane may form a contract ensuring both maintain a speed of 100 km/h with an allowed deviation of 2 km/h until the contract is either amended, ended, or safely terminated upon communication failure. Contracts are enforced by an *enclave*, a verifiable binary running on each vehicle that can be remotely attested through cryptographic hardware keys. The AV is architected such that driving commands must be signed by the enclave, which ensures the vehicle will not violate the agreed-upon parameters. Each vehicle's powertrain and braking Electronic Control Units (ECUs), the computers that control actuation of the gas and brakes, are restricted from taking actions not validated and signed by the enclave. By remotely attesting that platoon vehicles conform to this model and signing the platoon's contract before joining, each vehicle gains additional safety assurance that the other vehicles in the platoon, even compromised ones, are restricted from performing actions disallowed by the contract, such as suddenly braking.

Of course, contracts also have a significant downside: that vehicles restricted from making sudden velocity adjustments may consequently be restricted from responding to emergency situations, such as an obstacle appearing in the road. We cannot allow vehicles to unilaterally void a contract, as doing so would obviate this new protection against bad actors. To make matters worse, the wireless communication channels can be jammed by an adversary, even one not part of the platoon. In these worst-case scenarios, it may not be possible for the platoon vehicles to actively coordinate an emergency response. Therefore, we need a proscribed mechanism for voiding a contract safely in the presence of interference and need to minimize delay before returning autonomy to each vehicle. Human-driven vehicles are subject to the Perception Response Time (PRT) which indicates that it generally takes humans between 1 and 3 seconds to respond to unexpected circumstances [18, 26, 40]. If we can safely separate and return autonomy to platooning vehicles in a similar time, it may be desirable to use contracts to increase the safety factor for trailing vehicles when platooning. Our experimental results show that it should be possible to do so in under 1.5 seconds for platoon sizes up to 8.

In this work, we explore the problem space of returning autonomy to vehicles under contract as quickly as possible and how the various trade-offs affect the feasibility of introducing contracts to autonomous vehicle platooning. We present the following contributions:

A prototype framework and messaging protocol that supports the creation and maintenance of contracts as vehicles join and leave the platoon and maintains platoon safety in the presence of adversaries.
An analysis of the physical process of separating platooning vehicles to safe following distances and velocities with minimal delay.
An analysis of the trade-off space for minimizing delay in detecting communications failure while maintaining connectivity in unreliable conditions.
A simulated implementation that demonstrates contracts in action.[1]
Measurement of the wireless latencies and cryptographic computational delays that influence the critical path for detecting communications failure, as well as comparison to the current state-of-the-art.
A discussion of additional factors that affect the contract model, such as how contracts can continue to provide safety assurance in the presence of redundant braking systems.

2 THREAT MODEL

In this work, we assume the perspective of a vehicle occupant of an autonomous vehicle forming a platoon with other autonomous vehicles. In our model, all platooning vehicles are fully autonomous. The most important assumption is that an occupant *must* trust their own vehicle. A malicious vehicle can harm its occupants, e.g., drive itself off a cliff, entirely outside of the scope of platooning or contracts. It follows then that benign vehicles will act in their occupants' best interest, e.g., will avoid collisions with other vehicles to the best of their ability and not agree to unsafe contracts.

We assume that any or all *other* vehicles in the platoon can be compromised or malicious, and that the operating system and applications can be controlled arbitrarily by an adversary. Malicious vehicles can collude, may maneuver in any way within their control, and send arbitrary network traffic, including completely jamming the communication channel. We assume that in the absence of an attack, platooning vehicles can coordinate to navigate around obstacles or slow down as the situation merits, and so our work focuses on the worst-case scenario in which all communications are jammed or another Denial of Service (DOS) attack is present.

[1] https://github.com/jericks-umich/commpact

We assume benign vehicles will react to anomalous behavior by separating, avoiding, and blacklisting any non-compliant vehicle, generally outside the scope of this work. In comparison, this work is primarily focused on proactively preventing sudden, stealthy attacks that other approaches cannot address. Our protocol is designed so that any attack on either the integrity or the availability of the systems and messages will devolve into a DOS, which we have designed the protocol explicitly to accommodate.

We assume each vehicle is equipped with an enclave that provides attestation capabilities. If vehicles are unable to attest one another's enclaves, or if conditions are adverse, they will not form a platoon. Our architecture assumes that the powertrain and braking ECUs for each vehicle, during manufacturing, have been paired with the enclave and exchanged keys. This allows the enclave and ECUs to validate messages from each other, and prevents another device from impersonating them, regardless of communication medium. We assume that the powertrain and braking ECUs are the final arbiters of the car's movement. That is, physical attacks between an ECU and its actuators or that otherwise modify the movement of the car are outside the scope of our model. Our model is built on the assumption that the enclave and ECUs will behave as designed. As such, we assume that reducing the ECUs' attack surface to only accept commands from the enclave and remotely attesting the integrity of the enclave binary is sufficient to protect them from adversarial control.

Finally, we assume that the clocks of the enclaves and ECUs are synchronized across vehicles in the platoon. Although this paper assumes all clocks are perfectly synchronized, a real system should aim to ensure that its clocks are synchronized within approximately 1 ms. This should have only minimal effect on the equations presented later. Clock synchronization is a difficult problem in general, and even more so in an adversarial scenario. We leave it to future work. However, if the platoon vehicles' clocks cannot be synchronized, benign vehicles can still protect their occupants by simply not forming a platoon.

3 RELATED WORK

To date, the research on autonomous vehicle platooning has primarily focused on efficient communications, support for platooning maneuvers, and the string stability problem [14, 46]. The PLEXE [47] extension to the popular Veins [49] simulator has paved the way for researchers to evaluate different platooning strategies to meet these goals [22, 23, 43]. More recently, security researchers have begun to evaluate the safety implications of compromised autonomous vehicles engaging in platooning scenarios. Heijden [52] uses PLEXE to demonstrate how jamming and data injection attacks against several CACC controllers can lead to vehicle collisions. Petrillo [42] proposes a new CACC controller that is resilient to adversarial attacks and uses PLEXE to evaluate it against spoofing, message falsification, and packet loss. Anomaly and misbehavior detection techniques have also been explored [7, 12, 13]. However, as Liu [35] shows, a malicious vehicle with control over the vehicle's motion can cause a platoon collision directly. Even an instantaneous braking response cannot prevent collisions among heterogeneous vehicles. Reactive approaches to platoon safety are important, but insufficient.

Enclaves have typically been used in cloud and mobile applications. One of the earliest use cases for enclaves was enterprise-use of personal mobile devices, also known as Bring-Your-Own-Device (BYOD). Enterprises wish to allow users to access proprietary information, such as company email, on their personal smartphones, but often require assurance that the data will remain within company control. One product that delivers this is Samsung Knox [53]. Built on ARM TrustZone [34], Knox keeps company data within a company-controlled enclave on employees' personal devices, allowing administrators to, among other things, remotely wipe the enclave memory. In 2015, with growing concerns of data compromise on public clouds [21], Intel introduced SGX [37], with stronger cryptographic root-of-trust guarantees than previous offerings. SGX was groundbreaking because of its remote attestation capability. After forming a proof of attestation, it could be sent to a remote user and verified, ensuring that the application was not altered and was running on genuine Intel hardware [6]. Applications could be designed to only deploy keys or provision sensitive data after this attestation process was complete. Despite many attacks against the current generation of enclaves [8, 17, 32, 45], the concept is powerful and can enable many secure features once more refined and resilient iterations are available.

Several recent works have explored the use of secure enclaves for automotive applications. NXP has explored the use of enclaves to encrypt and protect sensitive data, similar to the use of a Hardware Security Module [48]. Virtual Open Systems [41] and Kim [28] have explored the use of TrustZone's secure world as a platform for segregating critical vehicle software and In-Vehicle Infotainment software while running both on the same hardware. However, fundamentally, these works follow the same model as a hypervisor restricting sandboxed software from tampering with privileged data while sharing the compute platform. The true promise of enclaves is the ability to run trusted code on *remote* hardware and receive attestation that it will faithfully execute.

To the best of our knowledge, our proposal is the first to leverage the remote attestation capabilities of enclaves to protect vehicle occupants by enforcing driving constraints on *other* autonomous vehicles.

4 BACKGROUND ON ENCLAVES

The technical background of enclaves, upon which AV contract enforcement is built, is largely outside the scope of this paper. In this section, we aim to highlight the important features that enclaves provide, without delving too far into the mechanisms by which they provide these features. Curious readers may refer to the literature on this topic [25, 34, 37] for more information.

Enclaves can be thought of as a hardware-enforced partitioned environment for executing *trusted* code. Untrusted code, which likely includes the main operating system and most applications, runs in the *insecure world*, while trusted applications run in the *secure world*. Memory used by the secure and insecure worlds is disjoint, and execution may only change worlds using prescribed instructions that perform a secure context switch, similar to a syscall or hypercall.

Intel, AMD, and ARM each have their own enclave architectures, Software Guard Extensions (SGX), Secure Encrypted Virtualization

(SEV), and TrustZone, respectively. In our prototype, we use Intel SGX, as its root of trust characteristics are more mature than those of TrustZone and hardware for SGX v1 is available for consumer purchase. Regardless, our model does not rely on the characteristics of a particular enclave technology, but rather on these key features:

> Remote cryptographic attestation of valid hardware
> Remote cryptographic attestation of an immutable, currently-executing enclave binary
> Ability to securely generate and exchange keys derived from a hardware-based random number generator

In this work we require an enclave to run on each vehicle. Since the binary for this enclave can be attested, one vehicle can trust that the enclave running on another vehicle will faithfully execute as expected. Our model expects that all vehicles will be using identical, or at least mutually trusted and compatible, enclave applications, which can be verified as such before a vehicle joins the platoon. Barring vulnerabilities in the enclave binary itself, enclaves can trust other vehicles' enclaves to behave as they would themselves.

It is not critically important on which computing device in the vehicle the enclave exists, although there are advantages for running it from vehicle's gateway controller. The enclave will run a small binary with a simple purpose: to validate the commands being passed to the powertrain and braking ECUs and filter any that conflict with the current contract. The ECUs must be configured to only accept commands that are signed by the enclave which ensures that the commands conform to the agreed-upon contract and reduces the attack surface of the ECUs. The ECUs cannot simply conform to the contract policy themselves without an enclave because without the enclave's ability to be remotely attested, other vehicles cannot trust that the code properly enforces the contract. When joining a platoon, each vehicle attests the enclave running on the other vehicles before exchanging keys. Later, when receiving a contract signed by an attested enclave of another vehicle, one can trust that the remote enclave will enforce the terms of the contract.

5 METHODOLOGY

Since our primary goal is safety, we must consider what happens in emergency situations. Each vehicle will be bound to the platoon contract and restricted in how it may react. This currently takes the form of speed and acceleration bounds, as shown in Table 6. In the common case, with a working communication channel and no adversarial interference, the platoon will be able to coordinate to adjust the contract and react directly to an emergency situation, whether that means coming to a complete stop, performing a lane change, or some other maneuver. Enumerating the many optimal solutions to specific emergency situations in a *cooperative* environment is outside the scope of our work. We focus instead on the *adversarial* environment, in which communications can be severed. In this environment, we wish to safely terminate the contract as soon as possible and return full autonomy to each vehicle, allowing them to respond to the emergency situation as individuals. We cannot simply terminate the contract while the platoon is still formed, as this would effectively negate the safety restrictions the contract places upon other vehicles. Consequently, we must separate the vehicles in the platoon to a Safe Separation Distance and Velocity (SSDV) before terminating the contract. This becomes our primary form of defense against adversarial actions. The platoon contract may be cooperatively terminated by any member vehicle, or by a timeout due to lack of communication, but must always remain in effect until the platoon has safely separated and the Emergency Termination Procedure (ETP) is complete.

Let us imagine a worst-case scenario. At some time t, an emergency occurs. We cannot predict this emergency, and so cannot preload instructions for the platoon vehicles to follow to react to this specific emergency. We also cannot assume that all vehicles in the platoon are even aware that an emergency has occurred. Perhaps the leader has just discovered an obstacle in the road but it is not visible to the following vehicles. Also, at time t, an adversary jams the communication channel.

In this situation, we have two phases before each vehicle can be released from the contract and may regain full autonomy. The first phase is the **Recovery Phase**, in which the platoon attempts to recover from its communications failure. Temporary wireless communications failures are common, and so we must find a balance between robust communications recovery and minimizing the delay before initiating the second phase of the termination procedure. The second phase is the **Separation Phase**, in which the platoon vehicles begin to separate from one another. The Separation Phase ends when the vehicles have reached the SSDV, at which point the contract ends. The sum of these two phases will dictate how long it takes for the platoon vehicles to react to a worst-case emergency situation.

We start by explaining the Separation Phase and ETP, as the separation of the platoon informs the design of the communications protocol for the Recovery Phase.

5.1 Separation Phase

In the event that the Separation Phase is triggered, the platoon vehicles must separate and return to a safe following distance from one another. This procedure must be defined in advance, as termination may commence due to a total disruption of communications and so no orchestration between vehicles can be relied upon during the procedure itself. Additionally, any flexibility given to vehicles to determine the parameters of this procedure, such as individual deceleration rate, must be tightly constrained and enforced by the enclave such that we can still provide assurance that the vehicles will not collide. We cannot assume that the vehicles can separate via lane change, as there may be another vehicle blocking the adjacent lane, or there may be no adjacent lane available. Separation must occur only from the rear of the platoon since we cannot assume the leader has room to speed up. We wish to be able to pre-calculate the amount of time this separation will take so our total delay can stay within a safe threshold every time platoon members are added or environmental conditions change.

Each vehicle will decelerate at a proscribed fraction of the platoon's maximum deceleration rate so as to maintain equal distance between vehicles. The absolute negative acceleration A for each vehicle in the platoon is given by:

$$A_n = \frac{n}{N} M \tag{1}$$

where N is the number of follower vehicles in the platoon, n is the vehicle's index in the platoon starting at 0 for the leader and ending

Platoon Size	v_0 m/s	a_0 m/s^2	a_1 m/s^2	a_2 m/s^2	d_0 m	d_{stop} m	t_{sep} ms
2	27.77	-8.82	-9.81	-8.82	1.0	1.0	158
3	27.77	-4.41	-9.81	-8.82	1.0	1.0	307
4	27.77	-2.94	-9.81	-8.82	1.0	1.0	451
5	27.77	-2.20	-9.81	-8.82	1.0	1.0	594
6	27.77	-1.76	-9.81	-8.82	1.0	1.0	728
7	27.77	-1.47	-9.81	-8.82	1.0	1.0	867
8	27.77	-1.26	-9.81	-8.82	1.0	1.0	982

Table 2: Separation procedure delays (t_{sep}) for separation decelerations (a_0) corresponding to the relative maximum braking rate for platoon sizes of 2 through 8.

Figure 1: Free-body diagrams showing distances, velocities, and accelerations of the Leader and Follower vehicles at the beginning of the separation procedure (t_0), the end of the separation procedure (t_{sep}), and the point at which both vehicles have come to rest (t_{stop}).

at N for the tail vehicle, and M is the minimum of the platoon vehicles' maximum braking decelerations.

5.1.1 Calculating Separation Phase Delay. During the termination procedure, we wish for the platoon vehicles to separate as quickly as possible. Since all vehicles in the platoon must be able to reach a safe separation distance and velocity without communications, we need to preemptively calculate the time it will take from the start of the termination procedure until the vehicles have sufficiently separated such that all vehicles may decelerate independently and no vehicle will be forced into a collision.

We first define the Safe Separation Distance and Velocity (SSDV) between the leader l and follower f as the distances and velocities such that if both apply their maximum braking acceleration they will not collide before coming to a complete stop. To simplify the scenario for visualization, we focus on a platoon made up of two vehicles, a leader and a follower, shown in Figure 1. At time t_0, both vehicles will be traveling at the platoon velocity v_0 and separated by a distance d_0, the Separation Phase will begin and the follower will decelerate at a_0.[2] At time t_{sep}, the vehicles will reach the SSDV. The leader will still have velocity v_0 but the follower will have decelerated to v_1 and they will be separated by a greater distance d_{sep}. Both vehicles may begin to brake at their maximum rates. At time t_{stop}, the vehicles will have reached zero velocity and should remain separated by some safe distance d_{stop}.

[2]Other pairs of vehicles in a larger platoon will also separate at a_0 *relative to one another*, but will have a slower effective v_0 at t_{sep}

We can solve for t_{sep} with the following equation, derived in Appendix A:

$$\begin{aligned} {}^1a_0^2a_1 \quad a_0a_1a_2{}^0t_{sep}^2 + {}^12a_0a_1v_0{}^0t_{sep} + \\ v_0^2{}^1a_1 \quad a_2{}^0 + 2a_1a_2{}^1d_0 \quad d_{stop}{}^0 = 0 \end{aligned} \tag{2}$$

Since at t_0 we either know or can estimate all terms besides t_{sep}, we can solve Equation 2 with the quadratic formula.

When we move to platoons with more than two vehicles, it becomes simple to extend the model. For an N-vehicle platoon, each vehicle will decelerate at its fraction of the platoon's maximum braking deceleration, as dictated by Equation 1. Thus, at t_{sep}, each pair of vehicles will be separated by the same distance d_{sep} and the same difference in velocity. The only difference between each pair is that, at t_{sep}, later vehicle pairs will have lower absolute velocities, and will therefore reach t_{stop} sooner. We may continue to use Equation 2 as an upper bound for t_{sep}. Table 2 shows the result of using Equation 2 for platoons initially traveling at 100 km/h, with a maximum deceleration rate of 0.9 G, but the potential for vehicles to decelerate at 1.0 G once the SSDV is reached.

5.2 Recovery Phase

Platooning vehicles will spend the majority of their time in a steady state with a contract active. During this time, each vehicle will participate in continuous communication to ensure the communications channel is still up. In the worst case, an unrecoverable communication failure must lead to the emergency termination of the contract and platoon.

LTE [9], 5G[19], and Dedicated Short Range Communications (DSRC) [27] protocols are being considered for intra-platoon communications. In this work, we use DSRC for its low latency and availability of hardware for testing, but any wireless technology with sufficiently-low message latency could be used. We do not rely on any specific characteristics of DSRC.

We assume the wireless spectrum is adversarially-controlled and therefore each vehicle will sign its messages. Since the keys are stored within our trusted enclave, we assume forging valid messages to be infeasible within our protocol's recovery period. We pick ECDSA signatures over the NIST P-256 (secp256r1) elliptic curve as it is both one of two curves supported for message authentication in DSRC [2] and also supported by the SGX enclave cryptographic library. Any attack on message integrity will therefore invalidate the message and be treated as a dropped packet. Thus, we are

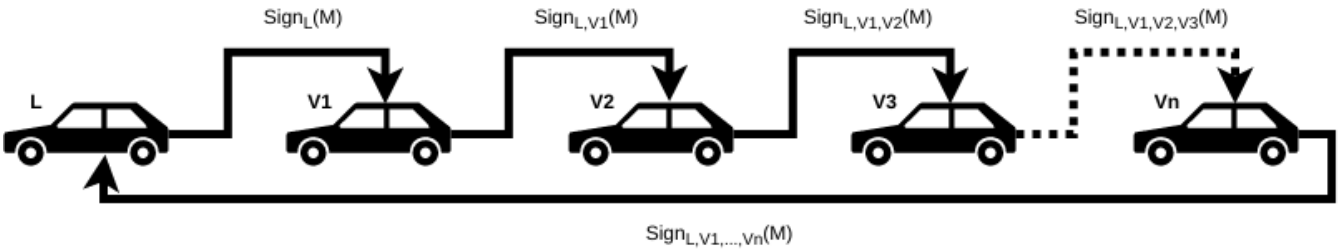

Figure 2: A basic contract chain in which M is the contract message. The Recovery Phase timeout is extended, starting with the leader and progressing down the platoon in order. If the Emergency Termination Procedure is ever triggered, the leading vehicles are guaranteed to end the Recovery Phase no earlier than the trailing vehicles since each prior vehicle must have extended its own Recovery Phase timeout before continuing the chain.

primarily concerned with Denial of Service (DoS) attacks on the wireless medium or systems, as opposed to any attacks on the confidentiality or integrity of the protocol.

We wish to ensure that the Recovery Phase will end and the Separation Phase will begin for all platoon vehicles simultaneously so that Equation 2 holds true. If all platoon vehicles currently plan to end the Recovery Phase at time t_x, and the leader wishes to coordinate an extension of the Recovery Phase until some later time t_y, we believe it is infeasible for each vehicle to be simultaneously assured that all other vehicles have extended the Recovery Phase until time t_y, due to the potential for adversarial tampering. Therefore, we relax our requirement for complete synchronization of the start of the Separation Phase. We require that trailing vehicles must always start the Separation Phase *no later than* the vehicle preceding them. Thus, Equation 2 will still ensure that no two adjacent vehicles will collide. With this modified requirement, we can take advantage of the fact that platoon vehicles are *ordered*. We use a **contract chain** to periodically extend the timeout and ensure that preceding vehicles *must* extend their Recovery Phase before trailing vehicles are able to do so. If at any point, the contract chain fails, vehicles preceding the failure will have an extended Recovery Phase timeout whereas trailing vehicles will continue to end the Recovery Phase at the previously-agreed time. The contract chain is shown in Figure 2.

As an example to help visualize how this works, let's assume that the Recovery Phase has a nominal duration of 500ms. At time t_0, all vehicles in the platoon will begin the Separation Phase at some t_{sep} such that $t_0 < t_{sep} < {}^1t_0 + 500ms^0$. In the absence of any other stimulus, the platoon leader L will extend its Recovery Phase timeout to $t_0 + 500ms$. It will then append its signature to the message, certifying that its Recovery Phase has been extended, and send that to the second vehicle in the platoon, v_1. v_1 will extend its own Recovery Phase timeout to $t_0 + 500ms$, append its signature, and pass it along to v_2. v_2 will be able to safely extend its own Recovery Phase timeout to $t_0 + 500ms$ since it knows that neither L nor v_1 will begin the Separation Phase before that time. Now let's assume that somewhere between v_2 and v_3, a communications failure occurs and v_3 is unable to receive the signatures from L, v_1, and v_2. v_3 will *not* extend its Recovery Phase timeout, and will therefore not be able to sign the message to assure any additional trailing vehicles that its timeout has been extended. Thus, v_3 and any other trailing vehicles will begin the Separation phase at t_{sep}, while L, v_1, and v_2 will begin the Separation phase at $t_0 + 500ms$. A successful contract chain will conclude with the trailing vehicle sending the entire confirmation chain, signed by all vehicles, to the leading vehicle for acknowledgement that the entire platoon is still present. A failed contract chain will be abandoned and after an appropriate timeout period the Leader will attempt a new contract chain. We note that after a contract chain fails, a subsequent successful contract chain will allow any trailing vehicles to update to the latest Recovery Phase timeout.

5.2.1 Recovery from communications failure. Since the wireless medium is known to be unreliable, we wish to be able to tolerate some number of missed contract chains before concluding that the communications channel has been severed. The precise number of failed chains before inducing the ETP is variable and dependent on the network congestion and a safety delay threshold before which the emergency termination procedure should always be triggered. Table 3 shows the probabilistic chance, for various packet loss rates, platoon sizes, and consecutive failed contract chains, for the Separation Phase to be erroneously triggered in the trailing vehicle due to environmental factors over the span of one million contract chains. We note that, even if the Separation Phase is triggered in some number of trailing vehicles, preceding vehicles who successfully extend their Recovery Phase timeout need not begin the ETP.

Ultimately, the number of failed contract chains to attempt before initiating the Separation Phase is dependent on the total latency for one contract chain. A lower latency means more chains can be attempted before reaching the timeout, increasing reliability. Increasing the number of vehicles in the platoon will increase the latency of the contract chain, reducing the number of chains that can be attempted before the timeout. If we wish to keep the total delay for both the Recovery and Separation Phases lower than some bound, the platoon leader will need to carefully monitor the packet loss rate and pick an appropriate number of chains to attempt before the timeout to keep the chance of a false positive below some bound. The leader can also split the platoon to keep the contract chain latency manageable.

5.2.2 Contract Principles. Contract chains need not always traverse the platoon in the same order. For instance, the Join, Leave, and Split procedures (described in section 7) allow contract chains to originate from vehicles besides the leader, traverse the platoon in reverse order, or other behaviors. Because the nuances of the

Packet Loss Rate	Platoon Size	Chance of Termination per 1,000,000 Recovery Chains							
		3 Failed Recoveries		5 Failed Recoveries		8 Failed Recoveries		16 Failed Recoveries	
0.01%	2	7.9972e-6	✓	3.1985e-13	✓	2.5584e-24	✓	6.5470e-54	✓
0.01%	4	6.3943e-5	✗	1.0228e-11	✓	6.5431e-22	✓	4.2829e-49	✓
0.01%	6	2.1568e-4	✗	7.7616e-11	✓	1.6752e-20	✓	2.8081e-46	✓
0.01%	8	5.1092e-4	✗	3.2684e-10	✓	1.6717e-19	✓	2.7968e-44	✓
0.1%	2	0.0079403	✗	3.1856e-8	✓	2.5447e-16	✓	6.4883e-38	✓
0.1%	4	0.061487	✗	1.0123e-6	✓	6.4495e-14	✓	4.1763-33	✓
0.1%	6	0.19193	✗	7.6334e-6	✓	1.6365e-12	✓	2.6942e-30	✓
0.1%	8	0.39505	✗	3.1942e-5	✗	1.6184e-11	✓	2.6402e-28	✓
1%	2	0.99956	✗	0.0030540	✗	2.4104e-8	✓	5.9281e-22	✓
1%	4	1	✗	0.087212	✗	5.5829e-6	✓	3.2447e-17	✓
1%	6	1	✗	0.47594	✗	1.2948e-4	✗	1.7810e-14	✓
1%	8	1	✗	0.92108	✗	0.0011702	✗	1.4857e-12	✓
5%	2	1	✗	0.9965	✗	0.0073431	✗	6.0189e-11	✓
5%	4	1	✗	1	✗	0.68071	✗	1.6001e-6	✓
5%	6	1	✗	1	✗	1	✗	4.3228e-4	✗
5%	8	1	✗	1	✗	1	✗	0.017835	✗

Table 3: Probabilistic chance of False Positive (ETP) due to packet loss over one million contract chains. Chances below 0.001% (1e-5) are shown with a green checkmark. Derived in Appendix B

communications protocol can be complex during these scenarios, we identify several principles which must always be guaranteed by the contract to maintain safety.

Principle 1. A vehicle may extend its Recovery Phase timeout to the minimum value of all vehicles in front of it.

Principle 2. A vehicle may increase its speed bounds only once all vehicles leading it have done so.

Principle 3. A vehicle may decrease its speed bounds only once all vehicles trailing it have done so.

There are two common categories of contract chains.

(1) A standard contract chain that travels sequentially through the platoon from the leader to the rear-most vehicle. Due to Principle 1, this is the only contract chain that can extend the Recovery Phase timeout.

(2) A contract chain that is ordered so as to *ask permission* from every vehicle that may be affected by the proposed operation. For instance, while splitting a platoon, the splitting vehicles must decelerate. Before they can do so, due to Principle 3, each must receive contractual permission to decelerate from all trailing vehicles before doing so itself. Thus, the contract chain must traverse the platoon in reverse order starting from the rear-most vehicle. This kind of contract chain *cannot* extend the Recovery Phase, since doing so would be a violation of Principle 1.

Further discussion of supported platooning scenarios can be found in Section 7.

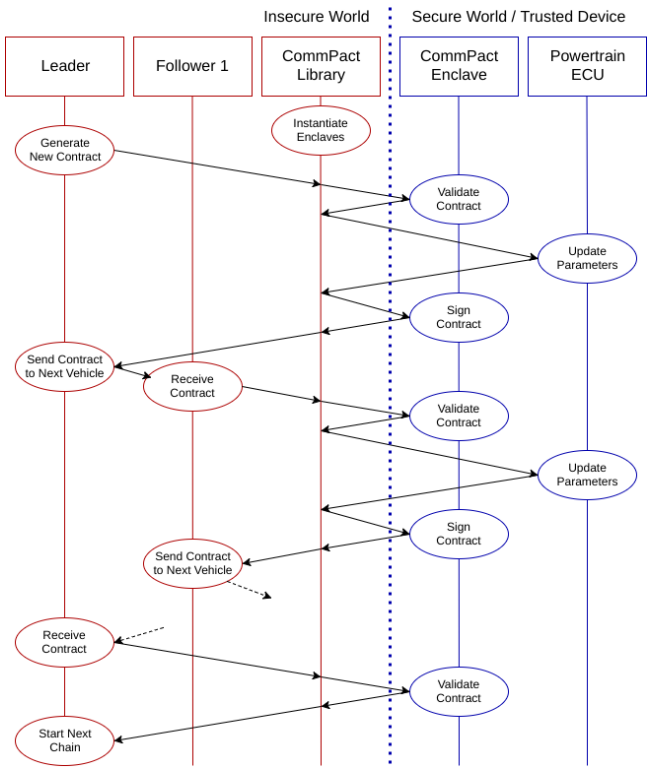

Figure 3: Contract chain critical path from when the leader generates a new contract to when the leader verifies that the contract has been signed by all vehicles. This takes approximately 50 milliseconds for an 8-vehicle platoon.

6 EVALUATION

Manufacturers who wish to implement platooning contracts will need to decide on an appropriate maximum delay from the time an attack occurs until AVs regain autonomy. This delay may be context-dependent, e.g. a 1.5 second delay may be tolerable on an interstate highway with barriers and traditional onramps and offramps, but a

rural highway with perpendicular intersections may necessitate a maximum delay of only 0.75 seconds. This delay is the sum of the Recovery Phase and the Separation Phase, and while the Separation Phase is primarily adjusted through platoon length, the Recovery Phase is dependent on several different variables. The Separation Phase delay can be approximately calculated with Equation 2 and at 100 km/h with a 10% safety factor, ranges from 158 ms (2-vehicle) to 982 ms (8-vehicle). The Recovery Phase delay can be set arbitrarily, but should balance responsiveness and reliability, which are dependent on wireless transmission latency, compute time, and the number of recovery chains that must fail before initiating the ETP. We have chosen an arbitrary, but conservative, reliability goal of 0.001% or lower chance to invoke the ETP due to environmental packet loss per 10 hours of platooning time. We believe that if this overall delay is similar or less than the human Perception Response Time (1-3 seconds), manufacturers may find it advantageous to augment vehicle safety with contracts while platooning.

To evaluate the Recovery Phase delay, we have implemented a prototype of platooning contracts using the PLEXE platooning extension to the Veins simulator. Our simulation runs on a Supermicro server with SGX-enabled X11SSZ-QF motherboard and an Intel Core I7-6700K at 4.0 GHz. Our ECUs are emulated using a Raspberry Pi 3B+ clocked at 1.4 GHz and connected to our server via Fast Ethernet (100 Mbps).[3]

To evaluate the additional processing latency imposed by an enclave, we have extended PLEXE to:

- Instantiate SGX enclaves and connect to an emulated powertrain/braking ECU for each simulated vehicle
- Generate and exchange enclave and ECU keys
- Sign contracts and update enclave and ECU parameters
- Validate signatures and contract parameters within each enclave
- Transmit contract chains between vehicles over the simulated DSRC communication channel

Additionally, we have augmented the simulation to incorporate the wireless transmission delays we measured using two Cohda Mk5 OBUs. The critical path for contract chain completion time is shown in Figure 3, and the results of our simulation over runs with platoons of sizes 2 through 8 are shown in Figure 4.

6.1 DSRC Wireless Latency

Although recent works [15, 39] have shown average delays for DSRC messages to be between 1 ms and 3 ms per transmission depending on data rates and conditions, these studies measure DSRC messaging between semi-trailer trucks and over long (100m-500m) distances, which may be worse than seen in a typical passenger-vehicle platoon. To evaluate typical latencies seen in a passenger vehicle environment, we performed our own measurement study on DSRC wireless transmissions using two Cohda Mk5 OBUs with antennas mounted atop a 2013 Toyota RAV4 and 2002 Honda Civic. These vehicles were positioned at one-car-length intervals between 1 and 7 car lengths apart to evaluate the impact that distance has on packet loss and transmission latency. Our contract payload consists of one 64-byte message and between 1 and 8 64-byte ECDSA

[3]Ethernet has been suggested as a replacement for the CAN bus in AVs to reduce latency and improve security. [20, 33].

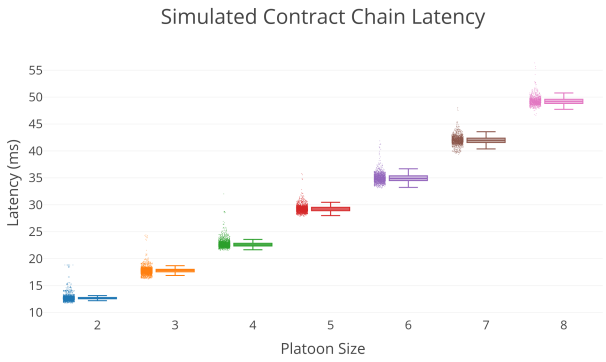

Figure 4: Contract chain latency results for platoon sizes of 2 through 8. Mean latencies of 12.70, 17.80, 22.68, 29.26, 34.98, 42.00, and 49.27 ms, respectively.

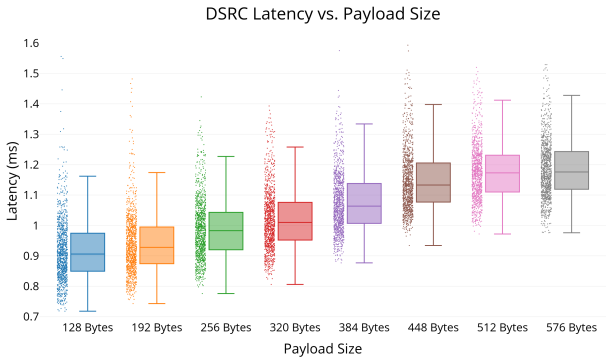

Figure 5: Latencies for messages transmitted between our two test vehicles via DSRC at 18 Mbps data rate. All tests were done with vehicles separated by a gap of approximately 1 meter, except for the 576-byte test which was performed with vehicles at a distance of approximately 7 car-lengths.

signatures (128-576 bytes). In total, we performed 13 trials of 1000 transmissions each.

The first 7 trials were performed one car length apart (1 meter gap) with payloads ranging from 128 bytes to 512 bytes to reflect the additional signatures appended to the contract chain as it traverses the platoon, each time traveling rearward by one car length. The remaining 6 trials were with the cars spaced 2 to 7 car lengths apart and used packet sizes ranging from 256 to 576 bytes, to reflect the messages sent from the tail vehicle to the leader. The results of the first 7 trials at one car length and the 576-byte payload at 7 car lengths are shown in Figure 5. We saw mean latencies ranging from 0.949 ms for 128-byte packets at 1 car-length to 1.200 ms for 576-byte packets at 7 car-lengths, with 98th percentile latencies from 1.258 ms to 1.485 ms.

In all of the 13 trials (13,000 transmissions), we only had a single packet lost (latency greater than 10ms). Our sample size is not large enough to report a packet loss rate with a high degree of confidence, and the results may be different in alternate conditions, so we do not attempt to draw any conclusions about packet loss rate from

Operation	i7-6700K	RPi 3B+	ASIC
Sign	0.193 ms	0.709 ms	0.325 ms [51]
Verify	0.321 ms	1.321 ms	0.212* ms [29]

Table 4: Average ECDSA Sign and Verify latencies on our experimental hardware and on reference ASICs from the literature. *We note that of the options presented by Knezevic [29] we show here the slowest con guration in their typical use category with the shortest critical path. Their fastest ASIC computes a veri cation in 0.037 ms on average.

Figure 6: Average latencies and packet loss rates for 576-byte payloads transmitted over DSRC at various data rates over 1000 iterations. Between trials 1 and 2, we switched which OBU transmitted and which received.

these tests. Consequently, we use a conservative 1% packet loss rate in future calculations of false positives per 10-hour period.

DSRC supports multiple data rates, ranging from 6 to 27 Mbps on a single 10 MHz channel. To evaluate what e ect this data rate had on latency and packet loss, we also performed a test of each supported data rate at a distance of 5 meters, recording latencies for a 576-byte packet as well as the number of packets lost. We performed two trials of 1000 packets at each rate, once from each OBU. Our results are shown in Figure 6. As expected, higher data rates reduce latency. We identi ed 18 Mbps as the optimal data rate to use for our contract latency test (Figure 5), as it minimized latency while retaining near-zero packet loss.

6.2 Compute Latency

Of the total contract chain latency, the majority comes from computational delay. Of this, ECDSA sign and verify operations dominate. For each contract chain, the vehicle enclaves will each sign the contract and verify between one and N (the size of the platoon) signatures, as shown in Figure 2. Each vehicle's ECU will verify the message sent to it by the enclave and sign a response to the enclave. For an 8-vehicle platoon, there are 8 sign and 36 verify operations performed by enclaves on our reference server, and 8 sign and 8 verify operations performed by ECUs on our Raspberry Pi 3B+. Table 4 shows the latencies for these operations on our hardware, as well as reference latencies for these operations on low-powered ASICs presented by the literature. [29, 51]

With the exception of ECDSA sign operations on our server, all other sign and verify operations in our simulation are slower than the current state of the art in automotive ASICs for ECDSA operations. If we were to substitute these reference architecture latencies for our simulation times, we would see an improvement from 49.27 ms to 34.46 ms on average per contract chain for an 8-vehicle platoon.

6.3 AV Contract Con guration

Based on the results presented in Figure 4, we can determine appropriate values for the Recovery Phase timeout at di erent platoon sizes. We wish to reduce false positives (triggering of the ETP due to packet loss) to a manageable level while still minimizing the total Recovery Phase duration. We target a false positive rate of under 0.001% per 10 hours of platooning in an 8-vehicle platoon. Given a contract chain duration of approximately 50ms on average, we can expect to attempt 720,000 contract chains in this period. Using an estimated packet loss rate of 1%, the probability that we will see a false positive in this period is 0.00706% if we set our ETP timeout at 450 ms (9 contract chains), and 0.00055% at an ETP timeout of 500 ms (10 contract chains). Thus, we should choose a Recovery Phase of approximately 500 ms so as to keep the false positive rate below our threshold.

Platoon Size	# Chains	FP Rate per 10 hours	Recovery Phase	Separation Phase	Total Delay
2	7	0.00034%	89 ms	158 ms	247 ms
3	8	0.00012%	142 ms	307 ms	449 ms
4	8	0.00089%	181 ms	451 ms	632 ms
5	9	0.00019%	263 ms	594 ms	857 ms
6	9	0.00078%	315 ms	728 ms	1043 ms
7	10	0.00017%	420 ms	867 ms	1287 ms
8	10	0.00051%	493 ms	982 ms	1475 ms

Table 5: Total expected delay to return autonomy to platooning vehicles at various platoon sizes, assuming a conservative 1% packet loss rate.

The total delay before autonomy can be returned to the platoon for di erent platoon sizes is shown in Table 5. We note that these probabilities can be recalculated continuously during platooning operation and the number of contract chains attempted and the platoon size can be adjusted to compensate for periods of even higher packet loss.

7 PLATOONING SCENARIOS

Besides our base example of a contract chain extending an existing contract, there are several common platooning scenarios we wish to support with contracts. In this section, we describe how our current model of contract chains can be used to support each of these scenarios. If at any time these procedures cannot be completed as described, the platoon may either abort the procedure or trigger the ETP.

The contract chain parameters for each of these scenarios are shown in Table 6.

Parameter	Message Type				Purpose
	E	J	L	S	
Contract ID	3	3	3	3	Identi er for the contract. Increments when the contract is changed.
Sequence Number	3	3	3	3	Identi er for the contract chain. Increments to prevent replays of old messages.
Sent Time	3	3	3	3	Prevents acceptance of delayed packets.
ETP Timeout	3	7	7	7	Extends the timeout period before the Emergency Termination Procedure commences.
Chain Order	3	3	3	3	Ordered list of platoon members. The contract chain follows this ordering.
Speed Restrictions	3	7	7	3	The speed bounds within which all platoon vehicles must remain.
Acceleration Restrictions	3	7	7	3	The acceleration bounds within which all platoon vehicles must remain.
Join Flag	7	3	7	7	Flag indicating that a new vehicle is joining the platoon.
Leave Flag	7	7	3	7	Flag indicating that the originating vehicle intends to leave the platoon.
Split Flag	7	7	7	3	Flag indicating that the originating vehicle intends to split the platoon.
New Vehicle Address	7	3	7	7	The address for the new joining vehicle.
Public Key	7	3	7	7	The public key for the new joining vehicle.
Join Position	7	3	7	7	The platoon position for the new joining vehicle.
Maximum Deceleration	7	3	7	7	The declared maximum deceleration rate for any new joining vehicle.

Table 6: Contract parameters included in each type of contract chain.
E = Contract Extension. J = Join Request Message. L = Leave Request Message. S = Split Request Message.

7.1 Creating and Joining a Platoon

A vehicle may wish to create a platoon with another vehicle or join an already existing platoon. Creating a platoon is a special case of joining where a leader vehicle, L, begins to advertise a new platoon service and can be treated as the leader of a one-vehicle platoon. The procedure for joining varies according to the relative positions of the joining vehicle and the platoon - whether they are in the same lane or in two adjacent lanes.

The joining vehicle, J, initiates the joining procedure by sending a request to the leader vehicle, L. The request includes J's enclave's public key and any maximum deceleration rate it requires of the platoon. After successfully receiving the request, L determines whether J is allowed to join the platoon based on a successful enclave attestation, parameters provided in the request, and the current state of the platoon. If all conditions are met, L will compute the position in the platoon for J to join, inform J and $J+1$ (the vehicle opening the gap, if applicable) of this position, and send a Join Request Message (JRM) to the platoon with J's parameters. The JRM can traverse the platoon in any order, but for simplicity will travel from head to tail. The JRM does not, by itself, add J to the platoon, but must complete (return to L with all signatures) before L will add J to the platoon. J will o cially be part of the platoon when L sends a contract chain extending the ETP timeout that includes J in the platoon Chain Order eld.

The Join procedure is composed of two phases. In the rst phase, J is not yet part of the platoon and is not a party to the contract. The leader sends the existing platoon's address, cryptographic keys, and position information in the JRM. During this phase, J positions itself in preparation for joining the platoon. If J is joining from an adjacent lane, J will position itself adjacent to the platoon while $J+1$ uses the available exibility in the contract's speed restrictions to open a gap. If J is joining from the rear of the platoon, it will position itself a short, but safe following distance from the tail vehicle. This concludes the rst phase. J (and $J+1$, if applicable) will inform L that phase one has completed and L will include J in the Chain Order eld of the next contract chain. Once J receives and signs this contract, it is bound by the contract, limited in its actions, and potentially subject to the ETP at any time.

Once phase two begins, J will begin the process of merging with the platoon. Depending on its starting position, it will either perform a lane change or will speed up within the bounds of the contract to approach the platoon from the rear. We note that it is J's responsibility, if in an adjacent lane, to ensure that it has su cient following distance from any other vehicles in that adjacent lane to avoid collision for the entire lane change period should communications fail and the ETP be required. Additionally, should either J or another vehicle attempt to enter the gap before J has joined the contract, $J+1$ should immediately initiate the splitting procedure to separate itself and any trailing vehicles from the platoon.

7.2 Leaving a Platoon

A vehicle may wish to leave a platoon and be free of its contract for a variety of reasons. Perhaps its highway o -ramp is approaching or the platoon is growing too large.

The Leave procedure is essentially the reverse of the Join procedure. There are two phases. In the rst phase, the leaving vehicle, Lv, will physically separate from the platoon, and in the second phase, will o cially leave the contract.

If Lv is the tail vehicle, it can slowly decelerate within the bounds of the contract parameters until it has reached a su ciently large following distance. If Lv is in the middle of the platoon, or is the Leader, it must perform a lane change. Just as in the Join Procedure, Lv must ensure that the adjacent lane is clear and that there is su cient following distance for it to undergo the ETP if necessary. Once the lane change is complete or safe following distance has been reached, phase two begins.

At the beginning of phase two, Lv has physically separated from the platoon, but is still bound to the terms of the contract. Before it can be released from the contract, it requires the other members of the platoon to agree that it has indeed separated from

the platoon and no longer poses a threat to the platoon's safety, should it suddenly brake or accelerate, L_v will initiate a Leave Request Chain (LRC) which is first sent to $v + 1$ and propagates rearwards towards the tail of the platoon. Once the LRC reaches the tail vehicle, it propagates forward from $v - 1$ until it reaches L. Once L has received the LRC, it can ensure that all vehicles have approved L_v's leaving of the platoon, and L will send L_v a message releasing it from the contract. Since L's enclave will not sign this message until all vehicles have signed the LRC, L_v cannot leave the contract early. Once L_v receives the message releasing it and its enclave verifies the signature, its enclave will inform its ECU that the contract is over and not to trigger the ETP. L_v will also acknowledge the message so that L can stop re-transmitting.

We note that, as a special case, a vehicle in a platoon of size 1 that is not currently undergoing a join procedure and has not recently undergone a split procedure may terminate its contract unilaterally.

7.3 Splitting a Platoon

There may be a variety of reasons to split a platoon. Perhaps, during a Join or Leave maneuver, a human-driven vehicle inserts itself in the gap between platoon vehicles, or perhaps network congestion has increased and to maintain an acceptable Recovery Phase delay, the platoon size should be reduced. We wish to have a way to split the platoon into two smaller platoons safely, while maintaining the contract's guarantees.

Since we may need to split the platoon immediately, we do not wish to require physical platoon separation before splitting the platoon. If a human-driven, and therefore unpredictable, vehicle has inserted itself into the platoon, prudence warrants either an immediate trigger of the Emergency Termination Procedure[4] or an immediate separation of the trailing portion of the platoon. We note that at the time a split is deemed necessary, any trailing vehicles should reject messages from standard contract chains so as to initiate the ETP as soon as possible should the Split fail. If the splitting protocol is unsuccessful, the ETP should take effect for any vehicles trailing the split location, but not for those preceding it.

To enable immediate splitting, we require agreement only from the trailing vehicles. For example, in a 6-vehicle platoon in which a non-platoon vehicle inserts itself between v_2 and v_3, only v_3, v_4, and v_5 must agree to split the platoon. The immediately-trailing vehicle, the Split Leader, initiates the split by sending a signed Split Request Message (SRM) to the tail vehicle. The SRM contains parameters and constraints for a new contract, specifically including new speed and acceleration parameters that will cause the rear platoon to separate. An SRM that does not require the rear platoon to separate is invalid and should be rejected by the tail vehicle. The SRM is passed forward and signed by each vehicle in the platoon until it reaches the Split Leader. As each vehicle accepts the terms of the new contract, the contract takes effect, thus guaranteeing for each vehicle that any vehicles behind it are already subject to the constraints of the new contract.

However, accepting a new split contract does not extend the recovery phase of the contract. If the split procedure fails at any point, the split leader and all trailing vehicles will still enter the Separation Phase in concert and begin to separate according to the ETP. Once the Split Request Message has reached the Split Leader, the Split Leader will become a platoon leader and will begin to initiate contract chain messages itself to maintain the separated platoon. For convenience, the new platoon leader may send the fully-signed SRM to the original platoon leader to inform it that the platoon has split.

As an additional restriction, after a successful Split procedure, the new Leader may not allow the new platoon to regain its previous speed until it has reached a safe following distance from its previous platoon. The enclave enforcing this does not have any way of securely determining the distance between platoons, as sensor input is untrusted, so should use the former platoon's speed parameters and the new platoon's speed parameters to estimate the duration this restriction remains in effect.

8 DISCUSSION

8.1 Redundant Safety Systems

In the course of our research on AV contracts, we have discussed with automotive industry representatives where our approach may be at odds with some safety features of future vehicles. Specifically, redundant braking systems are being developed as a fail-safe for autonomous vehicles in case the primary system fails.[34] In this model, incorporating an enclave as a single point of failure and beyond that, a deliberate restriction on braking ability, may be unpalatable for safety engineers. However, if a redundant braking system is not restricted by the enclave, the enclave can no longer attest to the vehicle's overall behavior since the redundant braking system can potentially be compromised. Ideologically, these two models seem incompatible.

However, by splitting up the purpose of a redundant braking system into specific goals, we may be able to integrate these systems in a compatible way. The first goal of a redundant braking system is to protect against hardware failure. If, for instance, the primary braking actuator physically fails, the backup actuator can take over. This goal is actually compatible with our contract model. Simply have both the braking systems only accept commands from the enclave, and initiate the ETP after a timeout if communication is blocked.

The second goal of the redundant braking system is to, in the event of an imminent collision, to apply the brakes immediately to limit the resultant damage. It is this behavior that is fundamentally at odds with AV contracts, and yet, we can still potentially make some allowances to enable compatibility. If a collision truly is imminent, then the contract shouldn't restrict vehicles from braking. However, if the vehicle can be tricked into thinking a collision is imminent when one is not, then we have negated any benefit from AV contracts. Herein lies the crux of the issue.

In our threat model, we consider both sensor data and the systems that process that sensor data to be untrusted. In the attacks we have seen, adversaries have typically exploited one part of the system (particularly the infotainment system) and then worked from system to system until reaching the CAN bus on which they

[4] Which may require a Recovery Phase delay if communications are disrupted simultaneously

could send malicious commands to the ECUs controlling vehicle motion. We propose that an emergency-only redundant braking system be designed so as to be entirely firewalled off from the rest of the vehicle's electronic systems. A one-way feed of sensor data can be provided to the emergency braking system, which should have its own processing unit dedicated to identifying imminent threats. As this braking system has only one purpose, that of detecting imminent collisions, its processing unit should be simpler and should not trigger the braking system for general control of the vehicle. It should also seek consensus of an imminent threat across multiple sensors, as analog sensor attacks will most likely be much more difficult to perform simultaneously across multiple sensors. If constructed in this manner, it may be possible for vehicles to continue to trust that the redundant braking system will not violate the contract parameters.

8.2 Congestion Testing of DSRC

Studies that evaluate the performance of DSRC protocols [15, 39] are sparse and, like our own testing, evaluate DSRC in good weather on uncongested networks. Our contract protocol is dependent on low-latency communications, and the disruption of these communications will at a minimum increase the delay before AVs can regain autonomy in an emergency situation. More testing in adverse conditions, such as in inclement weather and while sharing a channel with multiple platoons, is necessary to determine whether DSRC can serve as the communication medium for AV contracts in the future. However, AV contracts only use the DSRC protocol as a convenient low-latency medium to transmit messages between vehicles and could use another technology if DSRC is found to be unsuitable. 5G technologies will support ad-hoc communications [16], and 802.11 WiFi is also an option, utilizing the many more channels available in the 5 GHz band.

8.3 Steering Control

Our current AV contract model is limited in what restrictions it can enforce due to the enclave being unable to trust sensor data that is relayed to it through an untrusted system. Therefore, our current contracts focus on limiting speed and acceleration, both of which can be monitored and controlled through our trusted ECUs. However, recent work done by Chen et al. [11] demonstrates a highly-accurate method of detecting lateral motion, such as a lane change, through accelerometer data. An on-chip accelerometer that could provide trusted sensor data directly to the enclave, coupled with enclave-control of the power steering ECU, could potentially enable AV contracts to enforce restrictions on AV steering. This could enable vehicles to place temporary restrictions on one another when in close lateral proximity, further augmenting the safety benefits of AV contracts.

8.4 Needed Enclave Features

AV contracts are built upon the remote attestation and enforcement capabilities of enclaves. However, while this is necessary for AV contracts, it is not sufficient. While it may be relatively straightforward to fabricate an ASIC capable of thousands of ECDSA operations per second, doing so from within an enclave may be a challenge. Intel's SGX enclave cannot currently perform I/O or interact with other modules without the assistance of the untrusted OS, although TrustZone does support some limited direct I/O. An enclave designed for AV contracts must have trusted access to an accelerator for ECDSA operations.

Additionally, enclave hardware must be approachable from a cost perspective. Processors supporting Intel SGX are currently designed for cloud and PC use, not automotive use, and cost hundreds of dollars. This is certainly an unpalatable price point for automotive manufacturers. TrustZone can be found on cheap mobile processors, and is present on the Tegra K1 found in Audi's zFAS [4], central driver assistance controller. However, current generations of TrustZone do not provide the necessary remote attestation capabilities required for AV contracts. We hope that by identifying these requirements, the next generation of enclave technologies can support this use case.

9 CONCLUSION

In platooning, Autonomous Vehicle contracts are likely to be beneficial, significantly increasing the difficulty for adversaries to remotely cause vehicle collisions in platoons where vehicle occupants are most at risk. Our prototype shows contracts can be adjusted to fit changing conditions and maintain a conservative emergency response delay within 1.5 seconds for platoons of 8 vehicles, or as short as 0.25 seconds for 2-vehicle platoons. During an attack, vehicles under contract can physically separate and regain full individual autonomy more quickly than many human drivers can even begin to react. While AV contracts may not be suitable for all environments or situations, their use in normal operating conditions may someday save lives and merits further investigation.

REFERENCES

[1] 2015. Audi mastermind for piloted driving: the central driver assistance controller. https://goo.gl/ySjnpc

[2] 2016. IEEE Standard for Wireless Access in Vehicular Environments Security Services for Applications and Management Messages. IEEE Std 1609.2-2016 (Revision of IEEE Std 1609.2-2013) (2016).

[3] 2017. Continental's cutting-edge brake technology MK C1 enables the next step to highly automated driving. https://goo.gl/tiMcyD

[4] 2018. Redundancy for automated driving. https://goo.gl/3UKUhN

[5] S. J. Anderson, S. B. Karumanchi, K. Iagnemma, and J. M. Walker. 2013. The intelligent copilot: A constraint-based approach to shared-adaptive control of ground vehicles. IEEE Intelligent Transportation Systems Magazine (2013). https://goo.gl/No7k9H

[6] Andrew Baumann, Marcus Peinado, and Galen Hunt. 2014. Shielding Applications from an Untrusted Cloud with Haven. In 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14). USENIX Association.

[7] Norbert BiÄŸmeyer. 2014. Misbehavior Detection and Attacker Identification in Vehicular Ad hoc Networks. Ph.D. Dissertation. Technischen UniversitÃ¤t Darmstadt. https://goo.gl/5Azvmu

[8] Ferdinand Brasser, Urs Müller, Alexandra Dmitrienko, Kari Kostiainen, Srdjan Capkun, and Ahmad-Reza Sadeghi. 2017. Software Grand Exposure: SGX Cache Attacks Are Practical. In 11th USENIX Workshop on Offensive Technologies (WOOT 17) USENIX Association.

[9] C. Campolo, A. Molinaro, G. Araniti, and A. O. Berthet. 2017. Better Platooning Control Toward Autonomous Driving : An LTE Device-to-Device Communications Strategy That Meets Ultralow Latency Requirements. IEEE Vehicular Technology Magazine (2017). https://goo.gl/XewCqh

[10] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In 20th USENIX Conference on Security. https://goo.gl/pnftua

[11] Dongyao Chen, Kyong-Tak Cho, Sihui Han, Zhizhuo Jin, and Kang G. Shin. 2015. Invisible Sensing of Vehicle Steering with Smartphones. In Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services

[12] Kyong-Tak Cho and Kang G. Shin. 2017. Viden: Attacker Identification on In-Vehicle Networks. In *2017 ACM SIGSAC Conference on Computer and Communications Security*. https://goo.gl/Nr23m5

[13] Bruce DeBruhl, Sean Weerakkody, Bruno Sinopoli, and Patrick Tague. 2015. Is Your Commute Driving You Crazy?: A Study of Misbehavior in Vehicular Platoons. In *8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. https://goo.gl/dybbKM

[14] Pedro Fernandes and Urbano Nunes. 2012. Platooning With IVC-Enabled Autonomous Vehicles: Strategies to Mitigate Communication Delays, Improve Safety and Traffic Flow. In *IEEE Transactions on Intelligent Transportation Systems*. https://goo.gl/rUx6Pn

[15] Song Gao, Alvin Lim, and David Bevly. 2016. An empirical study of DSRC V2V performance in truck platooning scenarios. Digital Communications and Networks. https://goo.gl/2nXSQP

[16] A. Gohil, H. Modi, and S. K. Patel. 2013. 5G technology of mobile communication: A survey. In *2013 International Conference on Intelligent Systems and Signal Processing (ISSP)*.

[17] Johannes Götzfried, Moritz Eckert, Sebastian Schinzel, and Tilo Müller. 2017. Cache Attacks on Intel SGX. In *Proceedings of the 10th European Workshop on Systems Security (EuroSec'17)*. ACM.

[18] Marc Green. 2017. *Roadway Human Factors: From Science to Application*. Lawyers & Judges Publishing. https://goo.gl/fBrhae

[19] 5G-PPP Automotive Working Group. 2018. *A study on 5G V2X Deployment*. Technical Report. https://goo.gl/n4fv6G

[20] Nick Ilyadis. 2017. Challenges of Autonomous Vehicles: How Ethernet in Automobiles Can Overcome Bandwidth Issues in Self-Driving Vehicles. https://goo.gl/d2id7Q

[21] Mehmet Sinan Inci, Berk Gulmezoglu, Gorka Irazoqui, Thomas Eisenbarth, and Berk Sunar. 2015. Seriously, get off my cloud! Cross-VM RSA Key Recovery in a Public Cloud. (2015). https://goo.gl/4TeWGe

[22] Dongyao(Tony Jia, Kejie Lu, and Jianping Wang. 2013. On the network connectivity of platoon-based vehicular cyber-physical systems. (2013).

[23] D. Jia, K. Lu, and J. Wang. 2014. A Disturbance-Adaptive Design for VANET-Enabled Vehicle Platoon. *IEEE Transactions on Vehicular Technology* (2014).

[24] Maryam Kamali, Louise A. Dennis, Owen McAree, Michael Fisher, and Sandor M. Veres. 2017. Formal verification of autonomous vehicle platooning. Science of Computer Programming. https://goo.gl/26jfSd

[25] David Kaplan, Jeremy Powell, and Tom Woller. 2016. *AMD Memory Encryption*. Technical Report. https://goo.gl/rhiYd1

[26] Jim Keenan. 2014. Perception Response Time. https://goo.gl/1VBbyG

[27] J. B. Kenney. 2011. Dedicated Short-Range Communications (DSRC) Standards in the United States. *Proc. IEEE* (2011). https://goo.gl/iHJxPd

[28] S. W. Kim, C. Lee, M. Jeon, H. Y. Kwon, H. W. Lee, and C. Yoo. 2013. Secure device access for automotive software. In *2013 International Conference on Connected Vehicles and Expo (ICCVE)*.

[29] M. Knezevic, V. Nikov, and P. Rombouts. 2016. Low-Latency ECDSA Signature Verification - A Road Toward Safer Traffic. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2016).

[30] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. 2010. Experimental Security Analysis of a Modern Automobile. In *IEEE Symposium on Security and Privacy*. https://goo.gl/7qGqvR

[31] Michael P. Lammert, Adam Duran, Jeremy Diez, Kevin Burton, and Alex Nicholson. 2014. Effect of Platooning on Fuel Consumption of Class 8 Vehicles Over a Range of Speeds, Following Distances, and Mass. In *SAE 2014 Commercial Vehicle Engineering Congress*. https://goo.gl/bpTYZb

[32] Jae Hyuk Lee, Jin Soo Jang, Yeongjin Jang, Nohyun Kwak, Yeseul Choi, Changho Choi, Taesoo Kim, Marcus Peinado, and Brent ByungHoon Kang. 2017. Hacking in Darkness: Return-oriented Programming against Secure Enclaves. In *USENIX Security Symposium*.

[33] Seongha Lee and Byungwoo Kim. 2018. Study on Communication Performance of Automotive Ethernet Cable Using Design Factor Analysis. SAE Technical Paper 2018-01-0757.

[34] ARM Limited. 2009. *ARM Security Technology*. Technical Report. https://goo.gl/grznRT

[35] Jiafa Liu, Di Ma, AndrÃĪ Weimerskirch, and Haojin Zhu. 2016. Secure and Safe Automated Vehicle Platooning. *IEEE Reliability Society* (2016). https://goo.gl/RStZbX

[36] Mark Luckevich. 2018. V2V Communication in Platooning. https://goo.gl/k9VUsn

[37] Frank McKeen, Ilya Alexandrovich, Alex Berenzon, Carlos V. Rozas, Hisham Shafi, Vedvyas Shanbhogue, and Uday R. Savagaonkar. 2013. Innovative Instructions and Software Model for Isolated Execution. In *2nd International Workshop on Hardware and Architectural Support for Security and Privacy*. ACM.

[38] Charlie Miller and Chris Valasek. 2015. Remote Exploitation of an Unaltered Passenger Vehicle. https://goo.gl/o2rTXh

[39] Vivek N., S. V. Srikanth, Saurabh P., T. P. Vamsi, and Raju K. 2014. On Field Performance Analysis of IEEE 802.11p and WAVE Protocol Stack for V2V & V2I Communication. In *Information Communication and Embedded Systems (ICICES)*. https://goo.gl/4YTwsE

[40] Paul L. Olson and Michael Sivak. 1986. Perception-Response Time to Unexpected Roadway Hazards. *Human Factors: The Journal of the Human Factors and Ergonomics Society* (1986). https://goo.gl/PHmMKh

[41] Michele Paolino. 2015. ARM TrustZone and KVM Coexistence with RTOS For Automotive. Automotive-grade Linux Summit. https://goo.gl/N5ziKg

[42] Alberto Petrillo, Antonio PescapÃĺ, and Stefania Santini. 2018. A Collaborative Approach for Improving the Security of Vehicular Scenarios: the case of Platooning. (2018).

[43] Alessandro Salvi, Stefania Santini, and Antonio Valente. 2017. Design, analysis and performance evaluation of a third order distributed protocol for platooning in the presence of time-varying delays and switching topologies. (2017).

[44] S. Santini, A. Salvi, A. S. Valente, A. PescapÃĺ, M. Segata, and R. L. Cigno. 2015. A consensus-based approach for platooning with inter-vehicular communications. In *IEEE Conference on Computer Communications (INFOCOM)*. https://goo.gl/ZNPDmr

[45] Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, and Stefan Mangard. 2017. Malware Guard Extension: Using SGX to Conceal Cache Attacks. In *Detection of Intrusions and Malware, and Vulnerability Assessment, International Conference on*.

[46] Michele Segata, Bastian Blessl, and Stefan Joerer. 2014. Supporting platooning maneuvers through IVC: An initial protocol analysis for the JOIN maneuver. In *Wireless On-demand Network Systems and Services (WONS)*. https://goo.gl/MVnfth

[47] Michele Segata, Stefan Joerer, Bastian Bloessl, Christoph Sommer, Falko Dressler, and Renato Lo Cigno. 2014. PLEXE: A Platooning Extension for Veins. In *6th IEEE Vehicular Networking Conference (VNC)*.

[48] Richard Soja. 2014. *Automotive Security: From Standards to Implementation*. Technical Report. https://goo.gl/ZvveUe

[49] Christoph Sommer, Reinhard German, and Falko Dressler. 2011. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing* (2011).

[50] J. Straub, W. Amer, C. Ames, K. R. Dayananda, A. Jones, G. Miryala, N. Olson, N. Rockenback, F. Slaby, S. Tipparach, S. Fehringer, D. Jedynak, H. Lou, D. Martin, M. Olberding, A. Oltmanns, B. Goenner, J. Lee, and D. Shipman. 2017. An internetworked self-driving car system-of-systems. In *12th System of Systems Engineering Conference (SoSE)*. https://goo.gl/BcDXzY

[51] M. Tamura and M. Ikeda. 2016. 1.68 μJ/signature-generation 256-bit ECDSA over GF(p) signature generator for IoT devices. In *2016 IEEE Asian Solid-State Circuits Conference (A-SSCC)*.

[52] R. van der Heijden, T. Lukaseder, and F. Kargl. 2017. Analyzing attacks on cooperative adaptive cruise control (CACC). In *2017 IEEE Vehicular Networking Conference (VNC)*.

[53] Jack Wallen. 2017. Samsung Knox: A cheat sheet. https://goo.gl/GrzC5D

A CALCULATING THE SSDV

Distance Formula:

$$d_{final} = \frac{1}{2}at^2 + vt + d_{initial}$$

$$d_{final} \quad d_{initial} = \frac{1}{2}at^2 + vt$$

$$\Delta d = \frac{1}{2}at^2 + vt$$

Time to reach rest:

$$\Delta v = at$$

$$v_{final} \quad v_{initial} = at$$

$$t = \frac{v_{final} \quad v_{initial}}{a}$$

$$v_{final} = 0 \quad) \quad t = \frac{v_{initial}}{a}$$

Displacement of Vehicle after beginning to decelerate to rest:

$$\Delta d = \frac{1}{2}at^2 + v_{initial}t$$

$$\Delta d = \frac{{v_{initial}}^2}{2a} \quad \frac{{v_{initial}}^2}{a}$$

$$\Delta d = \quad \frac{1}{2}\frac{{v_{initial}}^2}{a}$$

Goal: Calculate t_{sep}.

At time t_0, follower f begins to decelerate. At time t_{sep}, it has reached a distance d_{sep} from leader l and a new velocity v_1.

$$v_1 = v_0 + a_0 t_{sep}$$

$$d_{sep} = \Delta d_l \quad \Delta d_f + d_0$$

$$d_{sep} = {}^1v_0 t_{sep}{}^o \quad {}^1\frac{1}{2}a_0 t_{sep}^2 + v_0 t_{sep}{}^o + d_0$$

$$d_{sep} = \quad \frac{1}{2}a_0 t_{sep}^2 + d_0$$

At time t_{sep}, both vehicles are released from the contract and can begin decelerating at their maximum rates, a_1 and a_2. They reach rest at different times t_{l_stop} and t_{f_stop} after traveling some distances Δd_l and Δd_f, culminating at a final distance between vehicles d_{stop}.

$$d_{stop} = \Delta d_l \quad \Delta d_f + d_{sep}$$

$$d_{stop} = {}^1 \quad \frac{1}{2}\frac{v_0^2}{a_1}{}^o \quad {}^1 \quad \frac{1}{2}\frac{v_1^2}{a_2}{}^o + d_{sep}$$

Substitute for v_1 and rearrange in terms of d_{sep}:

$$d_{sep} = \frac{1}{2}\frac{v_0^2}{a_1} \quad \frac{1}{2}\frac{{}^1v_0 + a_0 t_{sep}{}^{o2}}{a_2} + d_{stop}$$

Now, with two linear equations for d_{sep}, we can set them equal and solve for t_{sep}:

$$\frac{1}{2}a_0 t_{sep}^2 + d_0 = \frac{1}{2}\frac{v_0^2}{a_1} \quad \frac{1}{2}\frac{{}^1v_0 + a_0 t_{sep}{}^{o2}}{a_2} + d_{stop}$$

After simplification:

$$\begin{aligned} {}^1a_0^2 a_1 \quad a_0 a_1 a_2{}^o t_{sep}^2 + {}^12a_0 a_1 v_0{}^o t_{sep} + \\ v_0^2{}^1a_1 \quad a_2{}^o + 2a_1 a_2{}^1d_0 \quad d_{stop}{}^o = 0 \end{aligned} \tag{2}$$

B CHANCE OF ETP DUE TO PACKET LOSS

In our model, attacks are detected through a timeout. Should the contract chains passed between vehicles stop arriving, after some period of time, we invoke the ETP to ensure the safe separation of the platoon. However, there is a small chance that, simply due to environmental packet loss, enough consecutive contract chains fail to cause at least one vehicle to reach its timeout and initiate the ETP. We would like to quantify this chance so we can ensure it stays under some configurable bound. In our evaluation, we pick an upper bound of 0.001% chance per 10 hours of platoon operation, assuming a uniform 1% packet loss rate. Higher packet loss rates would necessitate either tolerating a greater chance of false positive, or extending the duration of the Recovery Phase, to the detriment of safety. If necessary, the platoon may be split to reduce the duration of the Recovery Phase to within acceptable bounds. Here, we quantify the probability that there will be at least one false positive (FP) in a large number of contract chains based on the packet loss rate, the length of the platoon and the Recovery Phase duration we are willing to endure.

For a contract chain to succeed, every individual packet transmission within the chain must succeed. For packet loss rate p and contract chain of length L, the probability of an individual contract chain failing $P_f{}^1L^o$ is:

$$P_f{}^1L^o = 1 \quad {}^11 \quad p^{oL}$$

For n repeated and independent contract chains, the probability that there is at least r or more consecutive failures, $P_{FP}{}^1n;r^o$, due to random packet loss equals the probability that there is at least r or more consecutive failures in the previous $n \quad 1$ contract chains, $P_{FP}{}^1n \quad 1;r^o$, plus the probability that nth contract chain is a failure and this nth failed contract chain is the r'th consecutive failure, $P^0{}^1n;r^o$:

$$P_{FP}{}^1n;r^o = P_{FP}{}^1n \quad 1;r^o + P^0{}^1n;r^o$$

$P^{0}(n; r)$ is composed of:

$$(1 - P_{FP}(n - r - 1; r)) \cdot (1 - P_f)^r$$

which is the probability that all below conditions are met:

- There are not r or more consecutive failures in the first $(n - r - 1)$ contract chains.
- The $(n - r)$'th contract chain succeeds.
- The last r contract chains, including the nth, all fail.

Therefore, the possibility that there are at least r or more consecutive failures $P_{FP}(n; r)$ is:

$$P_{FP}(n; r) = P_{FP}(n - 1; r) + (1 - P_{FP}(n - r - 1; r)) \cdot (1 - P_f)^r$$

using this equation, the probability of a false positive can be recursively calculated for different initial parameters p, L, n, and r. Table 3 and Table 5 show our results. If in-vehicle computation is limited, this calculation can be pre-computed and a simple lookup table to adjust Recovery Phase duration can be used during platooning itself.