May 16, 2019

State-of-the-art Security Attacks on Sensor-Rich Automated Vehicles with Wireless Linkage for Remote Access and Control: A Survey

Dajiang Suo, Shuyi J Chen, Leah Goggin

In this paper, we conduct a survey on the state-of-the-art security attacks on sensor-rich automated vehicles with remote access and control capabilities. In particular, we examine threats regarding on-board sensors and external communication linkages for remote access (or control) and software updates. First, (words about physical access/RKS). Second, a comprehensive review of attacks on image-based perception systems is presented with respect to the security of machine learning algorithms for object classification. To evaluate the vulnerability of ML algorithms to such attacks in real-world scenarios, we implement untargeted attacks on residual networks (ResNet) [19] using German traffic sign benchmark [20]. The result suggests that potential risks caused by adversarial examples do exist when applying deep neural network to traffic sign classification. Third, we will examine the connectivity and exploitation potential of other computational elements, particularly the head unit and CAN bus.

We present mitigations for each type of attack described and conclude with high-level recommendations for improving automotive security. We do not base our discussion on a particular commercial product in the market, but on published conference and journal papers, patents and security blogs that are publicly available.

I. Introduction

It's well-known that the security of embedded systems tends to be inconsistent at best, containing "vulnerabilities that we found were the kind that existed on PCs in the early to mid-1990s, when computers were first getting on the Internet" (Stefan Savage, quoted in Greenberg). This is true for both technical and economic reasons–embedded systems are perceived to be less likely to be attacked, and therefore merit fewer resources dedicated to defense. This effect extends to automotive security, as this literature review will describe. In response to the point that it's probably easier to hire a killer than to fatally attack someone's car, automotive security researcher Charlie Miller responds that "this is true of all car hacking. [I]t's a tough way to hurt someone."[32]

This calculus changes, however, with the prospect of the large-scale introduction of self-driving cars. As automation in cars becomes 1. more common, i.e. the number of automated cars on the road increases, and 2. more complete, i.e. the extent to which a car's physical movements can be controlled by some computational element increases, the number and skill of adversaries should be expected to increase sharply. In particular, a hypothetical fleet owned by a company like Uber or Waymo is likely to all run the same software, and therefore all contain the same vulnerabilities. While a single vehicle may not be "worth" attacking, many vehicles at once present an infrastructure-scale target that may attract criminal and even state-sponsored attackers.

In this paper, we describe the attack surface of sensor-rich vehicles, dividing it into physical security systems, the self-driving system (where applicable), and other computational elements. For each section, we review the current literature on known exploits and their mitigations. We conclude with a high-level overview of observed trends and suggestions for how the security posture of these vehicles can be improved in response.

II. Physical Security

The physical security of a vehicle is usually the first line of defense against malicious actors. The potential damage to the vehicle is nearly limitless if an adversary has physical access. From inside, the adversary can, with the right tools, access the on-board diagnostics (OBD) information, the infotainment unit, and in most cases, steal the car. Car manufacturers use both mechanical locks and electronic security systems to ensure the physical security, we focus our research efforts on latest attacks targeting the electronics means of entry.

A. Remote Keyless Systems (RKS)

Remote Keyless Systems (RKS) is a common way for users to unlock their cars electronically. In 2018, 62% of all new cars sold in the US are equipped with some form of RKS.[11] In the most basic scheme, the user has a keyfob that can issue commands wirelessly to the car with a button press. The commands are generally encrypted data modulated with VHF. The car demodulates the signal, extracts, decrypts, and acts upon the command. Since the issued commands travel over the air, any adversary can listen and analyze the signals using the correct antennae and a computer with software defined radio (SDR) tools. Consequently, the encryption scheme becomes crucial.

Garcia et. al. in 2016 presented several ways to defeat weak RKS implementations, highlighting weakness in a number of Volkswagen in-house encryption solutions, and describing a method to attack Hitag-2, an integrated data transponder solution sold by NXP to car manufacturers.[12] In 2018, Verstegen et. al. presented a brute-force attack on Hitag-2 that can recover the private key using a desktop graphics card in around a minute.[13]

Newer encryption schemes are relatively more secure, and without a method of attacking the cryptographic system of the RKS, Ibrahim et. al. in 2018 showed that the attackers can resort to the strategy of jam-listen-replay.[14] The attack involves installing a jamming/recording device on the vehicle. When the user issues a keyfob command, the car is unable to receive it properly, thereby forcing the user to unlock the car physically. At a later time, the attacker can use the recorded command to unlock the car when the user is away.

B. Passive Keyless Entry Systems (PKES)

The basic RKS scheme is unidirectional, the data flows from the keyfob to the car. More advanced RKS schemes uses bidirectional data flow, where the vehicle, when in proximity to the keyfob, authenticates the signal via a challenge-and-response protocol. The vehicle only acts upon the commands, including allowing the engine to start, after a successful handshake. This scheme is generally called Passive Keyless Entry and Start (PKES). Some vehicles will only implement the engine immobilizer part of the PKES, and use basic RKS for the rest of the vehicle.

Paradoxically, the PKES system is even simpler for an adversary to attack. All that's involved is a pair of transponders that extends the keyfob signal. In 2017, Team Unicorn, a security research group in China demonstrated a method of amplifying the keyfob signal to over 1000 feet. The group was able to trick a PKES-equipped car to unlock when the actual keyfob is one street block away.[15] This type of signal relay attack is ideal for malicious actors since it works across multiple encryption and authentication schemes. The signal relay attack is not new, as early as 2016, ADAC, a German automotive group showed that many car models sold between 2013 and 2015, by Audi, BMW, Toyota, and Honda, among others, are susceptible to the attack, albeit with a shorter relay range.[16]

C. Selected Survey Details

1. Garcia et. al.

In 2016, Garcia et. al. presented a paper at the 25th USENIX Security Symposium detailing 2 case studies regarding RKS security. Using commercially available radio components and keyfob firmware recovery programs, they were able to capture the keyfob command signal for analysis.

In the first study, the group analyzed 4 remote keyless entry schemes (VW-1, VW-2, VW-3, and VW-4) used by the Volkswagen group in their cars sold in Europe. They found that the older system, VW-1, did not use any cryptographic keys to encode the keyfob commands. Its rolling code security is entirely based on an obfuscation function implemented by a linear feedback shift register. An attacker only need to record and analyze one command signal to unlock the car. In the VW-2 and VW-3 system, the group determined that the remotes used a proprietary block cipher to encrypt the data payload. While they are more secure than VW-1, the group found that each scheme used a fixed 120-bit global master key. It is therefore possible for adversaries to recover the master keys from the keyfob and use them to unlock all vehicles implementing the respective schemes. VW-4 system uses a Feistel structured block cipher with a 128-bit encryption key. Again, the encryption key is the same for all vehicles implementing the scheme.

In the second study, the group tackled the Hitag-2 system sold by NXP and used by various car manufacturers as recently as 2016. The Hitag-2 system also uses a block cipher to encrypt the data payload. The group was able to implement a correlation attack to clone a keyfob using 4 recorded command signals. The group concluded that while the

Volkswagen group's scheme has a weakness in key diversification, the Hitag-2 scheme suffered from a cryptographically weak cipher.

2. Verstegen et. al.

In 2018, Verstegen et. al. improved upon previous work on Hitag-2 and was able to successfully recover the encryption key used by individual keyfobs in 1.2 minutes using only 2 eavesdropped command signals. The only hardware required was a desktop PC with a graphics card. The group also included the source code for the attack in the paper.

III. Self-Driving System

Research on self-driving systems has been mainly focus on on-board sensors for perception such as Lidar, camera and Global Positioning Systems, etc. For example, Shin [22] and Petit [21] demonstrate the possibility of spoofing point-of-clouds in Lidar by recording outbound optical signals and replaying back to optical receivers within Lidar. Petti et al. illustrate a saturation attack on Camera by pointing a LED light on the lens [21]. Yan implemented an attack on the ultrasonic sensors for automatic parking features in Tesla model-S [23].

In this section, we focus on "scenery attacks" that can result in incorrect classification by machine learning algorithms, namely, manipulating and modifying physical objects in real-world environment such that the distribution of test data becomes diverged from the distribution of the training data [24]. In particular, we look at the attacks caused by adversarial perturbations on traffic signs that are critical to the decision-making (e.g., stop, yield, decelerate, etc.) by autonomous vehicles. We should address though that there exist other adversarial examples of scenery attacks on perception systems of autonomous vehicles. For example, the Keen security lab [27] has conducted an experiment attack on Tesla's auto-pilot system. The car is diverged from the correct lane when attackers paste stickers to mimic the lanes on the road.

A. Attacker's model

We assume that adversaries have access to the testing set, i.e., adversaries can modify the traffic signs in the street by adding random stickers, createing adversarial examples of traffic signs that are printed out and pasted on the real ones, or replacing original ones with faked signs. Although Lu et al. [**?**] argue that "there is no need to worry about adversarial examples" because adversarial examples may not work when captured by camera from different angles and distances, Eykholt et al. [26] has successfully create robust adversarial examples (i.e., stop signs) that can fool deep neural networks in different situations.

B. Untargeted attacks on ResNet

We perform adversarial attacks on ResNet [19] by using German traffic sign recognition benchmark (GTSRB) [20]. Fig. 1 illustrates the detailed process. Feature extraction is first applied to ResNet to modify the number of output in the last layer to 43, which corresponds to the number of labels in the GTSRB dataset. Then, we conduct adversarial training to generate test sets with small perturbation ad compare the predicated labels with labels predicated on test set without adversarial perturbations.

1. Results and discussion

- Safety risk due to adversarial perturbation: the results from adversarial machine learning imply safety risks of automated-driving functions. Fig. 2 gives an example of STOP sign that is misclassified as speed limit sign (60km/h) after adding the small perturbation derived from adversarial training. Consider the scenario where an autonomous vehicle that base its object classification function on deep neural network such as ResNet is supposed to stop at an uncontrolled intersection following the indication of a STOP sign. If that sign was classified by ResNet as speed limit sign, the vehicle will not stop and may even run into pedestrians in the worst-case scenario. We should address though that not all attacks on traffic signs can cause safety hazards. For example, misclassfication of traffic sign "General caution" (label number '18' in the GTSRD dataset) may not result in safety risks. Fig. 3 list five traffic signs that we believe can pose most safety risks if misclassified, including speed limit (30km/h), speed limit (80km/h), Right-of-way at the next intersection, Yeild, STOP.
- Risk priority for different traffic signs: another interesting finding in our implementation of adversarial attacks is

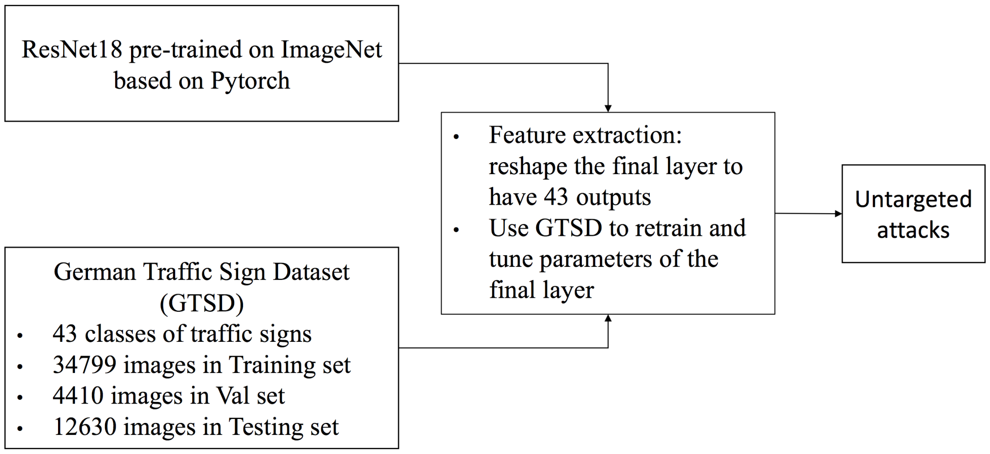
ResNet18 pre-trained on ImageNet based on Pytorch
German Traffic Sign Dataset (GTSD)
43 classes of traffic signs
34799 images in Training set
4410 images in Val set
12630 images in Testing set
Feature extraction: reshape the final layer to have 43 outputs
Use GTSD to retrain and tune parameters of the final layer
Untargeted attacks

Fig. 1 Detailed process of implementing adversarial attacks on ResNet

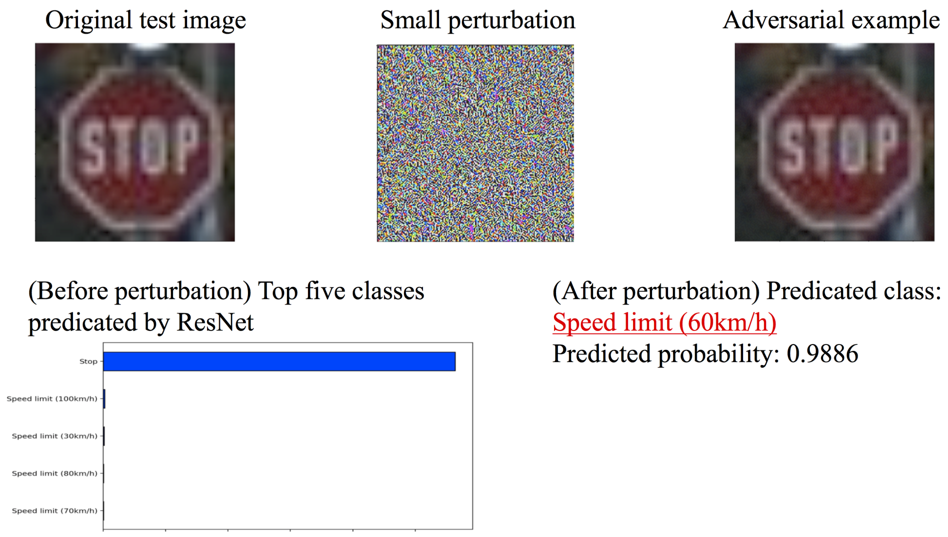

Fig. 2 An example of predication result of STOP sign (real class) predicated by ResNet before and after adversarial perturbation

that some categories in the GTSRD is more vulnerable to adversarial perturbation than other categories, as shown in Fig. 3. We perform untargetted attacks [**?**] by using the fast gradient sign method (1 step) on the test images of the five categories mentioned above. According to results in Fig. 3, Right-of-way at the next intersection is most vulnerable to small (adversarial) perturbation (with almost 0.9 decrease in predication accuracy), while the predication accuracy by ResNet on STOP sign has decreased by around 0.6.

- Pay attention to unintended perturbation: given the result that even small perturbations (one step training using fast gradient sign method) cuased by untargetted attacks can make deep neural network misclassify traffic sign, we believe it is reasonable and necessary to consider safety risks resulting from "unintended" perturbations although such risks may not necessarily lead to incidents. Fig. 4 show two pictures of traffic signs that are taken near the dorm of one of the authors. We can see people have added markers randomly although they may not have malicious intent.

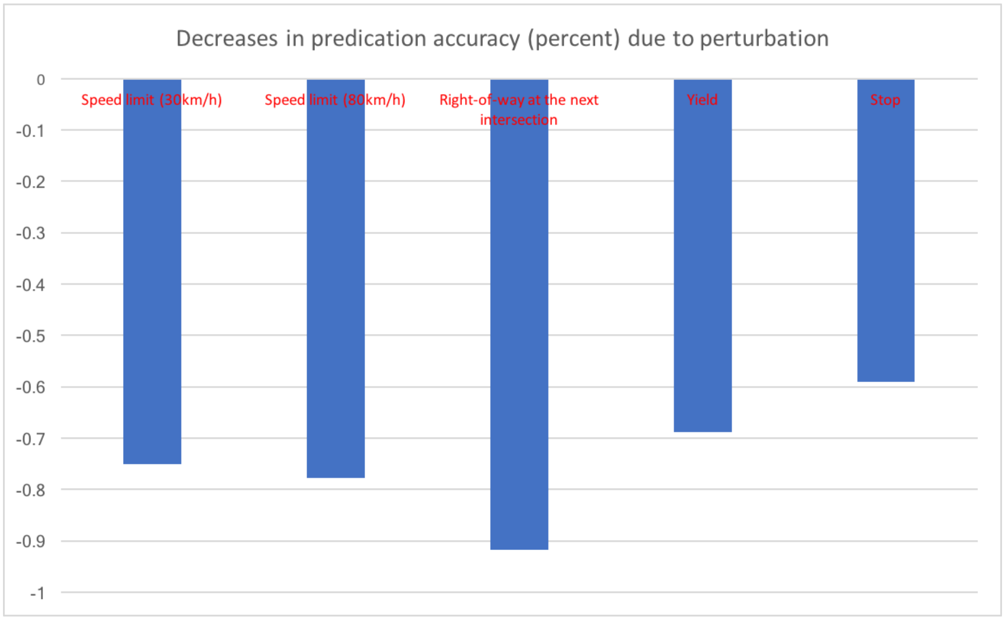

Fig. 3 Comparing vulnerabilities of different traffic signs with respect to adversarial perturbation

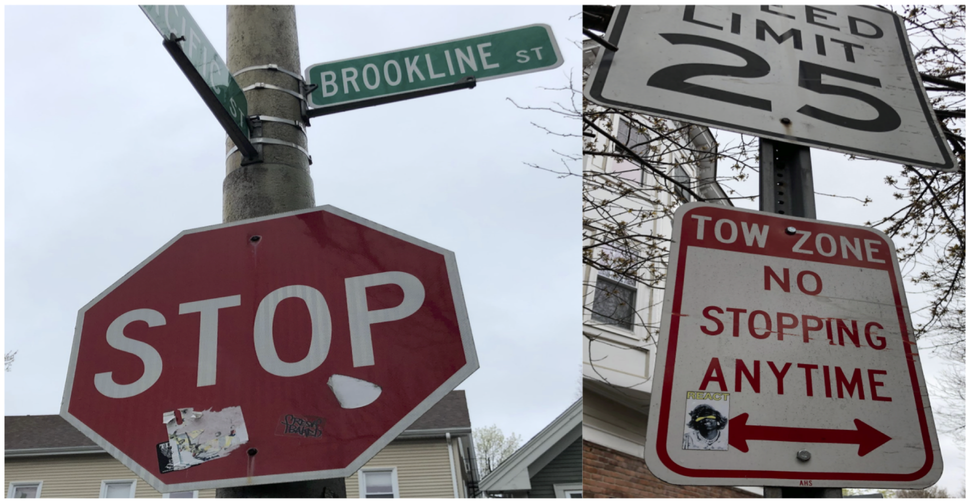

Fig. 4 Examples of unintended perturbations on traffic signs

C. Mitigation solutions

There are three types of solutions that can be used to mitigate such risks from different research perspectives. In the research community of machine learning, Madry et al. [**?**] suggest the use of robust optimization techniques during the training period to improve the robustness of deep neural network. For researchers in robotics field, sensor fusion techniques has been proposed for sanity checks based on inputs from on-board sensors including camera, Lidar, GPS and digital maps that may include information of positions of traffic signs [30]. In the wireless communication domain, Chow and Tsai [31] suggest the use of visible light communication to enable the communication and authentication between vehicles and infrastructure. They assume that there is a certificate authority that issue valid certificates to each vehicle and traffic signs. This solution requires the improvement of road infrastructure and connectivity of vehicles and may not be feasible in the short run.

IV. Other Computational Elements

A. ECUs, CAN, and OBD-II

Electronic Control Units, or ECUs, are the microcomputers scattered throughout the car which may control anything from the windows to the engine. They may reside on any of several types of buses, including Controller Area Network (CAN), Local Interconnect (LIN), MOST, Flex-Ray, Automotive Ethernet, and CAN-FD.[9] This review will focus on the CAN bus, which is the "de facto standard because it dramatically decreases the number of communication lines required and ensures higher data transmission reliability."[10]

Woo et. al. note that CAN was not designed for security. All messages are sent as broadcasts with no authentication or confidentiality built into the protocol, such that with diagnostic tools it is possible to connect to the bus and record data frames which can later be used in a replay attack. An attacker can obtain valid data frames for a target vehicle by connecting a diagnostic tool to another car of the same model, as Woo et. al. do in their demonstrated attack, so an attacker with a remote method of writing to the bus can carry out an attack without ever needing physical access to the target vehicle. That is to say, any reasonably well-resourced attacker with a method of putting frames on the bus is limited in what control they can exert only by the extent to which ECUs are controlled by the CAN bus at all. This limitation can be significant, as different functions may be located on different buses with varying levels of difficulty to access.

There are several routes an attacker might leverage to gain access. The CAN bus is externally accessible via the On-Board Diagnostics II (OBD-II) port. A range of consumer devices for accessing this port are available on the market, and they often pair with the user's smartphone and/or the car's infotainment unit via Bluetooth (for example, the Zubie Key and the PLX Kiwi). Some auto insurance companies ask customers to connect a custom dongle to their OBD-II port in order to monitor their driving habits. In some newer cars, the infotainment unit has direct access to the CAN bus (see Miller and Valasek proof-of-concept below).

Several security assessments and proof-of-concept attacks have been demonstrated:

1. Woo et. al.

This team uses a malicious Android app as an attack vector. Masquerading as a legitimate automotive diagnostics app, their app pairs over Bluetooth with one of the OBD-II dongles described above. Using the phone's connectivity (LTE, 4G, etc.), the app contacts an attacker server and can perform operations such as uploading sniffed CAN frames to the attacker for analysis and downloading previously-harvested CAN frames to be injected through the OBD-II port.

2. Miller and Valasek

In a first round of reverse engineering, they gain the (limited) capability to physically influence the behavior of the car with a laptop connected to the OBD-II port.[7] After criticism from manufacturers that an attack requiring physical access is unrealistic, they demonstrate a remote attack exploiting a Jeep Cherokee's Harman Kardon infotainment unit.[6] They gain a foothold by spotting an open port intended for inter-process communication and discover that it gives unauthenticated D-Bus access. From there, they find multiple ways of executing arbitrary shell commands (at least one command injection vulnerability, but also a method call that runs arbitrary commands by design). This attack is initially carried out over the in-car WiFi, which has a roughly 32-meter range. The authors eventually discover that not only does it work with a cellular connection using a femtocell, but that Sprint towers don't block the phone-to-vehicle connection, and "Chris in Pittsburgh [verifies] he can access the D-Bus port of the Jeep in St. Louis."

They also note poor randomness in the generation of the in-car WiFi WPA2 password. By disassembling the password generation function, they show that it is a deterministic function of epoch time at the time of generation, measured from the first time the system is booted. They estimate that if an attacker could guess this time within a month, the password would be brute forceable in two minutes. Not only that, but doing so is likely not necessary, since a race condition is present in which the WiFi password is set before the system retrieves the actual time from GPS, etc. Miller and Valasek found that at least in the case of the car they red teamed, the password corresponded to an epoch time of 32 seconds.

These methods give code execution on the OMAP chip. The CAN bus is managed by a separate chip on the same board, the v850, which not only communicates with the OMAP chip but can be updated by it. The authors proceed to reverse engineer the v850 firmware, modify it to "accept arbitrary CAN messages via a wireless interface", and flash the new, malicious firmware via the compromised OMAP chip.

The result is a fully-remote attack that allows arbitrary CAN bus access to "many Fiat-Chrysler vehicles" which may be "located anywhere in the United States", with which the authors are able to influence critical functions including steering and breaking. This demonstration "forced a 1.4 million vehicle recall by FCA" and "changes to the Sprint carrier network".

3. KeenLab

In the *Free-Fall: Tesla Hacking 2016* presentation to the BlackHat conference, the KeenLab team describes a similar ECU takeover via the infotainment unit.[17] Starting from a use-after-free vulnerability in WebKit, they get a low-privilege shell and exploit a kernel bug to get root access. They identify a "gateway" chip bridging the ethernet network and the CAN bus and find that it is only verified with a checksum. After reflashing the gateway chip, they have limited CAN bus access. They go on to find a fixed PRNG seed in the code generating the key necessary for full access to other ECUs.

4. Thuen

In Cory Thuen's talk at the 2015 S4 Conference, he describes red-teaming the Progressive car insurance dongle. He finds that firmware updates to the dongle are unsigned, boot is insecure, cellular is unauthenticated, and in general no secure communications are implemented.[18] He doesn't develop a proof-of-concept exploit, but says that the goal was just to assess the security posture of the dongles and concludes that it's nonexistent.

As a partial mitigation, Woo et. al. suggest extending the CAN protocol to provide confidentiality and authentication. They suggest signing and encrypting data frames using (different) ephemeral keys, providing forward and backward secrecy and preventing replay attacks. They are able to find space for a MAC in the existing CAN header format by sacrificing the 16-bit CRC field (since the MAC provides integrity checking) and the rarely-used first 16 bits of the extended ID field. Because CAN messages are ACKed and tagged with a sender ID, they also suggest that every ECU on the bus maintains a counter of messages exchanged with every other ECU, another line of defense against replay attacks.

Their paradigm assumes long-term symmetric keys pre-shared via a secure channel and correct behavior on the part of ECUs. They also allow for external devices to be added. While such devices are required to authenticate to the gateway ECU with a certificate, this protocol does not mitigate the case in which a legitimate device becomes compromised, as Thuen and Miller and Valasek show is a realistic scenario.

B. Firmware and Updates

At present, ECU firmware updates are delivered to the OEM from the supplier of the particular ECU (possibly after being passed up the chain from the supplier of some subcomponent), and generally distributed by the OEM to dealerships and/or owners.[9] The last step of this update process, i.e. the delivery of the firmware to the vehicle itself, may happen over-the-air (OTA) or require physical access. Firmware updates to automotive embedded systems are generally subject to the same risks as PCs, such as exploitation of insecurely-implemented distribution processes or compromise of repositories.[9] The defenses against these sorts of attacks are the same as in normal PC updates, namely code signing and verification.

However, Karthik et. al. observe in *Uptane: Securing Software Updates for Automobiles* that automotive firmware updates present challenges not found in the PC updating process. They note that "there are many different types of computers to be updated on a vehicle" and that conventional update hardening mechanisms "do not address [all] security attacks that can cause a vehicle to fail... [or] become unsafe", which may not require attackers to compromise the secret key. They also point out that remote update mechanisms are likely to proliferate, replacing the current most common mechanism of updates being applied by a technician or owner with physical access. As Miller and Valasek concede, while an attacker with physical access shouldn't be ignored as a threat, the possibility of a remote attacker manipulating firmware is a much more serious one.

1. Demonstrated Attacks

Both the Miller and Valasek remote attack and the KeenLab attack described in the previous section involve installing modified firmware on ECUs, capitalizing on code signing that is either flawed or missing entirely and relying on 1. first exploiting the infotainment unit and 2. the fact that the infotainment unit has the ability to update ECU firmware.

While these demonstrations are important from a security perspective, they do not represent a fundamental challenge to automotive security, since their mitigations are a matter of correct implementation of well-known measures, i.e. code signing.

2. Hypothetical Unaddressed Attacks

Karthik et. al. identify several specific complications to firmware update security that are not addressed by existing secure update frameworks:

- An attacker might send random data to an ECU that only has the precise amount of storage needed for a single firmware image, preventing it from booting even if its bootloader correctly determines that the loaded faux-firmware is illegitimate
- An attacker might jam or slow communication channels so that updates cannot happen at all, or man-in-the-middle and send an old version to a vehicle attempting to update, causing a known-bad firmware version to remain in place. A specific challenge is that "unlike desktop and server machines, ECUs do not typically have reliable real-time clocks. Thus, the primary [ECU] may not be able to tell whether a metadata file has expired."
- An attacker could block update traffic to some but not all ECUs, causing incompatible versions to be installed.
- An attacker who has compromised the signing key may release "an arbitrary combination of new versions of images", giving them far more fine-grained control than the previous attack and more opportunity to load a combination of firmware versions that will cause some undesired behavior.

Karthik et. al. present a full grid of possible attacks vs. attacker capabilities, chosen from various combinations of MitM from inside the vehicle, from outside it, and different key compromises. This assessment is done assuming the system in place is a lightly modified version of the widely-used and straightforwardly-named update framework The Update Framework (TUF).

Karthik et. al. offer mitigations in the form of *Uptane*, their update framework designed for use in automotive systems. In particular, they recommend the following features:

1) Additional Storage: Giving each ECU enough storage to always maintain a known-good copy of its firmware during the update process mitigates the first attack described above.
2) Broadcasting metadata: The CAN bus itself can be designed in such a way that it is impossible for a compromised primary ECU to show other ECUs different metadata, making it more difficult to load incompatible firmware versions.
3) Vehicle Version Manifest: A manifest of what firmware version is running on each ECU, cryptographically signed by each ECU and broadcast throughout the network, provides another defense against incompatible versions being loaded.
4) Time Server: Allowing ECUs to periodically sync with an (authenticated) outside real-time clock helps to mitigate attacks relying on ECUs not realizing that a file has expired.

V. Conclusions/Recommendations

We identify the following overarching themes and observations:

A. Automotive Security Is Hard

Many of the vulnerabilities described are essentially a failure to even attempt to mitigate known attacks (for example, not cryptographically signing firmware). While addressing this low-hanging fruit is certainly worth doing, even a serious attempt by auto manufacturers to comply with best practices is unlikely to secure vehicles against well-resourced, skilled attackers. There are many factors that make semi- or fully-automated vehicles difficult to secure, including a multitude of I/O channels (sensors, GPS, USB, WiFi, Bluetooth...), many submodules designed and supported by different vendors, submodules working together in new and complicated ways, and unpredictable real-world interaction that can't be exhaustively tested.

B. Build for Resilience to Failure

We recommend that automakers strive for perfect security, but recognize that achieving it in practice is unlikely. The connectivity and permissions within the system should be designed in such a way that more important, safety-critical components are reliable in the case of failure of more-likely-to-be-exploited components. *Uptane* is a good example of designing for resilience against partial compromise. In particular, several proof-of-concept attacks start by exploiting a bug in a non-critical function, such as the in-car WiFi, and escalate to flashing firmware to various components and controlling the CAN bus. A more secure design would be one in which the firmware update mechanism is as isolated as possible.

References

[1] Stephen Checkoway., *Comprehensive Experimental Analyses of Automotive Attack Surfaces*. In USENIX Security Symposium, vol. 4. 2011.

[2] Koscher, K. et al., *Experimental security analysis of a modern automobile*. In 2010 IEEE Symposium on Security and Privacy (pp. 447-462). IEEE.

[3] Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy, *Explaining and harnessing adversarial examples.*. arXiv preprint arXiv:1412.6572.

[4] Kurakin, Alexey, Goodfellow, Ian and Bengio, Samy, *Adversarial examples in the physical world.*. arXiv preprint arXiv:1607.02533.

[5] Eykholt, Kevin et al., *Robust physical-world attacks on deep learning visual classification*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1625-1634).

[6] Miller and Valasek, *Remote Exploitation of an Unaltered Passenger Vehicle*. http://illmatics.com/Remote%20Car%20Hacking.pdf.

[7] Miller and Valasek, *Adventures in Automotive Networks and Control Units*. http://illmatics.com/car_hacking.pdf

[8] Greenberg, Andy, *Hackers Reveal Nasty New Car Attack with Me behind the Wheel*. https://www.forbes.com/sites/andygreenberg/2013/07/24/hackers-reveal-nasty-new-car-attacks-with-me-behind-the-wheel-video/#64788b3a228c

[9] Kuppusamy, Uptane, *Securing Software Updates for Automobiles*. https://ssl.engineering.nyu.edu/papers/kuppusamy_escar_16.pdf

[10] Woo et al, *A Practical Wireless Attack on the Connected Car and Security Protocol for In-Vehicle CAN*. IEEE Transactions on Intelligent Transportation Systems (Volume: 16 , Issue: 2 , April 2015)

[11] Gareffa, Peter, *What You Need to Know About Keyless Ignition Systems* https://www.edmunds.com/car-technology/going-keyless.html

[12] Garcia, et. al. *Lock It and Still Lose It—On the (In)Security of Automotive Remote Keyless Entry Systems*. https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_garcia.pdf

[13] Verstegen et al, *Hitag 2 Hell – Brutally Optimizing Guess-and-Determine Attacks*. https://www.usenix.org/system/files/conference/woot18/woot18-paper-verstegen.pdf

[14] Ibrahim et al, *Key Is In The Air:Hacking Remote Keyless Entry Systems*. https://cri-lab.net/wp-content/uploads/2018/08/CarHacking.pdf

[15] Zeng et al, *Car Keyless Entry System Attack*. https://conference.hitb.org/hitbsecconf2017ams/materials/D2T2%20-%20Yingtao%20Zeng,%20Qing%20Yang%20and%20Jun%20Li%20-%20Car%20Keyless%20Entry%20System%20Attacks.pdf

[16] Brewster, Thomas, *Thieves Can Crack Open Audi, BMW, Ford Cars With Simple Keyless Fob Hack* https://www.forbes.com/sites/thomasbrewster/2016/03/21/audi-bmw-ford-thief-car-hacking/#2cccd9514f1e

[17] KeenLab *Free-Fall: Tesla Hacking 2016*. https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf

[18] C. Thuen, *Remote control automobiles*. 2015. https://www.youtube.com/watch?v=-xPjTfD1KZs

[19] He et al., *Deep residual learning for image recognition*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[20] *German Traffic Signs BenchMark Recognition* http://benchmark.ini.rub.de/?section=gtsrb&subsection=news.

[21] Petit et al., *Remote Attacks on Automated Vehicles Sensors: Experiments on Camera and LiDAR*. Black Hat Europe 11 (2015): 2015.

[22] Shin et al., *Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications*. In International Conference on Cryptographic Hardware and Embedded Systems, pp. 445-467. Springer, Cham, 2017.

[23] Yan et al., *Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-driving Vehicle*. DEF CON 24 (2016).

[24] Papernot et al., *Practical black-box attacks against machine learning*. In Proceedings of the 2017 ACM on Asia conference on computer and communications security, pp. 506-519. ACM, 2017.

[25] Lu et al., *No need to worry about adversarial examples in object detection in autonomous vehicles*. arXiv preprint arXiv:1707.03501 (2017).

[26] Eykholt et al., *Robust physical-world attacks on deep learning models*. arXiv preprint arXiv:1707.08945 (2017).

[27] Keen security lab, *Experimental attacks on Tesla's model*. https://keenlab.tencent.com/en/2019/03/29/Tencent-Keen-Security-Lab-Experimental-Security-Research-of-Tesla-Autopilot/

[28] Yuan et al., *Adversarial examples: Attacks and defenses for deep learning*. IEEE transactions on neural networks and learning systems (2019).

[29] Madry et al., *Towards deep learning models resistant to adversarial attacks*. arXiv preprint arXiv:1706.06083 (2017).

[30] Bißmeyer et al., *Assessment of node trustworthiness in vanets using data plausibility checks with particle filters*. In 2012 IEEE Vehicular Networking Conference (VNC), pp. 78-85. IEEE, 2012.

[31] Chow and Tsai, *Securing the visual channel: How my car saw the light and stopped learning*. In 2018 52nd Annual Conference on Information Sciences and Systems (CISS), pp. 1-6. IEEE, 2018.

[32] @0xCharlie (Charlie Miller). "this is true of all car hacking. it's a tough way to hurt someone." 24 May 2018, https://twitter.com/0xcharlie/status/999840403145838592