Securing Vehicle ECUs Update Over The Air

Kevin Daimi
Computer Science and Software Engineering
University of Detroit Mercy
Detroit, USA
email:daimikj@udmercy.edu

Mustafa Saed, Scott Bone, Muhammad Rizwan
HATCI Electronic Systems Development
Hyundai-Kia America Technical Center
Superior Township, USA
email: {msaed, sbone, mrizwan }@hatci.com

***Abstract*—Present-day vehicles involve many Electronic Control Units (ECUs). Future vehicles will have even more ECUs. Currently, the firmware of these ECUs is updated at the dealership when there is a need. The updates are sent to the dealership using electronic media. This process is very time-consuming and lacks the possibility of performing these updates to all vehicles of a certain model in parallel. A future trend by auto manufacturers would be performing these updates over the air. Firmware Over-The-Air (FOTA) will be prone to a variety of attacks. This paper proposes a security architecture for the FOTA updates and discusses how this security architecture is implemented for vehicles at customer locations, dealership sites, and production lines.**

Keywords—ECUs; Firmware; FOTA; Security Architecture; Security Protocol

I. INTRODUCTION

There are a number of bus systems or protocols available in today's in-vehicle networks. Examples of these include Local Interconnect Network (LIN), Controller Area Network (CAN), Media-Oriented System Transport (MOST), and FlexRay. The LIN protocol was introduced to complement the CAN bus. It is a low speed bus and supports various applications including door locks, and seat belts. The CAN bus has a maximum speed of 1 Mbps. Higher speed is supported by MOST, a standard multimedia and infotainment networking in automobiles. FlexRay was designed to be the next-generation and fault-tolerant protocol to support high-bandwidth and safety-critical applications [1]-[3]. Modern-day vehicles typically deploy more than one of these protocols. CAN is currently dominant, and vehicles include two or three CAN buses providing two to three different speeds.

Modern vehicles are equipped with 50-70 embedded electronic control units (ECUs), which supervise a great deal of their functionality [4]. This functionality has a broad set of tasks including overseeing door looks, climate, sunroof, body systems, transmission, advanced safety and collision avoidance systems, and pressure monitoring systems. On each ECU, a specialized and independent firmware is executed, and upgraded versions of the firmware are introduced as errors are identified or new functionality is added [5]. ECUs receive signals sent by sensors located at various parts and in different components of the vehicle. Based on these signals, ECUs control various key units in the vehicle [6].

The entire network, including the buses and the ECUs, need to be protected against security attacks. Various analyses of the buses, especially the CAN bus; have revealed various vulnerabilities in the available in-vehicle network protocols [7] [8]. The in-vehicle networks connecting the ECUs to the buses are not deemed closed network but an open network attracting many cyber-attacks. The fact that some ECUs, such as the immobilizer, are equipped with specific security capabilities does not rule out the reality that the security requirements; confidentiality, authenticity, availability, integrity, and nonrepudiation are not satisfied [9]. A security analysis, which was carried out recently on a production vehicle, showed that an adversary might tamper with the brakes when the car is running once access to the in-vehicle network via the Bluetooth is assured [10] [11]. Other attacks are made possibly through the On Board diagnostics (OBD-II) port. Compromising one ECU allows the attacker full access and control of all other ECUs since the in-vehicle network is fully connected [12]. Security needs to be considered in the early stages of the development process of vehicle electronics systems by demanding firmware standards that avoid firmware defects giving rise to cyberattacks, and by incorporating security mechanisms, such as authentication and cryptology, to enable the verification of the identity of the sender to prevent bogus and potentially harmful messages to be replayed/transmitted across the communication network [13].

All ECUs' firmware needs to be updated. Updates include urgent firmware fixes through recalls, feature upgrade, security patches, and customer complaints fixes. It is also possible to replace the whole firmware with a brand new one. Currently, all firmware updates are performed at the dealership. When the work is completed, the technician checks the targeted ECU to ensure it is functioning correctly. Assessment of the traditional approach signified that such updates are time and resource consuming, result in higher cost of labor and customer dissatisfaction, and prevent parallel updates as a result of physical equipment connection [14]. A future trend in auto industry is to adopt Firmware Over-The-Air (FOTA) updates. FOTA refers to the process of wireless firmware transfer to the ECUs [15]. Mobile phone companies have been successfully updating their software Over-The-Air. It is anticipated that FOTA will gain wide acceptance in automotive industry following the great success in mobile phone industry. With FOTA, updates will be performed at the customer (any) location and not at the dealership site. This will mean fast, effective, and cost efficient approach of firmware updating.

Firmware Over-The-Air (FOTA) definitely implies wireless communications. This will widely open the door for many

 ISBN: 978-1-61208-473-2

cyberattacks. Many of the current wireless security attacks will take advantage of the FOTA approach. The consequences will be disastrous as safety is involved. Therefore, there should be a serious and imminent move by auto industry to protect their vehicles' ECUs against all the possible attacks. To cope with such vital attacks, few researchers introduced their analysis and possible solutions to such challenges. Phung and Nilsson [16] proposed a threat model for the vehicle software architecture to pinpoint possible threats and suggested countermeasures for some improper conduct caused by malicious or poorly designed applications. Their approach is based on modifying the application at the wireless gateway of the vehicle before installation to guarantee safety and security of the vehicle through spotting likely attacks. The approach relied on the reference monitor component to decide whether to grant requests for resources based on security policy [17]. The vast majority of attacks take place when the software is being transferred from the manufacturer site and before reaching the gateway. In addition, setting a security policy for the arrived software is hard to achieve when many vendors provide different software and firmware to auto manufacturers.

Idrees et al [18] proposed on-board security architecture to facilitate the firmware update processes using both hardware and software modules. This was followed by a protocol to demonstration how their security architecture was employed to accomplish secure firmware updates for electronic control units (ECUs). Their approach is mainly based on a hardware security process to safeguard critical parts, such as secure key storage and the functioning of the cryptographic algorithms, of their architecture during the firmware update. The introduction of hardware is definitely valuable. However, from a security point of view, this will introduce the additional problem of hardware attacks in addition to software attacks. Furthermore, the paper indicated the use of a public key but no private key was specified.

Miller and Valasek [19] introduced possible attacks on various vehicles through the CAN bus and the Electronic Control Units (ECUs). They investigated a remote attack on an unaltered vehicle model and similar vehicles that causes a physical control of some parts of the vehicle. They hoped that their work on this remote attack will help in enhancing the security of connected vehicles in the future by avoiding the vulnerabilities that result in compromising the CAN and ECUs.

This paper presents security architecture for Over-The-Air update of ECUs. It covers the update of firmware at the production site, dealer site, and customer location. A security protocol to implement this architecture is introduced. Both symmetric and asymmetric cryptology will be used. The suggested architecture and protocol will ensure that the security requirements; confidentiality, integrity, authentication, and non-repudiation will be satisfied. The remainder of the paper is organized as follows: Section II will discuss the FOTA security architecture. Section III will introduce the implementation of the architecture via a security protocol. The approach is extended in Section IV to cover application software in addition to firmware. The paper is concluded in Section V.

II. FOTA SECURITY ARCHITECTURE

The FOTA security architecture is depicted in Fig. 1, which uses two different colors to highlight the connections between the components. It is composed of seven components: Certificate Authority (CA), Firmware Repository (FR), Firmware Distribution Authority (FDA), Vendor Firmware Packaging Manager (VFPM), Production Site Manager (PSM), Dealer Stock Manager (DSM), and Master ECU (MECU). All the components are connected to both CA and FDA. In addition, FDA is connected to CA.

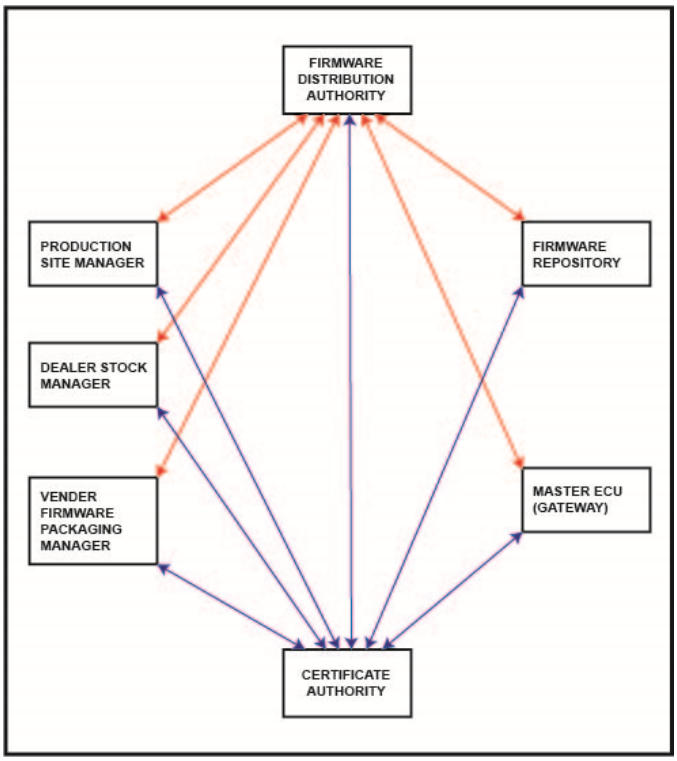

Figure 1. FOTA security architecture

The Certificate Authority (CA) is in charge of issuing certificates to all components including the Firmware Distribution Authority (FDA). The CA can be part of the manufacturing site or an independent party. The most important component is the Firmware Distribution Authority. The FDA is responsible for firmware updates of all vehicles at the dealership, production lines, and customer locations (garages, parking lots, streets, etc.). It receives the packaged firmware updates from vendors and stores them in the firmware repository prior to sending them to vehicles. Further responsibilities include ensuring all vehicles of that type and model have been updated, and the updated ECUs are functioning properly. Updates include improving the ECU's functionality, firmware bug fixes, and brand new firmware to completely replace the old version. It is assumed that the manufacturer site has the capability to verify the update is

 ISBN: 978-1-61208-473-2

functioning correctly. The FDA is also in charge of issuing the session keys and the Message Authentication Code (MAC) keys.

The Firmware Repository (FR) is the firmware storage at the manufacturer's site. It is in charge of storing the firmware received from the FDA and providing the FDA with the needed firmware when requested. For each firmware, additional information including update version number, update type (full, bug fix, and enhancement), ECU type, date it was received, size of updates in bytes, vehicle model, vendor ID, and checksum are stored. The Vendor Firmware Packaging Manager (VFPM) is responsible for preparing the firmware update and securely forwarding it to the Firmware Distribution Authority at the manufacturer's site to be stored in the Firmware Repository.

The Production Site Manager (PSM) is charge of updating all the vehicles in the production lines before sending them to dealerships. Updating all the used and new cars at the dealership is the responsibility of the Dealer Stock Manager (DSM).

The Master ECU (MECU) plays a major role in the firmware update. It is a gateway equipped with the needed hardware, software, and memory. MECU may be closed to disable interfaces like Universal serial Bus (USB), Universal Asynchronous Receiver/Transmitter (UART), and Joint Test Action Group (JTAG). For this purpose, the Telematics Control Unit (TCU) can also be used. The MECU receives the firmware updates from the Firmware Distribution Authority and updates the ECUs in question in addition to updating its own. It warns the FDA when the firmware update is completed. Note that both DSM and PSM communicate with MECU of their vehicles. They behave like brokers. The direct secure communication between the MECU of the customer and the FDA will be discussed in the next section. The behavior of the MECUs connected to the Dealership Stock Manger and Production Site Manager is similar. Therefore, it will be explained once. To elucidate the participating parties in the architecture, Table 1 should be relied on.

TABLE I. PARTICIPATING PARTIES

Symbol	Role
CA	Certificate Authority
FDA	Firmware Distribution Authority
FR	Firmware Repository
DSM	Dealership Stock Manager
PSM	Production Site Manager
VFPM	Vendor Firmware Packaging Manager
MECU	Master ECU
SDM	Software Download Manager
SCM	Software Charge Manager
RDM	Remote Diagnosis Manager

III. SECURING THE FOTA

Securing the FOTA updates will include securing the communication between the seven components of the above security architecture. For this purpose, cryptographic protocols are used. The protocol notations are introduced in Table 2 to illustrate the role they play in the protocol.

TABLE II. PROTOCOL NOTATIONS

Symbol	Meaning
PU_{FDA}, PR_{FDA}	Public & private key of FDA
PU_{FR} , PR_{FR}	Public & private key of FR
PU_{CA} , PR_{CA}	Public & private key of CA
PU_{DSM}, PR_{DSM}	Public & private key of DSM
PU_{PSM}, PR_{PSM}	Public & private key of PSM
PU_{VFPM}, PR_{VFPM}	Public & private key of VFPM
PU_{MECU}, PR_{MECU}	Public & private key of MECU
KS_{FR}	Session Key shared between FDA and FR
KS_{DSM}	Session Key shared between FDA and DSM
KS_{PSM}	Session Key shared between FDA and PSM
KS_{VFPM}	Session Key shared between FDA and VFPM
KS_{MECU}	Session Key shared between FDA and MECU
KM_{FR}	MAC Key shared between FDA and FR
KM_{DSM}	MAC Key shared between FDA and DSM
KM_{PSM}	MAC Key shared between FDA and PSM
KM_{VFPM}	MAC Key shared between FDA and VFPM
KM_{MECU}	MAC Key shared between FDA and MECU
$C(KM_X, F)$	MAC function
X	Refers to FDA, FR, DSM, PSM, VFPM, or MECU
T_{Si} i=1-12	Time stamps
T_1	Time stamp
T_2	Certificate validity period
N_X	Nonce for X
CR_X	Certificate of X
ID_U	Update ID
F	Firmware
ID_{ECU}	ID of the ECU to be updated
ID_V	Vendor ID
$H(B)$, $H(I)$	Hash function of bug and improvement messages
L	List of vehicles VIN numbers
E	Encryption
VIN	Vehicle identification number

A. Certificate Authority

The Certificate Authority (CA) is in charge of issuing certificates to all the other components. The CA shares its public key (PU_{CA}) with the components. A component requests its certificate by sending its public key (PU_X), its ID (ID_X) and a nonce (N_X) all encrypted with the public key of the CA. Here, X is used to denote any of the six components. Upon receiving the request, the CA decrypts it with its private key (PR_{CA}) and sends X its certificate encrypted with PR_{CA}. The certificate of the component (CR_X) will have the format below:

$$CR_X = E\,[PR_{CA}, (PU_X \,||\, ID_X \,||\, T_1 \,||\, T_2)]$$

The certificate and the nonce will be concatenated and then encrypted with the public key (PU_X) of the requesting component and sent to the component. Assuming the private key of the component (PR_X) is not compromised, this will assure no one but the requester can access the certificate.

$$CA \rightarrow X: E [PU_X, CR_X || N_X]$$

In addition to the public key and ID, the certificates include a timestamp, T_1, and a certificate validity period (expiration date), T_2. Both T1 and N_X are attached for additional assurance that the message involving the certificate is not a replay. The parties (components) receiving this message will decrypt it with its private key, verify T_1 and N_X and get its certificate (CR_X). Note that X will be replaced with FDA, FR, VFPM, PSM, DSM, or MECU in the next sections to denote the different components.

B. Firmware Distribution Authority

The Firmware Distribution Authority exchanges its certificate (CR_{FDA}) with all other components. It will decrypt the received certificates to obtain the public key and ID of each component.

FDA creates the session keys; KS_{FR}, KS_{VFPM}, KS_{PSM}, KS_{DSM}, and $KS_{MECU,}$ to be shared with each component, encrypts them with the corresponding public keys; PU_{FR}, PU_{VFPM}, PU_{PSM}, PU_{DSM}, and PU_{MECU}, and sends them to FR, VFPM, PSM, DSM, and MECM respectively. In a similar fashion, the FDA creates the MAC keys KM_{FR}, KM_{VFPM}, KM_{PSM}, KM_{DSM}, and KM_{MECU}, and sends them to the respective components.

When the Vendor Firmware Packaging Manager informs the FDA about the packaging of a firmware update via a secret message, FDA acknowledges the message. This step is then followed by the actual transfer of the firmware update. Once the update is received, the FDA sends a notification message to the Firmware Repository. After this message is acknowledged, the firmware update is forwarded to the FR as follows:

$$X_1 = E [PR_{FDA}, C (KM_{FR}, F) || Info || ID_U || T_{S1}]$$
$$FDA \rightarrow FR: E [KS_{FR}, F || E (PU_{FR}, X_1)]$$

The term C (KM_{FR}, F) refers to the MAC of the firmware; F. Info represents additional information, such as update version, update ID (ID_U), date received, ECU ID, vendor ID, vendor name, and type of update. T_{S1} is the time stamp. Note that both public key and symmetric key cryptology is used. Public key cryptology is used for small messages because it is slow, and the symmetric cryptology is used with possibly large messages, *F* in this case.

The MAC is signed with the private key (PR_{FDA}) of FDA. The expression X_1 is encrypted with the public key (PU_{FR}) of FR to provide confidentiality as only FR can decrypt this message with its private key (PR_{FR}). The MAC, C (KM_{FR}, F) provides the message authentication. In addition, the encryption with the symmetric key, KS_{FR}, designates further confidentiality and authentication.

Similar messages will be sent to the other parties with the exception of *info*.

$$X_2 = E [PR_{FDA}, C (KM_{VFPM}, F) || ID_{ECU} || ID_U || T_{S2}]$$
$$FDA \rightarrow VFPM: E [KS_{VFPM}, F || E (PU_{VFPM}, X_2)]$$

$$X_3 = E [PR_{FDA}, C (KM_{PSM}, F) || ID_{ECU} || ID_U || T_{S3}]$$
$$FDA \rightarrow PSM: E [KS_{PSM}, F || E (PU_{PSM}, X_3)]$$

$$X_4 = E [PR_{FDA}, C (KM_{DSM}, F) || ID_{ECU} || ID_U || T_{S4}]$$
$$FDA \rightarrow DSM: E [KS_{DSM}, F || E (PU_{DSM}, X_4)]$$

$$X_5 = E [PR_{FDA}, C (KM_{MECU}, F) || ID_{ECU} || ID_U || T_{S5}]$$
$$FDA \rightarrow MECU: E [KS_{MECU}, F || E (PU_{MECU}, X_5)]$$

C. Firmware Repository

After performing the required decryptions on the received message, E [KS_{FR}, F || E (PU_{FR}, X_1)], calculating and verifying the MAC, and ensuring T_{S1} is current, the FR stores the firmware, F, together with *Info* and any other data needed for indexing. Upon receiving a request from the FDA, it retrieves the firmware in question and sends it to FDA within the following message:

$$X_6 = E [PR_{FR}, C (KM_{FR}, F) || ID_{ECU} || ID_U || T_{S6}]$$
$$FR \rightarrow FDA: E [KS_{FR}, F || E (PU_{FDA}, X_6)]$$

The FR stores the date the request was received and the date the firmware, F, was sent for auditing purposes.

D. Vendor Firmware Packaging Manager

Auto manufacturers deal with several vendors. The security architecture above contains only one box for the Vendor. Therefore, the message sent here will include the vendor ID.

$$X_7 = E [PR_{VFPM}, C (KM_{VFPM}, F) || ID_V || || ID_{ECU} || ID_U || T_{S7}]$$
$$VFPM \rightarrow FDA: E [KS_{VFPM}, F || E (PU_{FDA}, X_7)]$$

Note that ID_V is the ID of the vender. ID_{ECU} represents the ID of the affected ECU. The message above is preceded by a notification message (of new update) sent and an acknowledgement message received.

In addition to the vendor initiating updates and packaging them, the FDA can request updates when a bug is discovered or an improvement is needed. A message containing the bug or the improvement will be sent to that specific dealer:

$$X_8 = E [PU_{VFPM}, B || E (PR_{FDA}, H (B) || ID_V || ID_{ECU} || T_{S8})]$$
$$FDA \rightarrow VFPM: X_8$$

Or,

 ISBN: 978-1-61208-473-2

$$X_9 = E\,[PU_{VFPM}, I \,||\, E\,(PR_{FDA}, H\,(I) \,||\, ID_V \,||\, ID_{ECU} \,||\, T_{S9})]$$
$$FDA \rightarrow VFPM: X_9$$

Here, B is the bug detail, I the improvement detail, H (B) and H (I) represent the hash function of B and I respectively, and ID_{ECU} is the ID of the ECU that has a bug or needs improvement. The VFPM will decrypt the message, calculate the hash and verifies it is the same as the received hash. When the verification is successful, the update process will take place. Note that only public key cryptology was used here because the message is not large.

E. Production Site Manager

The PSM is responsible for the updates of the ECUs in all the vehicles in the production lines. After receiving the message X_3 from the FDA and executing the required decryptions and verifications, PSM has to send the firmware to the MECU of the vehicles in that line. For this purpose, it will act like the FDA and communicate similar encrypted messages with the MECU of each vehicle. Once the update is received by the MECUs, the update will be implemented in parallel as they all received it. The approach used by the MECUs of production line's vehicles is the same as in (G) below. The MECUs will inform the PSM when the updates are completed for that update ID (ID_U). The PSM will then inform the FDA of all the vehicles that have their firmware updated by sending a message containing the list of vehicles VIN numbers, L. This is needed for ensuring that all vehicles are updated and for reporting purposes.

$$X_{10} = E\,[PR_{PSM}, C\,(KM_{PSM}, L) \,||\, U_{ID} \,||\, T_{S10}]$$
$$PSM \rightarrow FDA: E\,[KS_{PSM}, L \,||\, E\,(PU_{FDA}, X_{10})]$$

F. Dealer Stock Manager

The firmware updates at the dealership site are controlled by the DSM. The task of the DSM is similar to that of the PSM. The work will be completed by the MECUs in parallel here too.

G. Master ECU

The Master ECU is responsible for managing the updates of the firmware of the ECUs. For customers' vehicles, the MECU will communicate with the driver through the vehicle screen or via email to warn about a new update and request the vehicle to be turned off, as soon it is possible. Vehicles in the production lines and at the dealerships are assumed to be not running since they are under control.

The MECU will fulfill the required decryptions and verification of the MAC. Once successful, it will extract the firmware F, ID_{ECU}, and ID_U. The MECU will then communicate with the desired ECU based on the ID_{ECU} to start the updating process. For busses where there is limitation on the size of the data packets, such as eight bytes for the CAN bus, the protocol can use the Counter Mode (CTR) to divide the plain/cipher text (firmware) to be encrypted/decrypted into blocks of eight bytes each.

Upon completing the update using any secure process, the MECU of customer's vehicle will inform the FDA by sending the following message:

$$X_{11} = E\,[PR_{MECU}, \,||\, ID_{ECU} \,||\, ID_U \,||\, VIN \,||\, T_{S11}]$$
$$MECU \rightarrow FDA: E\,(PU_{FDA}, X_{11}]$$

For firmware update at the dealership and production sites, the MECU will send a similar message to DSM and PSM respectively. As, mentioned above, DSM and PSM will send a list of all the vehicles' VINs that are updated by collecting the information from all the MECUs within their site.

IV. EXTENDING THE SECURITY ARCHITECTURE

The above security architecture can be enhanced to deal with software apps in addition to firmware. In general, software is not free. To accommodate software download and software update, two components need to be added to the architecture; Software Download Manager (SDM) and Software Charge Manager (SCM). The latter is needed when the software is not free. Both SDM and SCM will be connected to FDA and CA. All the security measures used above will still apply here. With these two new components, the auto manufacturer will be able to implement Software On-The-Air (SOTA) in addition to FOTA.

An important component that could further be added in the future is the Remote Diagnosis Manager (RDM). Software companies, such as Microsoft, can remotely connect to our computers with our permission to diagnose problems. Upon customer request, it is anticipated that RDM will connect to the vehicle and diagnose problems. Once the problem is diagnosed, a message requesting firmware update for the particular ECUs will be sent to FDA. Another task that the RDM will be responsible for would be testing the update to certify the ECU is functioning as expected. The FDA will initiate the needed firmware update following the cryptographic protocol above. Certainly, RDM will be connected to both FDA and CA.

In the aforementioned protocol, it was assumed that one MECU will take care of updating all the ECUs. It is suggested adding more MECUs and dividing the ECUs among them. In other words, each MECU will be in charge of some ECUs. The added MECUs can play a backup role too in case an MECU is not functioning. Furthermore, if an MECU is compromised, it will not impact other MECUs (other ECUs).

A further extension will be replacing the Master ECU with the Telematics Control Unit (TCU). The TCU is a small computer that listens in on the communications of other electronic systems (ECUs) in the vehicle, then construes and disseminates that information as necessary. The TCU connects to the external server. This server could well be the

 ISBN: 978-1-61208-473-2

FDA. All that is needed is to make the TCU more powerful in terms of processing and memory.

V. CONCLUSION AND FUTURE WORK

With the FOTA gaining increasing popularity among auto manufacturers as a future trend, security measures need to be enforced to ensure the firmware travelling on the air will not be attacked. To account for possible security attacks, this paper presented a security architecture and protocol to protect the updating of firmware of various ECUs at the customer location, the dealership site, and the production lines. Both symmetric and asymmetric cryptology techniques were adopted. The suggested security architecture and protocol can be further extended to include Software On-The-Air (SOTA). Future work will also concentrate on the most suitable algorithms for symmetric and asymmetric cryptography, MAC and hash functions, and the length of the various keys.

REFERENCES

[1] Freescale Semiconductors, "In-Vehicle Networking," https://cache.freescale.com/files/microcontrollers/doc/brochure/BRINVEHICLENET.pdf, 2006, pp. 1-11, [retrieved: March, 2016].

[2] On Semiconductor, "Basics of In-Vehicle Networking (INV) Protocols," http://www.onsemi.com/pub_link/Collateral/TND6015-D.PDF, pp. 1-27, [retrieved: March, 2016].

[3] K. Parnell, "Put the Right Bus in Your Car," Xcell Journal, Available: http://www.rpi.edu/dept/ecse/mps/xc_autobus48(CAN).pdf, [retrieved: March, 2016].

[4] D. K. Nilsson, P. H. Phung, U. E. Larson, "Vehicle ECU Classification Based on Safety-Security Characteristics," in Proc. the 13th International Conference on Road Tarnsport Information and Control (RTIC'08), Manchester, England, UK, 2008, pp. 1-7.

[5] D. K. Nilsson, and U. E. Larson, "A Defense-in-Depth Approach to Securing the Wireless Vehicle Infrastructure," Journal of Networks, vol. 4, no. 7, 2009, pp. 552-564.

[6] National Instruments, "ECU Designing and Testing Using National Instruments Products," http://www.ni.com/white-paper/3312/en, 2009, [retrieved: March, 2016].

[7] T. Hoppe, S. Kiltz, J. Dittmann, "Automotive IT-Security as a Challenge: Basic Attacks from the Black Box Perspective on the Example of Privacy Threats," Computer Safety, Reliability, and Security, 2009, pp. 145-158.

[8] M. Wolf, A. Weimerskirch, C. Paar, "Security in Automotive Bus Systems," in Proc. the 2nd Embedded Security in Cars Workshop (ESCAR 2004), Bochum, Germany, 2004, pp. 11-12.

[9] I. Studnia, V. Nicomette, E. Alata, Y. Deswarte, M. Kaaniche, Y. Laarouchi, "Survey on Security Threats and Protection Mechanisms in Embedded Automotive Networks," in Proc. the 43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop (DSN-W), Budapest, Hungary, 2013, pp. 1-12.

[10] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, "Comprehensive Experimental Analyses of Automotive Attack Surfaces," in Proc. the 20th USENIX Symposium on Security (SEC'11), San Francisco, CA, USA, 2011, pp. 77-92.

[11] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. Mccoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, "Experimental Security Analysis of a Modern Automobile," in Proc. IEEE Symposium on Security and Privacy (SP), Oakland, CA, USA, 2010, pp. 447–462.

[12] F. Sagstetter, M. Lukasiewycz, S. Steinhorst, M. Wolf, A. Bouard, W. Harris, S. Jha, T. Peyrin, A. Poschmann, and S. Chakraborty, "Security Challenges in Automotive Hardware/Software Architecture Design," in Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2013, pp. 458-463.

[13] C. Lin, and A. Sangiovanni-Vincentelli, "Cyber-Security for the Controller Area Network (CAN) Communication Protocol," in Proc. International Conference on Cyber Security, Washington, DC, USA, 2012, pp. 1-7.

[14] Red Bend Software, "Updating Car ECUs Over-The-Air (FOTA)," www.redbend.com/data/upl/whitepapers/red_bend_update_car_ecu.pdf, 2011, pp. 1-14, [retrieved: March, 2016].

[15] Novero, "Automatic Over-The-Air Software Update," 2012, http://novero.com/wp-content/uploads/2014/12/Novero_Automotive_Software_Over_the_Air_Handout1.pdf, [retrieved: March 2016].

[16] P. H. Phung, and D. K. Nilsson, "A Model for Safe and Secure Execution of Downloaded Vehicle Applications," in Proc. Road Transport Information and Control Conference (RTIC 2010), London, UK, 2010, pp. 1-6.

[17] J. P. Anderson, "Computer Security Technology Planning Study," Deputy for Command and Management System, USA, Tech. Rep., 1972, http://csrc.nist.gov/publications/history/ande72.pdf, [retrieved: March, 2016].

[18] M. S. Idrees, H. Schweppe, Y. Roudier, M. Wolf, D. Scheuermann, O. Henniger, "Secure Automotive On-Board Protocols: A Case of Over-The-Air Firmware Updates," in Proc. the 3rd International Workshop Nets4Cars/Nets4Trains, Oberpfaffenhofen, Germany, 2011, pp. 224-238.

[19] C. Miller and C. Valasek, "Remote Exploitation of an Unaltered Passenger Vehicle," 2015, http://illmatics.com/Remote%20Car%20Hacking.pdf, [retrieved: March, 2016].

 ISBN: 978-1-61208-473-2