Securing Vehicle's Electronic Control Units

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

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

Abstract— **Electronic Control Units (ECUs) are essential for controlling many functions and systems in current and future vehicles. Modern vehicles incorporate over seventy ECUs. Those ECUs are vulnerable to security attacks. A number of these attacks can be fatal and can result in casualties. Undoubtedly, there is a critical need for protecting the ECUs infrastructure. This paper proposes an approach to secure vehicle's ECUs based on a grouping principle. Four groups are introduced. Each group is controlled by a Master ECU, and the Master ECUs are controlled by a Super Master ECU. Public key cryptology is adopted. Furthermore, the possibility of applying symmetric key cryptology, Elliptic Curve Cryptology (ECC), and One-Time Pad are investigated.**

Keywords— ECUs; Security Architecture; Security Protocols; Security Requirements

I. INTRODUCTION

Modern vehicles deploy a number of busses in their networks. Among these are the Local Interconnect Network (LIN), Controller Area Network (CAN), Media-Oriented System Transport (MOST), and FlexRay. LIN is used for the lowest data-rate functions, such as door locks, climate control, and mirror control. CAN is suitable for medium speed applications including body systems, engine management, and transmission. MOST lends itself to the high-speed data rates, and therefore, it is convenient for multimedia and entertainment. Finally, the FlexRay is suitable for safety-critical applications, such as steer-by-wire, stability control, and brake-by-wire. Connected to these buses are various Electronic Control Units (ECUs). ECUs are embedded systems controlling one or more of the vehicle's systems and subsystems. They play a crucial role in controlling many functions in vehicles. ECUs are made up of both hardware and firmware. They are named and differentiated based on what they are used for. For example, the Engine Control Module (ECM) controls various engine functions such as fuel injection, ignition timing and idle speed control system, the Electronic Brake Control Module (EBCM) is used in the anti-lock braking system (ABS), and the Powertrain Control Module (PCM) monitors and controls speed control, A/C, and automatic transmission [1]-[9]. It is critical to protect these ECUs for proper functioning of the vehicle and for safety purposes.

Nish [10] introduced a number of security issues in modern automotive systems. The communication of Tire Pressure Monitoring System (TPMS) with its sensor is unsecure and missing encryption and signature in the data protocol. As a result, the tire pressure warning lights can be turned on and off causing the driver to worry about the tire pressure when there is nothing wrong with it. Another issue regards the keyless entry systems. The passive keyless entry in modern cars can be subject to relay attack by intercepting and relaying the radio signal from the smart keys to the cars. The attackers can break into and steal the valuables left in the vehicle. Further issue that has a safety nature involves the On-Board Diagnostic port (OBD-II). This interface provides direct access to the vehicle for diagnosing and updating the firmware of ECUs. By connecting to this port through a USB or WiFi, some software on the attacking computer can re-program the ECUs causing considerable and possibly fatal damage.

Othmane, Weffers, Mohamad, and Wolf [11] proposed a taxonomy for vehicle security and privacy aspects. They stressed the security of communication links, data validity, devices security, identity, and access control. They attempted to provide an initial repository of threats to vehicle network. Security threats and the possibility of attacks can arise when drivers try to control the lights, windshields, wipers, air flow and the heater of their vehicles through Bluetooth or exercise remote starting or unlock doors using their PDA [12]. Any attack on the Bluetooth or the PDA will impact security of the vehicle and drivers safety. A vehicle's ECUs communicate through the in-vehicle network and it communicates with Service Providers through cellular network [13]. All the possible attacks on cellular networks will find their way to the vehicle and can impact the ECUs.

A security approach to protect the CAN protocol from masquerade and replay attacks was proposed by Lin and Sangiovanni-Vincentelli [14]. They provided a software-only solution with no additional hardware needed. The focus was on run-time authentication after ignition key is turned on and the security secret keys have been distributed to the ECUs. Han, Potluri, and Shin [15] introduced a security architecture to deal with the potential security attacks infiltrated by mobile devices, such as smart phones and tablets, interfacing with the vehicle to send/receive information to/from the vehicle. Three parties were adopted, the user device, the gateway, and the ECUs. Patsakis, Dellios, and Bouroche [16] stressed that the standards for in-vehicle security are distant from deploying long-established security policies and procedures. They analyzed the current auto industry policies and procedures with regards to security, and highlighted a number of vulnerabilities. In an attempt to overcome these vulnerabilities, they introduced a security

 ISBN: 978-1-61208-482-4

architecture to support mutual authentications of ECUs and various access rights for users.

Several attempts have focused on grouping ECUs for various purposes. In one of these attempts, ECUs were divided into four groups; Powertrain Master Control Unit, Chassis Master Control Unit, Cabin Master Control Unit, and Infotainment Master Control Unit [17]. Nilsson, Phung, and Larson [18] indicated five categories: Powertrain, Vehicle Safety, Comfort, Infotainment, and Telematics. The groups; Comfort Systems, Body Control, Real Time Systems, and Safety-Critical systems were suggested by Seo, Kim, Hwang, Kwon, and Jeon [8]. Powertrain Gateway, Body and Comfort Gateway, Chassis Gateway, and Infotainment Gateway were advocated in [4]. ECUs were also grouped as Powertrain, Safety, Comfort, and Infotainment and Telematics [2]. A further approach adopted by Cho, Bae, Chu, and Suh [19] proposed User-Friendly Diagnostic Unit, Engine-Transmission-Chassis-Body Unit, Safety Unit, and Telematics-Information-Entertainment Unit.

This paper proposes a security architecture for secure transmission of ECUs' messages. The ECUs are divided among four groups. Each group is controlled by a Master ECU (MECU). The resulting four MECUs are supervised by the Super Master ECU (SMECU). Public Key cryptology is adopted. Furthermore, the paper investigates the possibility of applying symmetric key cryptology, Elliptic Curve Cryptology, and stream ciphers. The security requirements are examined. The remainder of the paper is organized as follows: Section II introduces the proposed security architecture. Securing the ECUs using public key cryptology is dealt with in section III. Other possible approaches for securing a vehicle's ECUs are briefly introduced in section IV. These include symmetric key cryptology, Elliptic Curve Cryptology and stream ciphers. Finally, the paper is concluded in section V.

II. PROPOSED SECURITY ARCHITECTURE

In in-vehicle network, buses have ECUs connected to them. Three busses are shown in Figure 1 above; high speed CAN (CAN-HS), medium speed CAN (CAN_MS) and a LIN bus. To these buses various ECUs are connected. ECUs broadcast messages. In other words, messages are received by all ECUs, but only acted upon if the message concerns the receiving ECU. The Body Control Module (BCM) and the Instrument Cluster (IC) are connect to both buses; CAN-MS and CAN-HS. These will act as gateways to gate the messages received from one bus to the ECUs connected to the other bus. Table 1 provides the notations used in the hypothetical in-vehicle network.

The security architecture used in this paper is based on the principle of grouping ECUs. The grouping could be based on any subdivision approach. For example, ECUs may be grouped based on their location, functionality, or collaboration. The number of groups is not limited. In Figure 2 below, the ECUs are distributed among four groups of Master ECUs, $MECU_1$, $MECU_2$, $MECU_3$, and $MECU_4$. A number of ECUs are attached to each Master ECU. MECUs do not necessarily contain the same number of ECUs. For this reason, the subscript of the last ECU in each group has different letters. In other words, the use of one subscript letter was avoided to indicate possibly different number of ECUs. There is no direct connection between the ECUs of each group with the other groups.

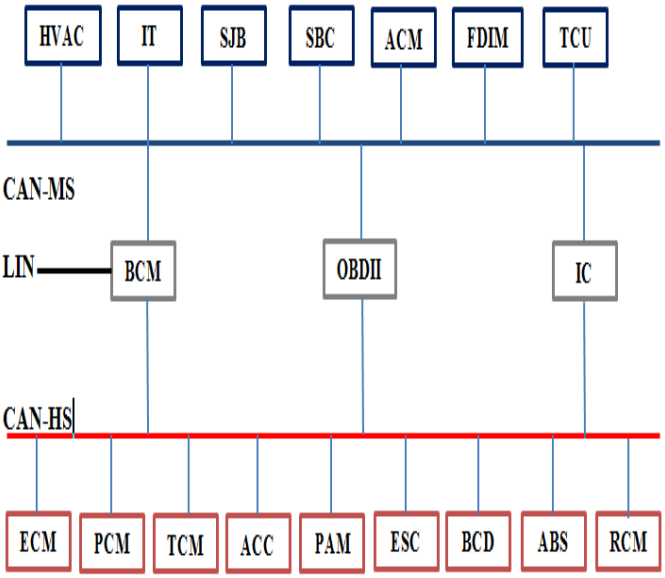

Figure 1. Hypothetical In-Vehicle Network

TABLE I
NOTATIONS USED IN IN-VEHICLE NETWORK

Symbol	Role
ECM	Engine Control Module
PCM	Powertrain Control Module
TCM	Transmission Control Module
ACC	Adaptive Cruise Control
PAM	Parking Aid Module
ESC	Electronic Stability Control
BCD	Blind Spot Detective
ABS	Anti-Lock Brake System Module
IC	Instrument Cluster
BCM	Body Control Module
HVAC	Heat, Ventilation, and Air Conditioning System
IT	Intrusion Detection
SJB	Smart Junction Box
SBC	Seat Belt Control
ACM	Audio Control Module
FDIM	Front Display Module
TCU	Telematics Control Unit
OBD-II	On-board Diagnostic System II

The four master ECUs are connected to the Super Master ECU. The SMECU is the heart of the security architecture. It is the only component connected to the outside word through the security architecture for manufacturer-vehicle communication, which secures various areas, such as Firmware On-The-Air (FOTA), Software On-The-Air (SOTA), and on-board diagnostics. Therefore, SMECU will

 ISBN: 978-1-61208-482-4

be protected by that security architecture, which is beyond the scope of this paper.

The SMECU manages the security of the four MECUs. When a message is broadcasted, it will not reach all the ECUs as shown in Figure 1 above. Only the MECU that controls the broadcasting ECU and the ECUs of the same group will receive it. The MECU of that group will then forward it to the SMECU. To broadcast to the ECUs of the other groups, the SMECU will decide which MECU will receive this message, by checking the message ID. Once received by the MECU, it will broadcast it to its members. By observing the message ID, the individual ECUs will decide to either ignore the message or act upon it. The security approach adopted by the proposed architecture does not allow any direct communication between the MECUs. This will prevent threats to the ECUs of one group from propagating to the ECUs of other groups.

The Super Master ECU manages key creation and distribution for the four MECUs. The individual MECUs manage key creation and distribution for their members (ECUs). The broadcasted messages are short. This implies that public key cryptology is appropriate here. However, the symmetric key cryptology can also be used.

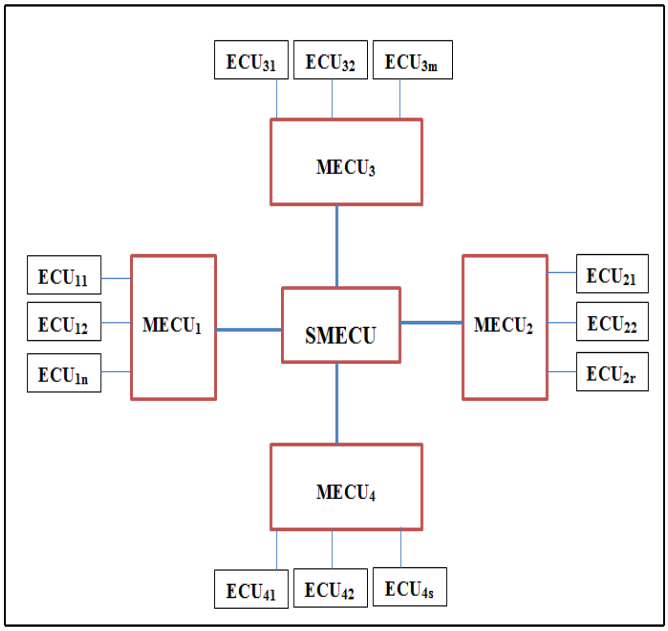

Figure 2. The Proposed Security Architecture

III. ECUs SECURITY

Due to bus frame limitation, an ECU message (payload) will be broken down into three messages. Each frame will contain the ECU ID, ECU_{ID}, Message ID, MSG_{ID}, and the Message, MSG. Messages not exceeding the size of the framework will be sent in one communication. To facilitate following the security protocols, Table 2 provides the protocol notations.

A. Initialization

All nodes will have their initial public and private keys pre-installed at manufacturing time. In addition, each MECU will have the public keys and IDs of its members and its members will have their MECU's public key and ID pre-installed. Further pre-installation include a shared secret value S_i between MECU i and its members, and a shared secret value Vj between each MECU and the Super MECU. Finally, the public keys and IDs of the four MECUs will be pre-stored at the SMECU's memory and the public key and ID of SMECU will be stored in each of the four MECUs.

TABLE II PROTOCOL NOTATIONS

Symbol	Meaning
MSG	Message
SMECU	Super Master ECU
$MECU_i$, *i= 1- 4*	Master ECU i
ECU_{ij}	ECU j of MECU i
PU_{ECU}, PR_{ECU}	Public & private key of ECU
PU_{MECU}, PR_{MECU}	Public & private key of MECU
PU_{SMECU}, PR_{SMECU}	Public & private key of SMECU
ID_{ECU}	ID of ECU
ID_{MECU}	ID of MECU
ID_{SMECU}	ID of SMECU
S_i	Secret value shared between MECU i & its ECUs
V_j	Secret value shared between MECU j & SMECU
K_{ECU}	Key shared between ECU and MECU
K_{MECU}	Key shared between MECU and SMECU
KM_{ECU}	MAC Key shared between ECU and MECU
KM_{MECU}	MAC Key shared between MECU and SMECU
$C(KM_{ECU}, MSG)$	MAC function for ECU and MECU
$C(KM_{MECU}, MSG)$	MAC function for MECU and SMECU
T	Time stamp
-O	Used after a subscript to indicate old
-N	Used after a subscript to indicate new
K_{ij}	Symmetric key shared between $MECU_i$ and ECU_j
K_i	Symmetric key shared between $MECU_i$ & SMECU
Gi	Group key shared between $MECU_i$ and its ECUs
H(X)	Hash code of X
SIG(X)	Signature of X
N_X	Private key of X in Elliptic Curve Cryptology
P_X	Public key of X in Elliptic Curve Cryptology

B. Keys Generation and Distribution

The pre-installed keys will be used once to distribute the newly created keys and then ignored. Each ECU will create its own public and private keys. The MECUs and the SMECU will also create their public and private keys.

Each ECU will send its public key to its MECU. The new public key, $PU_{ECU\text{-}N}$, will be encrypted by the current public key of the MECU, PU_{MECU}, and then by the old private key of the ECU, $PR_{ECU\text{-}O}$. In other words, the encrypted new public key is signed before sending it to MECU:

ECU → MECU: $E[PR_{ECU\text{-}O}, E(PU_{MECU}, PU_{ECU\text{-}N})]$.

Note that here only one message is needed. After carrying out the needed decryptions, the MECU will capture the new public key and store it.

 ISBN: 978-1-61208-482-4

The MECU will use a similar approach and send the encrypted and signed new key to the ECUs belonging to it.
MECU → ECU: E[$PR_{MECU\text{-}O}$, E(PU_{ECU}, $PU_{MECU\text{-}N}$)].

Each MECU sends its new public key to SMECU, and the SMECU will provide its new public key to the four MECUs after receiving theirs following the above style

MECU → SMECU: E[$PR_{MECU\text{-}O}$, E(PU_{SMECU}, $PU_{MECU\text{-}N}$)].
SMECU → MECU: E[$PR_{SMECU\text{-}O}$, E(PU_{MECU}, $PU_{SMECU\text{-}N}$)].

Having done that, all the old public and private keys are discarded. This approach is also used when the keys need to be changed periodically or when needed.

C. Secret Value Generation and Exchange

The new shared secret value, S_i, between $MECU_i$ and its ECU_s is generated as follows:

1) Zero the odd bits of S_i to get S'_i
2) Select an ECU_j at random to get its ID, ID_j
3) Compute $X = S'_i$ XOR ID_j
4) Encrypt X with the public key of $MECU_i$ to get Y, $Y = E(PU_{MECUi}, X)$
5) Zero the even bits of Y to get Z
6) Select an ECU_j at random to get its public key, PU_{ECUj}
7) Encrypt Z with this public key to get the new S_i, $S_{i\text{-}N} = E(PU_{ECUj}, Z)$

The same algorithm is used for generating the shared secret value, V_j, between MECU j and the SMECU after replacing ECU with MECU, and MECU with SMECU.

D. ECU's Public Key and ID Exchange

Each MECU is in charge of ensuring its members have the public keys and IDs of all other members. The MECU will send messages containing the public key, ID, and time stamp to each ECU. The messages are encrypted with the private key of the MECU and then with the public key of the ECU in question. For example, MECU4 will send the following two messages to ECU4s (refer to Figure 2 above):

$X_1 = E(PR_{MECU4}, PU_{ECU41} \| ID_{ECU41} \| T)$
$X_2 = E(PR_{MECU4}, PU_{ECU42} \| ID_{ECU42} \| T)$
$MECU_4 \rightarrow ECU_{4s}$: $E[PU_{ECU4s}, X_1)$
$MECU_4 \rightarrow ECU_{4s}$: $E[PU_{ECU4s}, X_2)$

An alternative would be to have the MECU issue certificates. However, certificates will require more communication traffic in this case.

E. Securing ECU Messages

An ECU message, MSG, includes the payload, ECU ID and message ID (MSG = Payload || ID_{ECU} || ID_{MSG}). There are other contents that fulfill other ECU or bus requirements. However, these will not be included in the security protocol. The broadcasting ECU carries out the following:

1) Encrypt MSG with its private key
2) Calculate the cryptographic hash for MSG || S_i, $H(MSG \| S_i)$
3) Sign the cryptographic hash using an agreed upon digital signature algorithm. This signature will be denoted by SIG [$H(MSG \| S_i)$]

It then broadcasts the following three protocol messages to its MECU and members of its group:

$M_1 = E (PR_{ECUij}, MSG \| T]$
$M_2 = E (PR_{ECUij}, H(MSG \| S_i) \| T]$
$M_3 = E (PR_{ECUij}, SIG [H(MSG \| S_i)] \| T]$

$ECU_{ij} \rightarrow X$: M_1
$ECU_{ij} \rightarrow X$: M_2
$ECU_{ij} \rightarrow X$: M_3

Here X is used to denote other ECUs in the group and $MECU_i$. The second and third protocol messages need to be padded to make them the same length as the first message.

Upon receiving these messages, the $MECU_i$ will broadcast a message to its members indicating which ECU broadcasted the message. The message contains the ID of the broadcasting ECU. This will allow the ECUs to use the right public key to decrypt each message. Assuming ECU_{11} from the group controlled by $MECU_1$ is broadcasting, $MECU_1$ broadcasts the following message:

$MECU_1 \rightarrow ECU_{1j}$: $E (PR_{MECU1}, ID_{ECU11} \| T)$.

At this point, each ECU is ready to decrypt the first message with the public key of the sender to get the message, MSG. It then checks the message ID, ID_{MSG}, to see if it needs to do anything. If the message does not concern it, there is no need to decrypt the other two messages. Otherwise, the receiving ECU calculates the hash of MSG || S_i and compares it to the received hash code in message two. Then, the signature received in message three is verified. If either the hash code or the signature cannot be verified, the message is ignored and $MECU_i$ is informed. This could imply a hardware or software issue at the sender site, or a possible attack.

$MECU_i$ will recover MSG from message M1, encrypt it first with its private key and then with the public key of SMECU. It then computes the hash of the message and V_j ($H(MSG \| V_j)$, and signs the resulting hash code.

 ISBN: 978-1-61208-482-4

The resulting messages will be sent to SMECU. The SMECU will perform the needed decryptions, and verifications of the hash code and signature. Based on message ID, ID_{MSG}, SMECU will make the decision on which MECUs should receive it. A similar approach will be used to create three different messages for each MECU that needs this MSG. Once these messages are received by the MECU and MSG is recovered, the MECU will broadcast the received message to its ECUs.

F. Fulfilling Security Requirements

Most of the sent messages are encrypted with the public key of the receiver. This ensures confidentiality because no one can decrypt the message but the one who owns the related private key. When a message is broadcasted, it cannot be encrypted by the public key of the receiver due to the fact that a number of simultaneous receivers exist. However, encrypting it with the private key of the sender will ensure only the members of the group can decrypt. It could be argued here that the message is confidential for other members of the group because only those members know the public key of the sender.

To ensure message integrity, all the messages have their hash code added. Furthermore, the hash code is signed with the private key of the sender. The receiver can verify the integrity of the received message by calculating the hash code and comparing the two hash codes. If there is a mismatch, the message has been modified.

To ensure that parties (ECUs, MECUs, and SMECU) are communicating with the right parties, the messages are encrypted with the private key of the sender. In addition the hash code also serves as the authenticator.

IV. Other Possible Security Approaches

A. Using Symmetric Keys

Symmetric key cryptology can also be used to secure the proposed security architecture. The initialization step will be the same as for the public key cryptology. The public and private keys are removed, and symmetric keys K_{ij} are shared between the MECUs and ECUs. Furthermore, symmetric keys K_i are shared between MECUs and SMECU. Each $MECU_i$ creates a group key, G_i to be shared with its ECUs. The SMECU generates four session keys and shares a unique one with each MECU. Encryption with symmetric key provides confidentiality and authentication. The three messages above will be re-written as:

$M_1 = E (K_{ij}, MSG \| T]$

$M_2 = E (K_{ij}, H(MSG \| S_i) \| T]$

$M_3 = E (K_{ij}, SIG [H(MSG \| S_i)] \| T]$

B. Utilizing Elliptic Curve Cryptology

Elliptic curve cryptology (ECC) is also effective in securing the above-mentioned architecture. The initialization will include pre-installing the global public elements $E_q(a, b)$, G, and n. Here, $E_q(a, b)$ is an elliptic curve with parameters a and b, q is a prime integer, G is a point on the elliptic curve whose order is a large value n.

Each group including the MECU and its ECU members will create their private keys, N_X, and calculate their public keys, P_X, where X indicates any ECU, or MECU. To illustrate this, the group of $MECU_1$ (refer to Figure 2 above) is selected. The following procedure is used:

1. $MECU_1$ selects its private key N_1 and calculates its public key P_1, $P_1 = N_1 \times G$
2. ECU_{11} selects its private key N_{11} and calculates its public key P_{11}, $P_{11} = N_{11} \times G$
3. ECU_{12} selects its private key N_{12} and calculates its public key P_{12}, $P_{12} = N_{12} \times G$
4. ECU_{1n} selects its private key N_{1n} and calculates its public key P_{1n}, $P_{1n} = N_{1n} \times G$
5. The MECU and ECUs broadcast their public keys. Therefore, each one of them will have all the public keys: P_1, P_{11}, P_{12}, and P_{1n}.
6. The messages M_1, M_2, and M_3 will be represented as points on the curve $E_q(a, b)$ when broadcasted.
7. The MECU will send the ID of the broadcasting ECU to allow the ECUs to use the right public keys.

The same procedure of generating and exchanging keys applies to MECUs and the SMECU but without the broadcasting of step 5. Instead the SMECU and each MECU will exchange their public keys. At the end, each MECU will have the public of the SMECU only, but the SMECU will receive the public key of the four MECUs. Step 7 will be deleted, as there is no broadcasting between the MECUs and the SMECU.

With elliptic curve cryptology, only the signature will be used. No hash code or message authentication code will be employed.

C. Employing Stream Cipher

Another approach would be using One-Time Pad (OTP). The keystream $S = \{S_0, S_1 \ldots S_n\}$ will be generated using a True Random Number Generator (TRNG), such as Intel Digital Random Number Generator (DRNG) [20], or the full-hardware implementation of a true number generator suggested by Schaumont [21]. For this purpose, the MECUs and the SMECU should encompass the hardware needed for generating true random numbers. Initially, all the shared key streams need to be pre-installed at manufacturing time.

The SMECU will create four different keystream using the installed hardware for TRNG, one for each MECU. The generated keystream will be encrypted with the old keystream (initially, the pre-installed one and later the current one) shared with each MECU and sent to MECUs. Likewise, each MECU creates a keystream using its TRNG hardware, encrypts it with the old keystream and send it to its members

 ISBN: 978-1-61208-482-4

(ECUs). Once these keystreams are established, the pre-installed ones are discarded.

To broadcast a message, that message should be encrypted with the keystream prior to broadcasting it. If the bus frame is full, another a frame will be used to transmit what is left of the message. The MECU in charge of the broadcasting ECU will forward it to the SMECU encrypted with the shared keystream. Once received by SMECU, it will be analyzed and sent to the MECUs that need the broadcasted message for their members (ECUs). Once the decryptions are performed, all used keystreams will be discarded and new keystreams will be generated.

V. CONCLUSION AND FUTURE WORK

The Electronic Control Units (ECUs) play a critical role in controlling many of the functions of current day's vehicles. Because these ECUs are part of the in-vehicle networks, the possibility of security attacks is inevitable. To protect the vehicle ECUs against various network attacks, a security architecture based on the notion of master and super master ECUs to ensure ECUs' secure message broadcasting was proposed. This architecture was implemented using public key cryptology. The master and super master ECUs also simulated the role of a Key Distribution Center (KDC) through being in charge of generating keys for the units under their control. The super master ECU controlled the broadcasting of ECUs' messages from one group of ECUs to the other groups. Furthermore, the paper showed that other security approaches are reasonable. To this extent, symmetric key cryptology, Elliptic Curve Cryptology, and stream ciphers were investigated.

Future work will concentrate on the implementation phase. During this phase, the optimal grouping of ECUs will be determined. A comparison of the four approaches; public key cryptology, symmetric key cryptology, stream ciphers, and Elliptic Curve cryptology will be carried out to select the most suitable approach for securing the ECUs. Furthermore, the most convenient algorithm that takes into consideration the computing resources limitations of the ECUs will be adopted.

REFERENCES

[1] CCS, "Electronic Control Units (ECUS)," 2014, http://www.ccs-labs.org/teaching/c2x/2014s/05-ecus.pdf, pp. 1-27, [retrieved: April 2016].

[2] L. Delgrossi, "The Future of the Automobile Vehicle Safety Communications," 2014, https://cache.freescale.com/files/automotive/doc/white_paper/BODYDELECTRWP.pdf, [retrieved: April 2016].

[3] ETAS GmbH, "Electronic Control Unit (ECU) – Basics of Automative ECU," 2014, http://www.scribd.com/doc/268828296/20140121-ETAS-Webinar-ECU-Basics#scribd, pp. 1-30, [retrieved: April 2016].

[4] Freescale, "Future advances in Body Electronics" https://cache.freescale.com/files/automotive/doc/white_paper/BODYDELECTRWP.pdf, 2013, pp. 1-18, [retrieved: April 2016].

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

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

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

[8] S. Seo, J. Kim, S. Hwang, K. Kwon, and J. Jeon, "A Reliable Gatway for In-Vehicle Networks Based on LIN, CAN, and FlexRay," ACM Transcation on Embedded Computing Systems, vol. 4, no. 1, Article 7, 2012, pp. 1-24.

[9] T. Tomonari, "EMC Countermeasures for In-Vehicle Communication Networks," TDK Corporation, https://product.tdk.com/en/products/emc/guidebook/eemc_practice_09.pdf, pp. 1-7, [retrieved: April 2016].

[10] P. Nisch, "Security Issues in Modern Automative Systems," 2012, pp. 1-7, http://www.panisch.com/wp-content/uploads/2012/06/Security_Issues_in_Modern_Automotive_Cars.pdf, [retrieved: April 2016].

[11] L. B. Othmane, H. Weffers, M. M. Mohamad, and M. Wolf, "A Survey of Security and Privacy in Connected Vehicles," in Wireless Sensor and Mobile Ad Hoc Networks, Part III, Springer, New York, 2015, pp. 217-247.

[12] S. Mahmud and S. Shanker, "In-Vehicle Secure Wireless Personal Area Network (SWPAN)," IEEE Transcations on Vehicular Technology, vol. 55, no. 3, 2006, pp. 1051-1061.

[13] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. Mccoy, B. Kantor, D. Anderson, H. Shacham, and 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.

[14] 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.

[15] K. Han, S. D. Potluri, and K. Shin, "On Authentication in a Connected Vehicle: Secure Integration of Mobile Devices with Vehicular Networks," in Proc. the 2013 ACM/IEEE 4th International Conference on Cyber-Physical Systems (ICCP'13), Philadephia, PA, USA, 2013, pp. 160-169.

[16] C. Patsakis, K. Dellios, and M. Bouroche, "Towards a Distributed Secure In-Vehicle Communication Architecture for Modern Vehicles," Computers and Security, vol. 40, 2014, pp. 60-74.

[17] Continental Automotive GmbH, "Electronic Vehicle Management - New Options for Commercial Vehicle Controllers," http://www.continental-automotive.cn/www/download/automotive_cn_cn/general/contact_services/downloads/commercial_vehicles/flc_vcu_cvam_en.pdf, [retrieved: April 2016].

[18] D. K. Nilsson, P. H. Phung, and 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.

[19] K. Y. Cho, C. H. Bae, Y. Chu, M. and W. Suh, "Overview of Telmatics: A System Architecture Approach," International Journal of Automative Technology, vol. 7, no. 4, 2006, pp. 509-517.

[20] Intel Digital Random Number Generator (DRNG), May 15, 2015, https://software.intel.com/sites/default/files/managed/4d/91/DRNG_Software_Implementation_Guide_2.0.pdf, [retrieved: April 16].

[21] S. Schaumont, "True random Number Generation," Circuit Cellar, No. 268, 2012, pp. 52-58.

 ISBN: 978-1-61208-482-4