MACsec Extension over Software-Defined Networks for In-Vehicle Secure Communication*

Ju-Ho Choi, Sung-Gi Min
Department of Computer Science and Engineering
Korea University
Seoul, Republic of Korea
jubong@korea.ac.kr

Youn-Hee Han
School of Computer Science and Engineering
Korea University of Technology and Education
CheonAn, Republic of Korea
yhhan@koreatech.ac.kr

***Abstract*—The automotive industry has recently introduced Advanced driver assistance systems (ADAS) for safety and better driving. Many Electronic Control Units (ECUs) have been installed in the vehicle to support ADAS, and massive data stream flows over the in-vehicle network. Therefore, the Ethernet backbone, which can guarantee the high bandwidth, has emerged as an in-vehicle communication technology. However, security on automotive Ethernet has not yet been proposed. The IEEE MACsec with IEEE 802.1X Authentication and Key Management (AKM) may be applied for the in-vehicle secure communication, but it has a constraint that its security scope is based on a point-to-point approach. Whenever a frame arrives at the switches in the transmission path, the decryption and re-encryption of the frame are repeated. It may adversely affect the performance of ADAS related to the driver's safety by increasing the end-to-end latency. We therefore propose a new MACsec extension over the Software-Defined Networks (SDN) for an in-vehicle secure communication, which is based on IEEE 802.1X authentication mechanism. The proposed scheme extends the security scope of MACsec from point-to-point to end-to-end by delegating AKM process of ECUs and switches to SDN controller. It could minimize the cryptographic processes of the ECUs and switches without any modification of the existing MACsec standard, and could protect an automotive system from any manipulation by unauthorized third parties. The experimental results show that the proposed scheme is applicable for an in-vehicle secure communication.**

***Index Terms*—In-vehicle secure communication; automotive Ethernet; IEEE 802.1AE; authentication and key management (AKM)**

I. Introduction

In the recent automotive industry, the introduction of the Advanced Driver Assistance Systems (ADAS) such as collision avoidance and automatic parking has been actively attempted. More than 80 Electronic Control Units (ECUs) are distributed and interconnected within the vehicle to support ADAS, and it requires a large bandwidth to expedite massive data stream. Therefore the Ethernet backbone [1], which can guarantee the high bandwidth and the transmission of time sensitive date, is attracting attention as the next-generation technology to be used in in-vehicle communication.

The ADAS could help drivers to drive safely and prevent drivers from accidents, but at the same time it also poses the risk of exposure to various security threats. Several car hacking demonstrations [2] have shown that the vehicle's safety critical systems such as steering or braking can be manipulated remotely. As a result, the secure communication based on the Ethernet backbone has become a more critical issue in modern automotive systems. The coexistence of security and safety must be ensured in in-vehicle communications.

The original approach of Ethernet [3] has the advantages of simplicity and zero-configurability, but there are security vulnerabilities [4] such as intrusion and man-in-the-middle attacks. In order to solve this problems, the IEEE 802.1AE [5] standard for Local and Metropolitan Area Networks has been release. In terms of functionality, 802.1AE guarantees the confidentiality and integrity of all frames on an Ethernet link, but it has the limitation that its security scope is only valid on a single LAN. In other words, the encrypted frame transmitted by the source node should be decrypted whenever it encounters another LAN, and should be re-encrypted with a valid key on each LAN. As the repeated encryption and decryption of the frames may causes a delay in frame transmission, many automotive applications, which require strict latency guarantees [6], may not meet the safety related requirements such as end-to-end latency.

In this paper, we propose a MACsec extension architecture over Software-Defined Networks (SDN) for in-vehicle secure communication. The proposed scheme adapts the SDN concept so that the central SDN controller delegates the Authentication and Key Management (AKM) procedure of the Electronic Control Units (ECUs) and switches. In the proposed SDN based AKM mechanism, the SDN controller authenticates the ECUs, rather than using mutual authentication between ECUs. It not only reduces the cryptographic process cost of ECUs and switches participating in the AKM procedure, but also protects an automotive system from any manipulation by unauthorized third parties. The switches with forwarding rules installed from the SDN controller could block the frames sent by unauthorized ECUs, and only ECUs certified by the SDN controller could communicate within the networks.

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2017R1D1A1B03032303) and also supported by the MIST (Ministry of Science and ICT), Korea, under the National Program for Excellence in SW (2015-0-00936) supervised by the IITP (Institute for Information and Communications Technology Promotion).

In addition, the SDN controller establishes a secure association between the ECUs on behalf of them. As a result, the security scope of the MACsec is expanded from point-to-point to end-to-end without any modification to the existing MACsec standard. The frame transmitted by the source ECU is sent to the destination ECU without being decrypted even though it passes through the intermediate switches, so that it is possible to reduce the computation cost of the network caused by repetitive encryption and decryption of the frame. Consequently, the in-vehicle network has the flexibility that safely and quickly accommodates to the changing needs of an automotive systems, such as installing a new electronic control unit in a vehicle.

The rest of this paper is organized as follows. Section 2 provides the overview of a MACsec and a 802.1X AKM procedure. Section 3 describes the overall design of the proposed architecture. Section 4 and 5 presents the operation of the proposed AKM scheme in detail. Section 6 shows the simulation results. Finally, Section 7 concludes this paper.

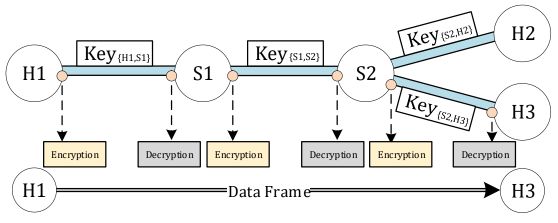
II. Related Works

A. IEEE 802.1AE

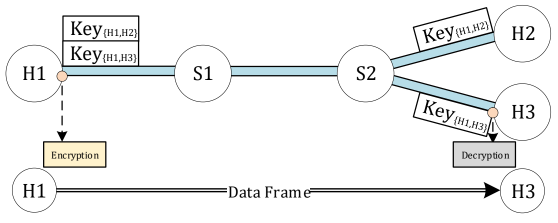
The 802.1AE [5] , also referred to as Media Access Control Security (MACsec), is the IEEE MAC Security standard for secure communication between nodes on Ethernet links. It is capable of preventing most layer 2 security threats such as denial of service, intrusion and man-in-the-middle attacks. The confidentiality and integrity of all traffic on the Ethernet link can be guaranteed by the MACsec.

However, the MACsec cryptographically protects frames based on a hop-by-hop rather than an end-to-end basis. The network fragment presented in Figure 1 shows the difference between the point-to-point approach and the end-to-end approach of MACsec. In the hop-by-hop approach shown in Figure 1(a), a different key is used for each link (LAN). Therefore, a node that receives a frame should decrypt the frame and re-encrypt the frame when it has to forward this frame to another link. The encryption and decryption of the frame occurs three times in the Figure 1(a). In other words, the more links a frame goes through, the more the number of times the frame is encrypted and decrypted increases. On the other hand, if the the end-to-end approach is applied to the MACsec as shown in Figure 1(b), frame encryption and decryption occur only once at the source and destination host. However, even with these advantages, applying the end-to-end approach directly to MACsec has limitation. As shown in Figure 2, the MACsec encrypts the Virtual LAN (VLAN) tag of the Ethernet frame. Therefore, the switches receiving the encrypted MACsec frame could not identify and modify the VLAN tag of the frame if it does not have a corresponding security key. For example, if the destination node is assigned to different VLAN and the intermediate switches cannot modify the VLAN tag of the encrypted MACsec frame, the frame may not reach to destination node.

The SDN concept [8], which separates the control and data plane of the switches, can be a solution that could extend



(a) Host-to-host Approach.



(b) End-to-host Approach.

Fig. 1. MACsec Approach.

MACsec security scope from point-to-point to end-to-end, and overcome the limitation of the end-to-end approach described above. If the central controller, which knows the current states of the network, dynamically installs packet processing rules into the switches, the switches can forward the MACsec frames without decrypting the frames.

B. 802.1X MAEsec Key Agreement Mechanism

To encrypt an Ethernet frame via IEEE 802.1AE, a Secure Channel (SC) between MACsec devices should first be established. The MACsec Key Agreement (MKA) protocol [7] is used to generate such a SC. According to the MKA protocol described in section 9 of IEEE 802.1X, the MACsec devices on the same LAN should first belong to a unique Connectivity Association (CA). Members of the CA have shared Connectivity Association Key (CAK) and Connectivity Association Key Name (CKN), and they identifies members in the CA through whether they own this CAK and CKN. Then, the MACsec device establishes an unidirectional SC to the peer MACsec device belonging to the CA. The SC consists of a series of transient sessions called Secure Association (SA). Each SA is designated as a Secure Association Key (SAK). The MACsec device having this SAK encrypts the non-MACsec frame into a MACsec frame during transmission and vice versa during reception.

III. In-vehicle MACsec extension over SDN Architecture

In the proposed architecture, the security scope of the MACsec is extended from point-to-point to end-to-end by adapting SDN concept. The proposed architecture consists of the following three type of network entities: Electronic control unit, OpenFlow switch, and SDN controller. Figure 3 shows

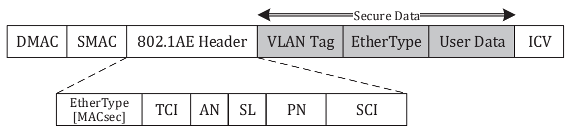


Fig. 2. MACsec frame format.

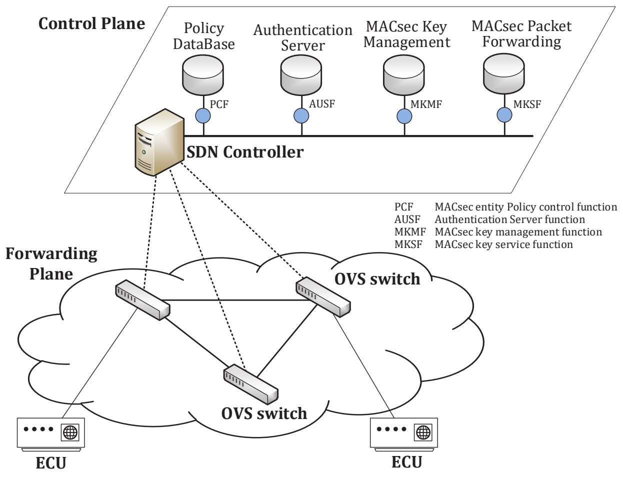


Fig. 3. The proposed MACsec over SDN Architecture

the in-vehicle network architecture for the proposed MACsec extension over SDN mechanism.

A. ECU

A Electronic Control Unit (ECU) refers to all of the potential embedded devices within the vehicle. It controls the important units such as an engine based on the signal coming from various sensors. The ECU may need to communicate not only with ECUs directly connected to it but also with ECUs that are separated by more than 1 switches. For example, displaying the information collected by the sensor at the rear of the vehicle on the glass in front of the driver would require more than 1 L2-hop communication.

The ECUs should have a kernel capable of running IEEE 802.1AE security standard [5], and they also run MACsec Key Agreement (MKA) protocol, which is defined in 802.1X [7] as a companion of 802.1AE. The original MKA provides mutual authentication and secure key generation between nodes that want to belong to a MACsec connectivity association. However, in the proposed architecture, as the authentication of all ECUs is performed through the SDN controller, mutual authentication between ECUs is not required. In addition, in the proposed architecture, the SDN controller generates the Secure Association Keys (SAKs) on behalf of ECUs. Therefore, the ECUs operate the modified MKA with only the function to receive the created SAK and to install it in their interface. We assume that a ECUs have the Authentication Server (AS) identifier to which they can connect and a AS-registered Pre-Shared Key (PSK).

B. SDN Controller

The SDN controller [9] [10] is the strategic control point of the proposed scheme. As the SDN controller knows where each node is located and how each node is connected, it can act as an intermediary that extends MACsec security scope from point-to-point to end-to-end. The SDN controller should have at least four application: a policy database service, an authentication service, a MACsec key management service, and a MACsec packet forwarding service. These services may be directly operated at the SDN controller or may be operated on a node physically separate from the SDN controller, and the SDN controller requests the services in the form of the network functions [11].

The role of the policy database service is to manage information about which of the ECUs connected to the network should establish the SAs with each other, and notifies the SDN controller of the information of these permitted MACsec pair when the SDN controller requests this information. The authentication service is needed to authenticate the newly connected ECUs according to the 802.1X authentication mechanism [7]. The MACsec key management service generates a SAK on behalf of the ECUs, resulting in the establishing of an SA between the ECUs. In the proposed architecture, the ECUs do not create the SAK directly, but the SDN controller collects all the information needed to generate the SAK from these services and generate the SAK instead of the ECUs. The generated SAK is transmitted securely to ECUs. The MACsec packet forwarding service modifies the packet forwarding rules and actions of the switches whenever a new MACsec pair's SA is established. The SDN controller uses the OpenFlow [12] protocol to communicate with the switches as the OpenFlow is a standard interface for communication between machines and network switches. Through this dynamic remote administration of switches' packet forwarding tables, the data frame that the source ECU has encrypted using 802.1AE [5] can reach the destination ECU without being decrypted on the switches.

C. OpenFlow Switch

In the proposed architecture, OpenFlow switches are used without any modification. It identifies received MACsec frame based on the MAC address of the frame, and forwards the frame according to the forwarding rules that are added and modified by SDN controller.

IV. Proposed Authentication Mechanism

Figure 4 shows the authentication procedure when a new ECU is connected to the OpenFlow switch. The purpose of this authentication procedure is twofold: first, to determine whether a new connected ECU is allowed to connect to the LAN; second, to establish a Secure Association (SA) for secure communication between the SDN controller and the ECU.

The SDN controller periodically broadcasts EAPoL-Announcement message. The EAPoL-Announcement message

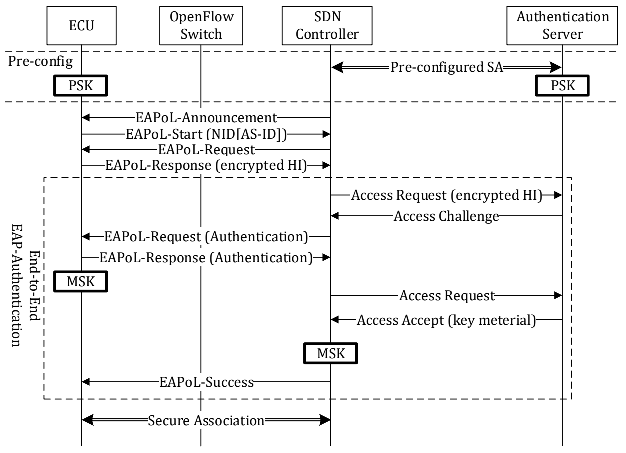


Fig. 4. The proposed MACsec over SDN Authentication Mechanism

includes information such as access information, MACsec cipher suites, and Network Identifier (NID) of the Authentication Server (AS). After the ECU receives the EAPoL-Announcement message, it initiates the Authentication and Key Management (AKM) mechanism in the manner of IEEE 802.1X [7] by sending the EAPOL-Start message to the SDN controller. The message contains the NID of the AS. When the SDN controller receives the EAPOL-Start message, it verifies whether it has pre-configured SA with the AS corresponding to the NID-AS in the EAPoL-Start message. If it does not have a SA with the AS, it establishes the SA with the AS. Otherwise, the SDN-controller performs the authentication procedure described in IEEE 802.1X-2010, Section 8. The SDN controller then requests the ECU to send the Host Identity (HI) via the EAPoL-Request message. Then, the ECU transmits the HI encrypted with the PSK through the EAPoL-Request message, and the SDN controller relays it to the AS. The AS attempts an access challenge based on EAP authentication mechanism such as EAP-Message Digest5 (MD5) [13] or EAP-Transport Layer Security (TLS) [14]. Consequently, a Master Session Key (MSK) is generated between the SDN controller and the ECU after the authentication procedure is successfully completed.

V. Proposed MACsec Key Management Mechanism

Figure 5 shows the MACsec key management procedure when a SA between ECUs should be established in the network. After each ECU has completed authentication procedure from the AS, they should establish an SA between them for end-to-end secure communication. However, according to the MACsec Key Agreement (MKA) protocol described in section 9 of IEEE 802.1X, a generated Secure Association Key (SAK) is only valid between hosts located on the same LAN. To extend the range of SAK between ECUs from point-to-point to end-to-end, the SDN controller generates a SAK on behalf of the ECUs as an intermediary and transmits it via the secure channels that it has established with each ECU.

Algorithm 1 Packet Forwarding Rule at OpenFlow Switch

```
flow : [
    priority: 50000,
    timeout: 0,
    isPermanent: true,
    deviceId: of:000008e9e00000c2,
    treatment: [
        instructions: [
        type: OUTPUT,
        port: 3
        ]
    ]
    selector: [
        criteria: [
          type : Ether Source,
          mac : ECU 1,
          type : Ether Destination,
          mac : ECU 2,
          type : Ether Type,
          ethType : 0x88e5
        ]
    ]
]
```

The SDN controller requests the information about the MACsec key pair that needs to establish an SA to the Policy DB (PDB). It then registers the received MACsec key pair information and generates a SAK for the ECUs based on this information. The key derived function described in section 6 of IEEE 802.1X is used for generating the SAK. The SDN controller sends an EAPoL-MKA message to each ECU, and the message contains the encrypted SAK and the encrypted MAC address of the pair ECU. Since the messages are transmitted via the already established secure channel, any node other than the corresponding ECUs cannot decrypt the SAK. At the same time, the SDN controller issues a forwarding rule to the OpenFlow switches to which each ECU is connected. The Algorithm 1 shows an example of a packet forwarding rule at OpenFlow-Switch (OF-Switch) 1. The forwarding rule described in Algorithm 1 means that if the source MAC address of the received frame is ECU 1 and the destination MAC address is ECU 2 and the EtherType of the frame is 0x88e5, then OF-Switch 1 has to send the frame to OF-Switch 2. Consequently, OpenFlow switches with this forwarding rule could send the received frame to appropriate OpenFlow switch without frame decryption. The ECU that receives the EAPoL-MKA message registers the MACsec pair ECU and installs the SAK in its interface. As a result, the SA between the ECUs is successfully established with the help of the SDN controller.

VI. Experiments

In this section, we present an experimental testbed for the proposed scheme to proof that it is applicable in the in-vehicle secure communication. Figure 6 shows the system topology of the testbed. The freeRadius 3.0.15 version is used as an AAA server for the authentication of the ECUs. The Open Network Operating System (ONOS) [9], which is the popular open-source SDN controller, is used as the SDN controller. We

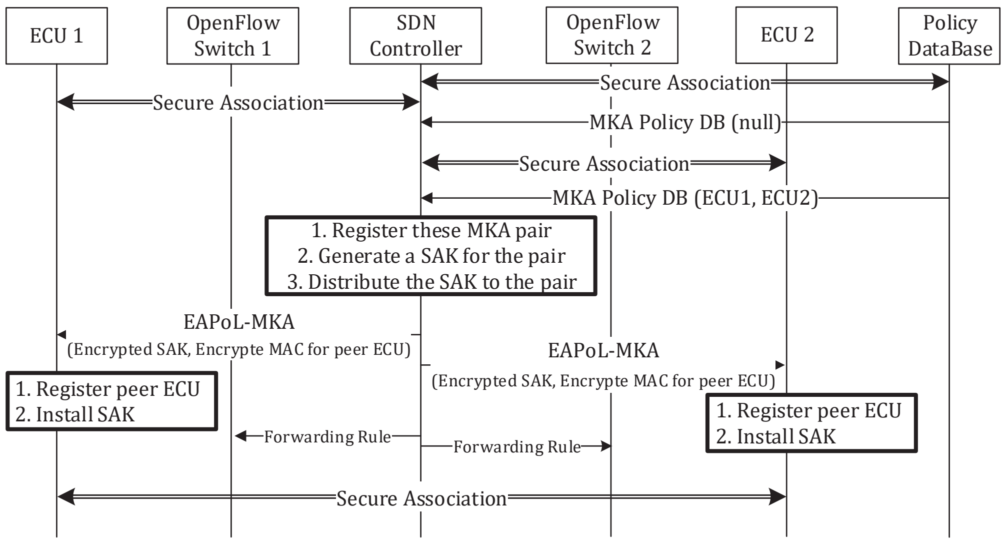


Fig. 5. The proposed MACsec over SDN Key Distribution Mechanism

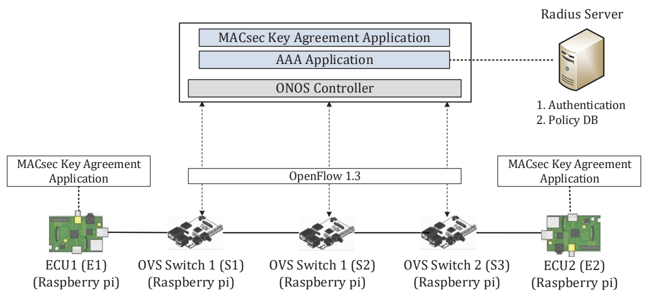

Radius Server
MACsec Key Agreement Application
AAA Application
ONOS Controller
1. Authentication
2. Policy DB
MACsec Key Agreement Application
OpenFlow 1.3
MACsec Key Agreement Application
ECU1 (E1) (Raspberry pi)
OVS Switch 1 (S1) (Raspberry pi)
OVS Switch 1 (S2) (Raspberry pi)
OVS Switch 2 (S3) (Raspberry pi)
ECU2 (E2) (Raspberry pi)

Fig. 6. Experiment network model

employ a Raspberry Pi 3 Model B as the network entities. Two Raspberry Pi takes the role of ECU in the vehicle, and two Raspberry Pi with Open vSwitches (OVSs) connect the ECUs to the SDN controller via Ethernet link. The OVS interacts with the ONOS controller with the OpenFlow protocol version 1.3.2 [15].

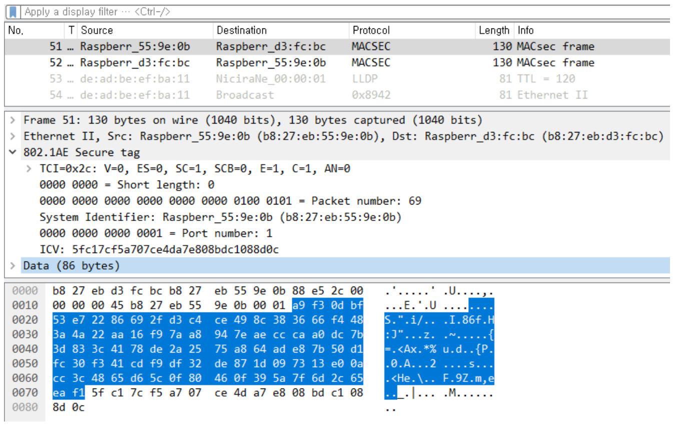
The AAA authentication application and the controller MACsec Key Agreement (MKA) application run over the ONOS controller, while the host MKA application run over ECU 1 and ECU 2. The radius server acts as an authenticator for the ECUs and as a Policy DataBase (PDB) for the potential MACsec pair. After both ECU 1 and ECU 2 have successfully authenticated, the ONOS controller requests the MACsec pair policy to the radius server. After the ONOS controller receives the PDB from the radius server, it registers ECU 1 and ECU 2 as a MACsec pair in its MACsec Pair Management Table (MPMT) and generates a SAK to be used between the MACsec pair. The AES-CMAC-128 [16] algorithm is used for generating the SAK and the AES Key Wrap [17] is employed to protect the confidentiality and the integrity of the SAK. The generated SAK is transmitted securely through the secure channel that the ONOS controller has established with each ECU. At the same time, the ONOS controller installs a forwarding rule to the OVSs.

ECU 1 and ECU 2 obtains the MAC address of the peer ECU and the generated SAK from the EAPoL-MKA message. They install the SAK on their Ethernet interface. After that, ECU 1 sends the ICMP Echo (ping) [18] packet to ECU 2.

Figure 7 shows the captured packets for the outgoing interface of the ECU 1 and the incoming MACsec interface of the ECU 2. Figure 7(a) shows that the ICMP ping packet is encrypted by the SAK, and the 802.1AE security tag is attached to the Ethernet frame. This encrypted frame can not be decrypted by any node except ECU 1 and ECU 2 which have established SA with each other. Figure 7(b) presents that the receiving MACsec interface of ECU 2 successfully decrypts the ping packet. Through the experimental results, we can confirm that the existing point-to-point based MACsec is extended to end-to-end by introducing the SDN concept.

VII. Conclusions

The proposed MACsec extension over SDN architecture expands the security scope of the 802.1AE from point-to-point to end-to-end without any modification of the 802.1AE. It minimizes the cryptographic processes required on the electronic control units and intermediate switches by delegating the AKM process of them to the SDN controller. Compared to IEEE 802.1AE with IEEE 802.1X AKM, the proposed scheme encrypts and decrypts the frame once at the source ECU and at the destination ECU, respectively. As a result, the overall transmission delay of the frame could be decreased, and it could ensure more stringent end-to-

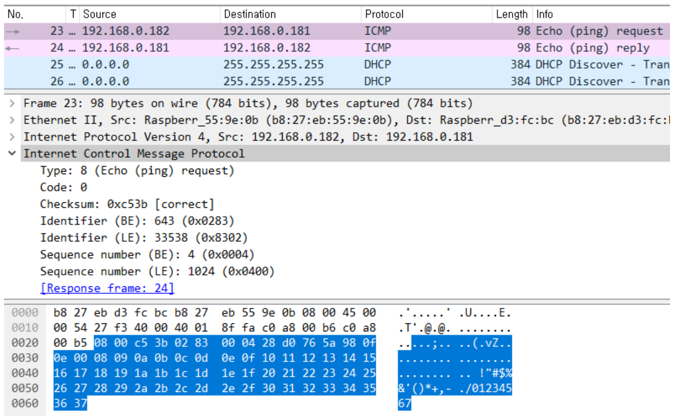


```
Apply a display filter ··· <Ctrl-/>
No.      T Source             Destination        Protocol     Length Info
     51 … Raspberr_55:9e:0b  Raspberr_d3:fc:bc  MACSEC          130 MACsec frame
     52 … Raspberr_d3:fc:bc  Raspberr_55:9e:0b  MACSEC          130 MACsec frame
     53 … de:ad:be:ef:ba:11  NiciraNe_00:00:01  LLDP             81 TTL = 120
     54 … de:ad:be:ef:ba:11  Broadcast          0x8942           81 Ethernet II

> Frame 51: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits)
> Ethernet II, Src: Raspberr_55:9e:0b (b8:27:eb:55:9e:0b), Dst: Raspberr_d3:fc:bc (b8:27:eb:d3:fc:bc)
v 802.1AE Secure tag
  > TCI=0x2c: V=0, ES=0, SC=1, SCB=0, E=1, C=1, AN=0
    0000 0000 = Short length: 0
    0000 0000 0000 0000 0000 0000 0100 0101 = Packet number: 69
    System Identifier: Raspberr_55:9e:0b (b8:27:eb:55:9e:0b)
    0000 0000 0000 0001 = Port number: 1
    ICV: 5fc17cf5a707ce4da7e808bdc1088d0c
> Data (86 bytes)

0000  b8 27 eb d3 fc bc b8 27  eb 55 9e 0b 88 e5 2c 00   .'.....' .U....,.
0010  00 00 00 45 b8 27 eb 55  9e 0b 00 01 a9 f3 0d bf   ...E.'.U ........
0020  53 e7 22 86 69 2f d3 c4  ce 49 8c 38 36 66 f4 48   S.".i/.. .I.86f.H
0030  3a 4a 22 aa 16 f9 7a a8  94 7e ae cc ca a0 dc 7b   :J"...z. .~.....{
0040  3d 83 3c 41 78 de 2a 25  75 a8 64 ad e8 7b 50 d1   =.<Ax.*% u.d..{P.
0050  fc 30 f3 41 cd f9 df 32  de 87 1d 09 73 13 e0 0a   .0.A...2 ....s...
0060  cc 3c 48 65 d6 5c 0f 80  46 0f 39 5a 7f 6d 2c 65   .<He.\.. F.9Z.m,e
0070  ea f1 5f c1 7c f5 a7 07  ce 4d a7 e8 08 bd c1 08   .._.|... .M......
0080  8d 0c                                              ..
```

(a) Encrypted Ping Packet at ECU 1.



```
No.      T Source             Destination        Protocol     Length Info
→    23 … 192.168.0.182      192.168.0.181      ICMP             98 Echo (ping) request
←    24 … 192.168.0.181      192.168.0.182      ICMP             98 Echo (ping) reply
     25 … 0.0.0.0            255.255.255.255    DHCP            384 DHCP Discover - Tran
     26 … 0.0.0.0            255.255.255.255    DHCP            384 DHCP Discover - Tran

> Frame 23: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
> Ethernet II, Src: Raspberr_55:9e:0b (b8:27:eb:55:9e:0b), Dst: Raspberr_d3:fc:bc (b8:27:eb:d3:fc:
> Internet Protocol Version 4, Src: 192.168.0.182, Dst: 192.168.0.181
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xc53b [correct]
    Identifier (BE): 643 (0x0283)
    Identifier (LE): 33538 (0x8302)
    Sequence number (BE): 4 (0x0004)
    Sequence number (LE): 1024 (0x0400)
    [Response frame: 24]

0000  b8 27 eb d3 fc bc b8 27  eb 55 9e 0b 08 00 45 00   .'.....' .U....E.
0010  00 54 27 f3 40 00 40 01  8f fa c0 a8 00 b6 c0 a8   .T'.@.@. ........
0020  00 b5 08 00 c5 3b 02 83  00 04 28 d0 76 5a 98 0f   .....;.. ..(.vZ..
0030  0e 00 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   ........ ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

(b) Decrypted Ping Packet at ECU 2.

Fig. 7. Captured ICMP Ping packet at ECU 1 and ECU 2.

end latency for the safety related automotive applications. In addition, the proposed scheme could prevent the an automotive system from any manipulation by unauthorized third parties. Since the communication is possible only between the nodes authenticated by the SDN controller, it is possible to block the third parties from interfering with this communication. The experimental results show that the proposed scheme is practical for in-vehicle communication.

REFERENCES

[1] T. Steinbach, K. Muller, F. Korf, and R. Rollig, "Demo: Real-time Ethernet in-car backbones: First insights into an automotive prototype," in Vehicular Networking Conference (VNC), pp. 133134, 2014.

[2] M. Charlie and V. Chris, "Remote Exploitation of an Unaltered Passenger Vehicle," Black Hat Conference, 2015.

[3] IEEE, IEEE Standard for Ethernet, in IEEE Std 802.3-2012 (Revision of IEEE Std 802.3-2008), IEEE: New York, NY, USA, pp. 1634, 2012.

[4] T. Kiravuo, M. Sarela, and J. Manner, "A Survey of Ethernet LAN Security," IEEE Communications Surveys and Tutorials 15, pp. 14771491, 2013.

[5] IEEE, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security in IEEE Std 802.1AE-2006, IEEE: New York, NY, USA, pp. 1142, 2006.

[6] R. Alieiev, A. Kwoczek, and T. Hehn, "Automotive requirements for future mobile networks," IEEE MTT-S Int. Conf. Microw. Intell. Mobility (ICMIM), pp. 14, 2015.

[7] IEEE, IEEE Standard for Local and Metropolitan Area Networks: Port-based Network Access Control. in IEEE Std 802.1X-2010 (Revision of IEEE Std 802.1X-2004), IEEE: New York, NY, USA, pp. 1222, 2010.

[8] D. Kreutz et al., "Software-defined networking: A comprehensive survey," Proc. IEEE, vol. 103, no. 1, pp. 1476, 2015.

[9] P. Berde, M. Gerola, J. Hart, and Y. Higuchi, "ONOS: Towards an open, distributed SDN OS," in Proc. 3rd Workshop Hot Topics Software Defined Networking, 2014.

[10] J. Medved, R. Varga, and A. Tkacik, "Opendaylight: Towards a Model-Driven SDN Controller Architecture," Proc. 15th IEEE WoWMoW, pp. 16, 2014.

[11] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," IEEE Access, vol. 3, pp. 25422553, 2015.

[12] http://www.opennetworking.org.

[13] R. Rivest, "The MD5 Message-Digest Algorithm," IETF, RFC 1321, 1992.

[14] D. Simon, B. Aboba and R. Hurst, "The EAP-TLS Authentication Protocol," IETF, RFC 5216, 2008.

[15] Specification, "OpenFlow Switch. 1.3.2," 2013.

[16] JH. Song, J. Lee, and T. Iwata, "The AES-CMAC Algorithm," IETF, RFC 4493, 2006.

[17] J. Schaad and R. Housley, "Advanced Encryption Standard (AES) Key Wrap Algorithm," IETF, RFC 3394, 2002.

[18] A. Conta, S. Deering, and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," IETF, RFC 4443, 2006.