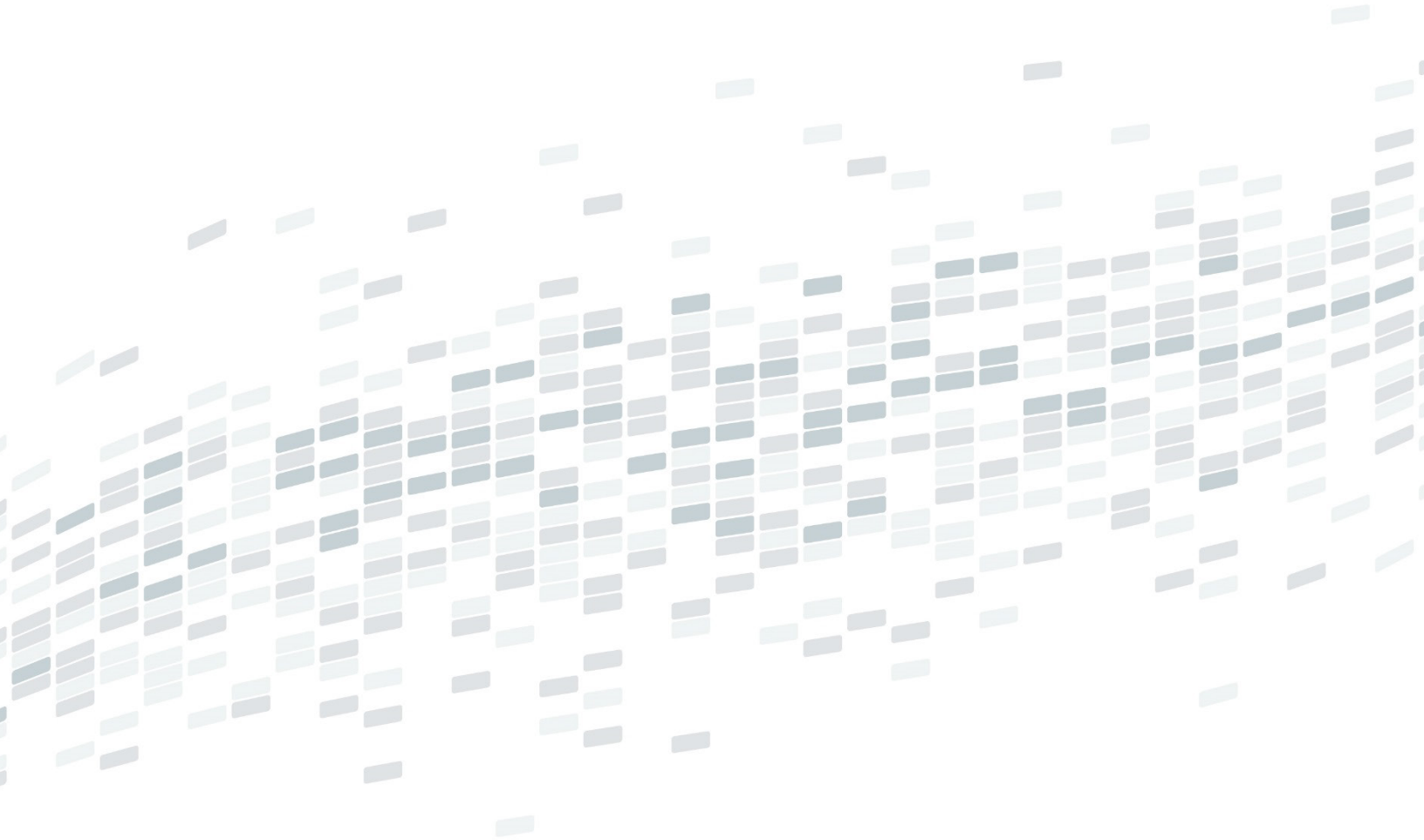
WHITE PAPER
www.visteon.com

Visteon®

Securing Inter-Processor Communication in Automotive ECUs



Securing Inter-Processor Communication in Automotive ECUs

Karthik Shanmugam, Technical Professional, Software

Abstract

Modern cars now come with sophisticated telemetry which often involve connecting to the internet over mobile telephone networks or Wi-Fi. The telemetry or cloud functions of the car is typically handled by a Telematics Control Unit or the Infotainment System. The microcontrollers (Host Processor) powering the ECUs are very powerful and often have operating systems such as Linux or QNX to drive the large displays or perform modem functionalities. These powerful microcontrollers take several seconds to startup and does not offer hard real-time performance - both of which are critical to handle the vehicle CAN network. Hence, it is common to include a less powerful microcontroller to the ECU to perform the management of the vehicle CAN network. These smaller microcontrollers (Vehicle Processor) can startup fast and provide hard real-time performance. The Host Processor and the Vehicle Processor are connected by the Inter-Processor Communication Link (IPCL) to exchange messages between them. This often overlooked communication link is also a security vulnerability in the vehicle. This was made obvious when the vehicle functionalities of the 2015 Jeep vehicle was controlled remotely by unauthorized actors, which involved compromising the communication link and reprogramming the Vehicle Processor to take control of the Vehicle CAN bus. This paper analyses the threat vectors pertaining to IPCL and provides solutions that address each of those threats with minimal impact to the performance of the communication link.

Introduction

The advancements in the modern car are driven by the consumer electronics industries such as smartphones and Internet of Things devices. Today's customers expect the same level of features and convenience from their cars as they expect from their consumer electronics devices. This requires the car to have a communication channel to the Internet over mobile telephone networks or Wi-Fi to provide advanced features such as Over-The-Air software updates, remote car operations and uploading diagnostic telemetry. Few examples of such services are GM OnStar and Chrysler Uconnect. These advanced functionalities are handled by the dedicated Telematics Control Unit or the vehicle's infotainment system which has the necessary hardware to communicate over cellular networks or Wi-Fi.

The microcontrollers (Host Processor) that are present in these ECUs are powerful and require advanced operating systems such as Linux or QNX to perform their duties. These operating systems will take several seconds to startup and does not provide hard real-time performance which are required to manage the vehicle CAN bus. Such ECUs will have an additional low power microcontroller (Vehicle Processor) which can startup fast and provide real-time performance to manage the vehicle CAN bus. The Host Processor and the Vehicle Processor are connected by the Inter-Processor Communication Link (IPCL) to exchange messages between them, it makes the IPCL link as the gateway to the Vehicle CAN network. The IPCL link often realized by industry standard serial communication buses such as UART or SPI. This makes the IPCL link a window into the operation of the ECU to understand what sort of messages triggers specific vehicle functionalities and how to exploit them. The IPCL link has since become the attack surface and it is important to protect the IPCL link from the security threats. In case of the 2015 Jeep hack [1] the IPCL link was exploited to patch the program in the vehicle processor to take control of the vehicle CAN bus.

The paper analyzes the threats pertaining to the IPCL link and provides a solution that addresses those threats effectively. The solution involves protecting the integrity, freshness and the confidentiality of the data exchanged over the communication medium using industry standard encryption and security best practices with minimal overhead.

Threat Analysis

The threats are described as anything that would contribute to the tampering, destruction or interruption of any service. In case of the IPCL the following threats have been analyzed.

- Snooping or Spying
- Man-in-the-Middle attack
- Replay attack
- Brute force attack

Snooping

Snooping or spying the IPCL link involves monitoring and logging the data sent over the communication medium. It is trivial to attach probes to the UART or SPI lines and log the signals to analyze the data. By exercising the valid features of the remote service and by analyzing the data sent over the communication link, it may be possible to reverse engineer the commands required to trigger certain critical vehicle functions such as starting the engine or braking. This vector requires physical access to the ECU to be compromised, but can be very effective in finding an exploit that can be triggered remotely.

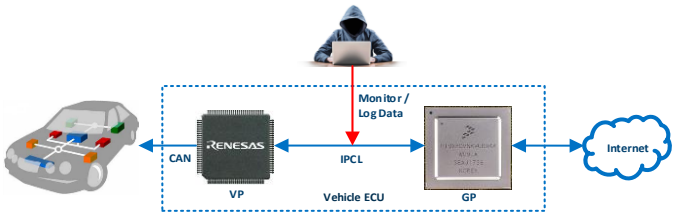


Figure 1. Representation of snooping of data on the IPCL link.

Man-in-the-Middle Attack

Man-in-the-Middle attach - or in short mitm - is a real-time attack where the malicious actor places himself in between the Vehicle processor managing the CAN bus and the Host Processor that may have an external network connection to the internet. The malicious actor then can modify the data being exchanged between the Vehicle CAN network and the external network connection by modifying the data that are being exchanged in real time.

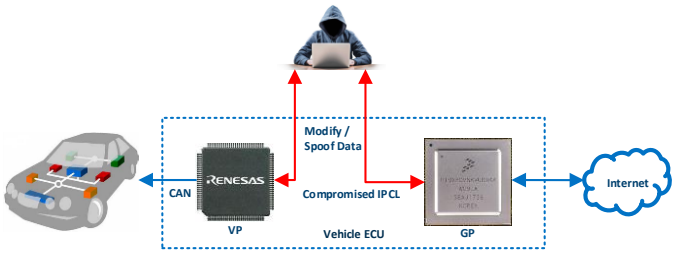


Figure 2. Representation of Man-in-the-middle attack modifying or faking data on the IPCL link.

Replay Attack

Replay attack involves recording the transactions on the IPCL bus during valid transfers and replay the recorded transfers to effect the same outcome when it is not intended to be. For example, someone can record the transaction during software update of the vehicle processor and replay the sequence with modified payload to re-write the vehicle processor firmware and take control of the vehicle CAN bus.

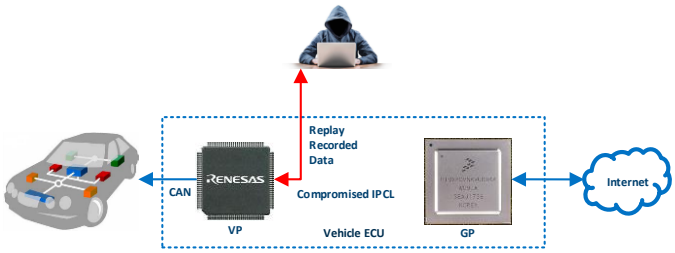


Figure 3. Representation of replay attack where playing a pre-recorded sequence to send spurious data on the IPCL link.

Brute force Attack

Brute force attack is employed when the bus transactions has to be authenticated with a pass phrase. The attacker tries to guess the pass phrase or the key that will allow the attacker to spoof the malicious messages as legitimate messages.

Once an exploitable security hole has been found, the compromised software in the host processor can be remotely used to control the vehicle processor to send messages over the vehicle CAN bus.

Threat Mitigation

Secure IPCL

Securing the IPCL data link is thus paramount in the ECUs where it can be connected to an external network. The below picture shows a typical IPCL frame commonly used to communicate between vehicle and host processors.

Figure 4. Typical IPCL data frame

The IPCL message frame consists of a message header to identify the message type and routing information followed by the payload and a message footer to verify the integrity of the message using message checksum or CRC. This payload carries data in the plain text offering no protection from any of the above discussed threats. The following security features are essential to protect the IPCL bus from the security threats.

1. Origin authentication
2. Content protection
3. Time limited validity
4. Velocity control

To secure the IPCL message frame the frame structure is modified as shown below

Figure 5. Secure IPCL data frame

The Secure IPCL frame message header is updated to provide additional security attributes for the payload and a time limited token to avoid record and replay attacks. The payload is encrypted if total confidentiality is required or plain-text only if the tamper detection is required. The secure footer consists of CMAC code to authenticate the sender of the frame and also to ensure the integrity of the message. We shall see in the coming section how the modified IPCL frame allow us to address all the threat vectors that has been analyzed.

Time limited validity

Time limited validity is an important feature to avoid the record and replay attach and also to avoid generating the same cryptographic signature when the payload is not changing. A time limited token is introduced in the header to limit the validity of the message to prevent record and replay attacks The time limited token is verified by the receiver to ensure the freshness of the received message This makes the receiver reject any messages that are replayed because the token value would be expired and will not be valid. This prevents the record and replay attack.

Origin Authentication

The Origin Authentication ensures that the message has been sent by the authenticated sender and not by anyone else. The authentication and the integrity of the message is ensured by using the Cipher based Message Authentication Code (CMAC) instead of the checksum to verify the payload. The commonly used encryption algorithm to perform CMAC is AES-128 (Advanced Encryption Standard).

The sender feeds the entire message frame into the CMAC algorithm along with the specific session-key to generate the CMAC code that is appended to the message instead of the regular checksum.

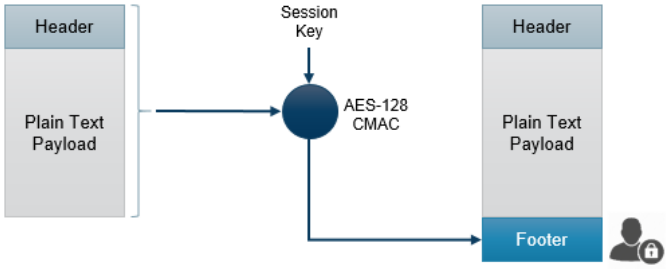


Figure 6. Generation of CMAC

The receiver performs the same operation to re-generate the CMAC code and compares against the received value, if the value matches it validates the origin and the integrity of the messages. To achieve proper verification the session-key has to be identical in the sender and the receiver.

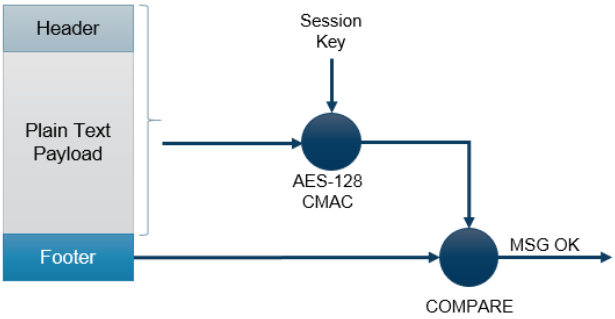


Figure 7. Verification of CMAC

Content Protection

The Origin Authentication prevents the message from being tampered with or spoofed by any malicious actor but it still exposes the content for logging and inspection. The payload is encrypted using the Cipher Block Chaining (CBC) to protect the content from being snooped and analyzed.

The sender performs the same steps required for Origin Authentication to generate the CMAC and uses the CMAC as the Initial Vector (IV) and session-key to encrypt the payload along with the time limited token. The content now is fully encrypted and prevents snooping or logging of the messages.

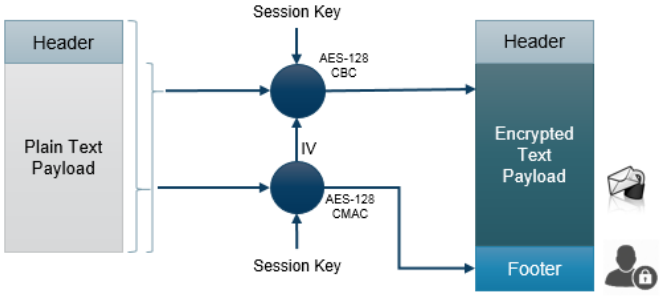


Figure 8. Secure IPCL Message Encryption

The receiver decrypts the received message using the session-key and the received CMAC as the IV, once the message is decrypted the CMAC is calculated for the plain text message and verified against the received CMAC value to make sure the message is authentic.

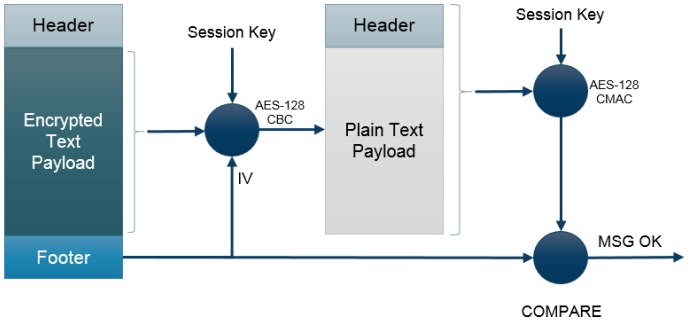


Figure 9. Secure IPCL Message Decryption

Performing crypto operations are not cheap and they are a time consuming operation if performed entirely in the software. It is recommended to use the cryptographic hardware accelerators and secure storage mechanisms provided by the microcontroller to speed up the crypto operation to reduce the processing overhead.

Velocity Control

Brute force attacks are used to find or guess the password or the session-key so that any malicious actor can again pose as a legitimate sender. To prevent brute force attacks the velocity control is implemented wherein any security related failure such as invalid CMAC or failed encryption will be followed by the bus idle time where no new messages will be accepted into the system. The idle time is gradually increased for the subsequent failures. This mechanism forces brute force an exponentially time consuming activity to the point of uselessness.

Measurement Setup

The following diagram shown in the Figure 1 describes the test setup implemented to evaluate the proposed implementation and measure the

overhead that was added with respect to the additional security features.

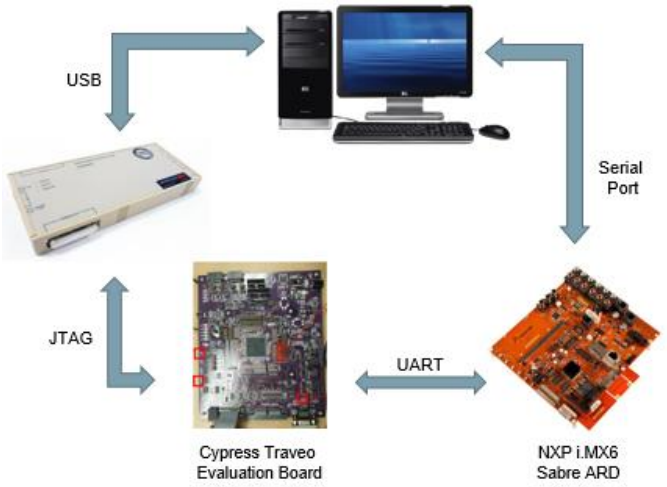


Figure 1. Measurement Setup

The test setup consists of the 1) Cypress Traveo microcontroller with ARM Cortex R5 core and an in-built Secure Hardware Extension module to perform cryptographic operations, 2) NXP i.MX6 microcontroller based Sabre Automotive Reference Design board. The physical transport between the microcontrollers is the UART. The Cypress Traveo is connected to the PC using the Lauterbach JTAG debugger to control the operations of the Cypress Traveo microcontroller. The i.MX6 is connected to the PC using the RS232 serial connection. The Cypress Traveo microcontroller runs the AUTOSAR Operating System and the i.MX6 microcontroller runs the QNX Operating System.

The simplified software block diagram with the relevant components are shown in the Figure 2. The IPCL Stack implements the protocol and the security feature described in this paper, the test applications are used to generate the IPCL frames for measurements.

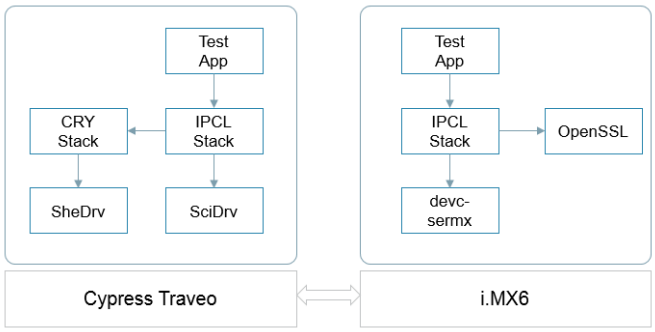


Figure 2. Software Block Diagram

The methods proposed in this the paper has been implemented and found to be successfully preventing all the attack vectors discussed. The following measurements were made to profile the overhead in terms of timing due to addition of the security features.

Table 1. Latency measurements with various security features enabled.

Security Level	Latency (ms)	Payload Size (bytes)
Unsecure	17	100
Authenticated	18	100
Encrypted	20	100

Summary/Conclusions

It is imperative and vital that the IPCL communication link between the vehicle processor and Host processor is protected. Updating ECU software over-the-air is becoming prevalent in the modern vehicle where the host processor will update the firmware of the vehicle processor over IPCL, it is vital to secure the firmware update process using the appropriate security measures described in the paper. Choosing the correct microcontroller with appropriate security hardware is necessary to keep the overhead to the minimum. The IPCL Security is one of the many vectors that can be used to compromise and ECU. The ECU will also need to be protected using Secure boot, Secure JTAG to prevent tampering of the firmware using physical access.

References

1. Black Hat USA 2015: The full story of how that Jeep was hacked. https://www.kaspersky.com/blog/blackhat-jeep-cherokee-hack-explained/9493/
2. O'Flynn, C. and d'Eon, G., "Power Analysis and Fault Attacks against Secure CAN: How Safe Are Your Keys?," SAE Int. J. Cybersecurity 1(1):3-18, 2018, doi:10.4271/11-01-01-0001.
3. Zou, Q., Chan, W., Gui, K., Chen, Q. et al., "The Study of Secure CAN Communication for Automotive Applications," SAE Technical Paper 2017-01-1658, 2017, doi:10.4271/2017-01-1658
4. Carl A. Sunshine. 1976. Factors in interprocess communication protocol efficiency for computer networks. In Proceedings of the June 7-10, 1976, national computer conference and exposition (AFIPS '76). ACM, New York, NY, USA, 571-576. doi:10.1145/1499799.1499879
5. "Specification of Module Secure Onboard Communication - Version 4.2.2", 2015, [online] Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_SWS_SecureOnboardCommunication.pdf
6. The AES-CMAC Algorithm, https://tools.ietf.org/html/rfc4493
7. The AES-CBC Cipher Algorithm and Its Use with IPsec, https://tools.ietf.org/html/rfc3602

Contact Information

Karthik Shanmugam, karthik.shanmugam@visteon.com, Visteon Technical and Services Center, Chennai.

Acknowledgments

I would like to acknowledge my co-workers CN, Aananth (A.A.), Adiraju, Srini (S.), Sarat Chandra Prasad, Gingupalli (G.) and Shimabukuro, Jorge (J.) for their guidance and feedback.

Definitions/Abbreviations

AES	Advanced Encryption Standard
CAN	Controller Area Network
CBC	Cipher Block Chaining
CMAC	Cipher based Message Authentication Code
ECU	Electronic Control Unit
FOTA	Firmware Over The Air
IPCL	Inter Processor Communication Link
KRV	Key Response Value
KVV	Key Verification Value
RS	Random Seed
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver Transmitter