Securing CAN Communication Efficiently With Minimal System Impact

Bernd Elend, Principal Engineer; Thierry Walrant, Senior System Security Architect; Georg Olma, Global Business Development Manager

Securing CAN communication efficiently can be a challenge. Different options exist today with payload authentication such as AUTOSAR® SecOC [1]. These solutions use cryptography and require management for crypto keys in addition to higher bandwidth and processing power, potentially resulting in longer transmit delays.

NXP's new secure CAN transceiver family provides a very efficient solution to secure CAN communication without using cryptography. It helps developers avoid the system impacts experienced with other solutions.

NXP believes that secure CAN communication is possible with transceivers that offer distributed intrusion detection and containment methodology without cryptography [2]. CAN message identifier (ID) filtering mechanisms in the transmit and receive path help prevent and contain network security attacks such as spoofing, remote frame tampering and denial of service by flooding. By monitoring and filtering network traffic on the bus, the secure CAN transceiver protects that CAN bus from any ECU attempting to send unauthorized malicious messages.

From a security perspective, the obvious choice is to use cutting-edge solutions to protect against security threats with cryptographic message authentication code (MAC), based on cryptography and associated secure key management. Secure microcontrollers are designed with crypto accelerators to support these state-of-the-art solutions. Despite this unique hardware support, this solution is not always the most efficient for secure CAN communication. Secure microcontrollers will likely be used to secure end-to-end communication over multiple CAN networks, other networks such as Ethernet or LTE and secure boot, authentication diagnostics and authenticated firmware updates.

Cryptographic checks of message authenticity consistently result in an increased message latency, busload and computing power consumption. So, applying these kinds of solutions can be prohibitive, or lead to compromises such as:

1. Applying message authentication codes (MAC) to only a low fraction of the CAN messages in a network
2. Truncating the MAC and/or freshness value beyond reasonable limits
3. Having potential unsecured grace periods [3].

Existing ECU designs cannot be upgraded when the selected µC-family does not have sufficient compute power to secure the CAN messages. In addition, key management for the entire vehicle lifecycle is a complex challenge. Finally, a complete system re-design from Classical CAN to CAN FD, that compensates the increase of busload by introducing SecOC may not always be feasible. However, this article will show that the CAN bus can be be secured more efficiently—without bandwidth overhead, major configuration changes, and lifecycle management needs—through a secure CAN transceiver.

NXP's approach for securing the CAN bus with minimal system impact complements the crypto-based security solutions with an additional layer in a defense-in-depth (DiD)[1] concept, or can operate as a standalone solution. In both cases, the secure CAN transceiver can provide an efficient and strong level of protection and hack containment.

Security features of the secure CAN transceiver

Spoofing prevention on transmit side

One way for the secure CAN transceiver to protect the bus from a compromised ECU is by filtering messages based on CAN message IDs in the transmit path. If the ECU tries to send a message with an ID that is originally not assigned to it, the secure CAN transceiver can refuse to transmit it to the bus. It invalidates the message and denies subsequent transmissions. CAN message ID filtering can be done using a passlist of IDs that the manufacturer can configure. For example, the IDs for Unified Diagnostic Services (UDS), as specified in ISO 14229 for off-board testers, may be excluded from the passlist. This would prevent a compromised ECU from starting a diagnostic session with another ECU in the vehicle to manipulate calibration values, for example.

Spoofing prevention on receive side

A complementary protection is to invalidate messages on the bus in case they are received with a CAN message ID assigned for transmission. This method means each ECU can protect its own IDs in the event that a rogue ECU manages to send a message with the same ID; e.g., an aftermarket device that is not part of car OEM remit and therefore does not have a secure CAN transceiver with a configured Transmission Passlist. When any ECU sends a message on the bus, the secure CAN transceiver of the legitimate ECU can actively invalidate that message by writing an active error frame to the bus. It can do this based on the same passlist as the filtering in the transmit path. The compromised sender will repeat the spoofed message 16 times before suspend-transmission behavior kicks in, limiting the busload contribution. Another 16 repetitions will occur before the attacking ECU enters Bus-Off state. This short peak of busload may be perceived as a negative effect of the countermeasure. On the other hand, this action would only ever take place when the system is under attack, when the countermeasure side-effect would be the least of your concerns.

Tamper protection

Invalidating messages on the CAN bus can also be used to prevent tampering. The secure CAN transceiver can check whether there was a legitimate message on the bus for which its ECU has won arbitration but stopped transmission in the data field (due to receiving a dominant bit while sending recessive), and the tampering ECU completed the message. This is a clear sign that a compromised ECU has stepped into the transmission. During the error-passive state when the CAN controller of the ECU is not reporting errors, this is particularly valuable. Note that message tampering would be a method to bypass the spoofing prevention on the receive side.

Flooding prevention

Limiting the contribution of an ECU to the overall bus load per time unit can help to prevent flooding the bus, when implemented at the sender side. To prevent flooding, a leaky bucket mechanism is used. The bucket is filled while messages are sent and emptied continuously. Further transmissions are stopped upon bucket overflow. Thus, flooding protection increases the availability of the bus, thwarting a communication overflow. To allow bursts of messages by one ECU for a certain amount of time, the size of the bucket can be set appropriately. Messages having a low priority can be excluded from filling the bucket, as they would lose arbitration anyway and thus are unsuitable for a flooding attack. This allows diagnostic services such as software upload/download to exchange messages with low priority at a high rate without triggering the flooding protection.

Preventing spoofing and tampering with these countermeasures legitimizes the sender without the hurdles related to cryptography protocols and a related assets management approach. This offers a less disruptive approach than the state-of-the-art solution. In addition, it supports the DiD concept in which transferring a stolen cryptographic key to a rogue ECU is rendered useless, since the ECU cannot send the CAN message IDs of the messages that it could authenticate with the stolen key.

[1] DiD is a concept in which multiple layers of security countermeasures are placed throughout a system. This provides redundancy in the event a single security countermeasure fails, or a vulnerability is successfully exploited. By applying DiD, the overall system protection is increased, since an attacker will need to circumvent multiple countermeasures to launch a successful attack.

System impacts with state-of-the-art solution

To legitimize a message by a state-of-the-art solution, a payload authentication based on cryptography is used, such as with AUTOSAR SecOC. The original unprotected payload is supplemented by a message authentication code (MAC) and a parameter to identify the freshness of the message. The former is computed according to cryptographic algorithms with a shared secret key. The latter is required for replay protection as it ensures a new and different message authentication code for identical occurrences.

Note: Authenticated payload is either in plain or encrypted and is processed the same way for the purpose of authentication. Recommended security practices require encryption and authentication to be separated, such as by using different shared secret keys for each operation. Therefore, payload encryption is not considered relevant for the sake of this discussion.

The resulting value of cryptography-based payload authentication solutions applied to the CAN network may be impacted by factors that will be discussed here. The level of relevance depends on the security design and architecture. Therefore, not all described factors may apply to every reader's context.

Secure communication startup delay

After system power-on, or ignition, a certain time is required to agree and align the freshness property of the forthcoming authenticated message. This period delays the readiness of the authentication for the first critical message. Although different methods can be considered, it usually involves some agreement with small or extensive payload exchanges among participating ECUs. One can agree on a secure time distribution or secure monotonic counter synchronization. Another way is to refresh the shared keys at each startup and use a reset freshness value. However, given the unpredictable, asynchronous and unequal count of ECU reboot cycles after ignition; the agreement protocol is complex and takes time. Unpredictability could come from possible power losses when the engine starts and unequal count is when there is not an identical distribution of power losses. AUTOSAR defines a grace period [3] that copes with this problem, but it leads to a period after ignition—or forced reboot due to attack—when critical messages are not protected.

Refreshing the keys in a later, stable period could be considered as an alternative. This migrates the complexity to the management of a monotonic counter, which may lead to other problems such as non-volatile memory wear outs or counter desynchronization under chaotic power on.

NXP's secure CAN transceiver helps ensure that any protected message, including the very first, on the bus is legitimate.

Increase of busload and bandwidth

Increase of the busload to transport additional parameters such as MAC and freshness value leads to additional bandwidth needs. The size of the additional parameters impacts the level of security, or the rate at which yet another busload consumption is required. NIST [4] recommends a minimum MAC of 64 bits, i.e., 8 bytes, a fairly important penalty for classical CAN. The size of the freshness counter dictates the rate at which the key must be refreshed, or the rate at which some other secure base counter/freshness synchronization is required.

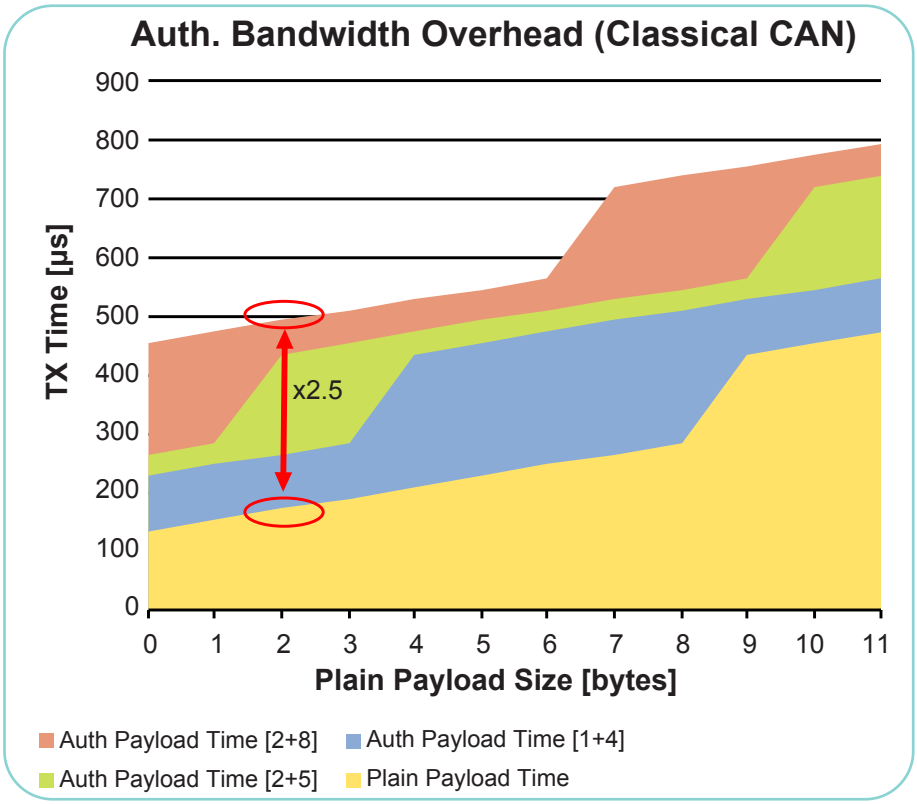

Figure 1. Authenticate bandwidth overhead

The figure shows the bandwidth overhead. It compares transmission times for classical CAN messages with an 11-bit CAN message ID at 500 kbit/s. The original plain payload is in grey with the corresponding values for different authentication parameters [Counter+MAC] (values in bracket are in bytes).

Take red circles, for example. Assuming an unprotected message payload of 16 bits for a common classical CAN message, a counter of 2 bytes and an MAC of 5 bytes for an acceptable protection, the busload increases by a factor of 2.5.

The numbers in the figure refer to message authentication overheads only and do not include a potential increase of the busload to run management protocol for refreshing keys or freshness values.

NXP's secure CAN transceiver operates on the classical CAN and CAN FD protocol without transmitting any additional single bits.

Increase of processor load and functional latency

Extra functional latency will likely be from an extra buffer or interrupt handling and communication to the hardware accelerator, rather than the calculation of MACs during generation and verification. The extra processing load decreases the application performance. The transmission time for the extra bits on the bus is also increased by the order of magnitude of the busload overhead from Figure 1. As seen in Figure 1, it takes 172 µs of a 16-bit unprotected message payload, while it takes 436 µs for the same message with protected payload. The functional latency and real-time behavior of distributed vehicle functions is changed with respect to current architecture and vehicle design.

NXP's secure CAN transceiver protects the CAN communication independently of the µC and does not add any latency.

Additional software complexity

The CAN controller is in charge of relieving the application from complex management and interrupt handling by managing TX buffers or TX queues. In automotive applications, it is common to handle messages with different priorities. Different parts, such as software threads, run separately in parallel. They make use of different TX buffers or TX queues in the CAN driver or CAN controller. The order in which the messages are delivered by the application parts or threads to the CAN driver or controller does not necessarily correspond to the appearance order on the bus. For authenticated messages using a global freshness value—such as global synchronized time—the order on the bus must follow the order of the freshness value used to compute the message authentication code. Otherwise, it is considered old or replayed by the receiver. The pending CAN message payload, waiting to win the arbitration, already contains the message authentication code and the freshness value. Therefore, any higher priority message provided by the application will invalidate the freshness and related authentication of all pending messages. This means the usual power of CAN controller TX buffers and TX queues is depleted and an application can no longer simply send and forget to the CAN controller. Moreover, it needs to be designed to recalculate authentication of invalidated messages pending transmission, which adds extra processing to the application and top-up system latency.

NXP's secure CAN transceiver is a pure hardware only solution.

Need for key management

Any cryptography-based solution relies on the usage of keys, either symmetric or asymmetric. The effort and the complexity of appropriate key management is often underestimated. Payload authentication for real-time communication with a high-performance requirement often uses symmetric cryptography and, therefore, shared secret keys. The security level of a system depends in part on the protection and the privileges of the shared secret keys. By design, a key should have dedicated and limited privileges to reduce the impact of a security breach of the the key itself. If the key is stolen or used, it's stored securely but used by a compromised application. Either of these incidents will provide an application to gain access to the key privilege. In both instances, an application is gaining unauthorized access to a privilege.

Keys should be unique or diversified per individual vehicle, so that a stolen key on vehicle A cannot grant access to the equivalent function on any other similar vehicle; this is fleet attack resilience. Special considerations are required to assign key privileges, key creation and distribution during manufacturing and vehicle life cycles. A static diversified key per vehicle and function can be deployed and a central database for key injection during module replacement maintained. Another option is to assign roles per ECU with certificates for key usage extension and asymmetric cryptography, letting the vehicle dynamically assign or reassign keys. Diffie-Hellman (DH) protocol is sometimes understood as a standard way to distribute keys, but it is not sufficient. The protocol makes sure a secret can be secretly exchanged between two entities, but does not provide any indication with whom the secret is exchanged. This means attackers may participate to a DH session and acquire related valuable secrets. The latter requires additional processes to authenticate the other party such as asymmetric certificate schemes. Therefore, the symmetric key management for payload authentication may translate into certificate management and dynamic usage within the vehicle.

The choices for key management policy over an entire vehicle life cycle are diverse but require specific thoughts and deployment efforts. Moreover, traditional key management solutions may not directly apply to the automotive industry. The key management strategy may depend on design choices for other aspects listed above, such as freshness management after power-on. NXP's secure CAN transceiver operates efficiently, without cryptography – thus without any complex key management.

NXP's secure CAN transceiver operates efficiently, without cryptography – thus without any complex key management.

Conclusion

State-of-the-art cryptography-based solutions:

- Need some form of key management
- Need a way to handle freshness value at startup for replay protection
- May delay the readiness of authentication by a grace period
- Introduce transmission latency for computation of the message authentication code
- Increase the busload to transport the additional parameters (i.e., Mac and freshness value)
- Consume processing cycles of the sender and receiver applications
- May increase the sender application architecture complexity to order the pending transmission according to message priorities and freshness values, i.e., a new higher priority message may invalidate message authentication code of pending low priority messages due to reordered freshness value

The concept proposed by NXP is implemented solely in a secure CAN transceiver. It operates completely independently and in isolation from the microcontroller. This means it provides an inherent level of security and is specifically designed for minimum system impact to overcome the lack of sender identification in a CAN protocol specification. It can be introduced into a network in a stepwise approach (ECU by ECU), without impacting other ECUs, the message latency or the busload, and without increasing the processor load. The implemented spoofing protection mechanism makes sure that whenever a protected message is received by the target ECU (i.e., message receiver) it has been transmitted by the expected sender. Also, the bus is protected immediately after turning on the ignition; our countermeasures do not require any initialization (of individual ECUs) or synchronization (of multiple ECUs on a bus).

Such secure CAN transceivers are provided as hardware replacements for today's standard CAN transceivers. They help developers avoid major hardware and software changes on the ECU and do not affect the operation of other ECUs. This makes the proposed approach a fast, low-effort, non-disruptive and highly cost-effective way to introduce security to the CAN bus—either as a standalone protection mechanism or as an extra layer of defense in addition to other security solutions.

References

[1] AUTOSAR SecOC - Specification of Module Secure Onboard Communication, **https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_SWS_SecureOnboardCommunication.pdf**

[2] iCC 2017 - Cyber security enhancing CAN transceivers (NXP), **https://www.can-cia.org/fileadmin/resources/documents/conferences/2017_elend.pdf**

[3] [ECUC_SecOC_00051] SecOCEnableForcedPassOverride, **https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_SWS_SecureOnboardCommunication.pdf**

[4] SP 800-38B: Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication, **https://csrc.nist.gov/publications/detail/sp/800-38b/final**

How to Reach Us:

Home Page: **www.nxp.com**
Web Support: **www.nxp.com/support**

USA/Europe or Locations Not Listed:
NXP Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.nxp.com/support

Europe, Middle East, and Africa:
NXPHalbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.nxp.com/support

Japan:
NXP Japan Ltd.
Yebisu Garden Place Tower 24F,
4-20-3, Ebisu, Shibuya-ku,
Tokyo 150-6024, Japan
0120 950 032 (Domestic Toll Free)
http://www.nxp.com/jp/support/

Asia/Pacific:
NXP Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@nxp.com

Document Number: SECURECARTRANA4WP REV 1