Achieving Multi-Level CAN (FD) security by complementing available technologies

Olaf Pfeiffer, Embedded Systems Academy,
Thierry Walrant, Georg Olma, NXP Semiconductors,
Andreas Walz, Prof. Dr. Axel Sikora, Hochschule Offenburg

Securing existing industrial communication protocols like Controller Area Network - CAN (FD) - requires a look at all protocol layers. Although security solutions are known on separate layers, there are practical limits to their application. As an example, adding multiple security layers to a simple I/O node like an encoder might not always be feasible due to resource constraints in small microcontrollers.
Our paper examines, how security solutions for individual layers can be combined to best complement each other in a real-world CAN/CANopen system.
The discussed complementary security mechanisms are:

1. **Black- and white-list filtering of the received and transmitted CAN (FD) frames, plus limitation of the transfer rate of individual devices (flood protection) and secure configuration methods**
2. **Authentication of CAN (FD) frames and secure grouping with CANcrypt**
3. **End-to-end security protocols like TLS to typically secure communications beyond the local network and to implement remote end-to-end security, for example for remote diagnostics**

We investigate on the entire lifecycle of a system from production to integration, commission/adding/removing components, key distribution and management, as well as diagnostics and service. Always considering resource requirements, we examine the limits of each method and show how they can complement each other and significantly improve overall system security with a smart combination of security mechanisms.

CAN (FD) security challenges and limitations

If in embedded systems devices computing performance, memory resources and communication bandwidth were unlimited, we would be using best in class asymmetric cryptography with extremely long keys in all devices.
However, in deeply embedded systems, microcontroller resources are limited. Therefore, embedded security is always a trade-off. System designers must decide how much security can be afforded by each device? They must consider the entire system and distribute the different methods in an optimum architecture so that they can complement each other.
Before introducing existing CAN (FD) security methods, let us look at an exemplary CAN (FD) system to determine the typical security threats and attack vectors available.

System Example: Elevator/Lift

The issues around CAN (FD) security can be illustrated with the help of a real-life example, such as an elevator system. Such a system typically comes with multiple CAN branches (domains), including a bridge or gateway. It requires some diagnostic access for maintenance, possibly also a remote access connection for servicing and external setting of system parameters. The physical protection of the CAN wires is somewhat limited; a determined local attacker can possibly get access to CAN wiring behind buttons and displays. However, such local attacks are limited to a single system, remote access potentially provides access to “all” systems.
Figure 1 shows a simplified system. One CAN domain connects the fixed installations: four floor user interface units (each with

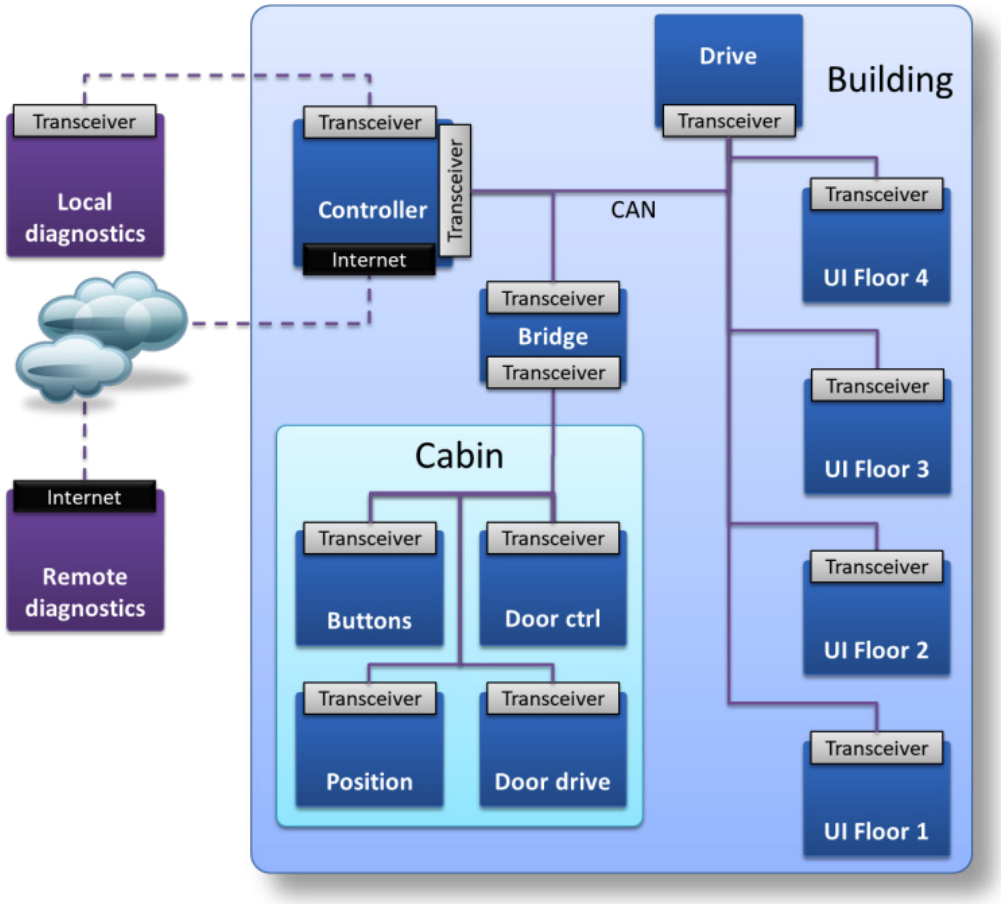

Figure 1: An elevator system with local and remote access for maintenance/diagnostics

button and display), an electrical drive and the main control unit with remote access and maintenance interfaces. The second domain connects all the electronics of the cabin. A bridge links the two domains.

Attack options

The attack options range from a local over remote to backend attacks.

Local, physical attack:
"Unlimited physical access" means an attacker could cut wires as well as remove, add or replace any devices. Added electronics could have their own wireless interfaces and allow receiving all CAN (FD) messages and injecting CAN (FD) messages. More advanced electronics could even perform bit-level manipulations. Attacks could bypass CAN (FD) altogether by manipulating sensors or actuators. At this level, an attacker also has access to all maintenance and debug ports that microelectronic devices might have.

Remote access attack:
All CAN (FD) devices with additional communication interfaces (for example through Bluetooth (BT), WiFi or other service interfaces) can be accessed through these other interfaces, thus increasing the attack surface. Once an attacker has access to such a device connected to CAN (FD), the attacker can monitor the entire CAN (FD) communication (of the local domain) and inject CAN (FD) frames as desired. Without any protection, an attacker can typically take control of major system functions, as important control commands can be injected.

Backend access attack:
Today many CAN (FD) systems are also "cloud" connected, which means there is one or multiple servers in a backend system that have authorized access to all CAN (FD) systems connected. Attackers that have compromised such a backend, immediately have access to all the connected CAN (FD) systems and all functionality provided by the backend system.

CAN (FD) security limitations

On CAN (FD) level, we cannot protect from a backend access attack. Whatever the backend system is authorized to do with the CAN (FD) system, an attacker with full backend access can do with the system.

Regarding the unlimited physical access, security options are limited, too. If cryptographic keys used are stored in flash memory of non-secure microcontroller systems, attackers can potentially extract them. There are even commercial extraction services available to help with such tasks.

Which CAN (FD) security solutions are available?

As shown in [EWC2019], multiple technologies are available for CAN (FD) security.
The methods introduced typically only provide a solution for a portion of the overall security requirements. The “brute force” method to secure a CAN (FD) system would therefore be to combine “all methods” in “every device” of the system. However, is that really required and affordable? Do we need to fully implement “everything”, or can these methods complement each other?
Before answering these questions, let us review the major CAN (FD) security technologies available.

CAN ID protection, white- and black listing (ID Guard)

The principle of an ID Guard is based on the possibility to monitor the CAN traffic at the data-link layer and to take immediate action during a transmission whenever the observed behavior does not comply with the system definition. The ID guard ensures the compliance of the system specifications of the CAN mapping. The ID guard legitimates the successful CAN communication, hence every received CAN message can only be transmitted by the legitimate sender. Possible mechanisms are as follows.

- Transmission filtering is used for spoofing protection by a local node. It prevents the compromised local host to impersonate another node on the bus.
- Transmission rate limiting is used to prevent some sort of DoS or flooding by restricting bus load usage to the expected level
- Bus monitoring and filtering is used for spoofing protection from the bus. It prevents other node on the bus to impersonate the transmission of the local host.
- Tampering protection is used to detect corruption of transmission during Error Passive state.

When a security incident is detected by the ID guard – any violation of the specification detected by the above mechanisms – the message transmission gets invalided resulting in an unsuccessful reception or the node gets disconnected from the bus.
One solution for ID guarding was previously introduced by [iCC2017El].

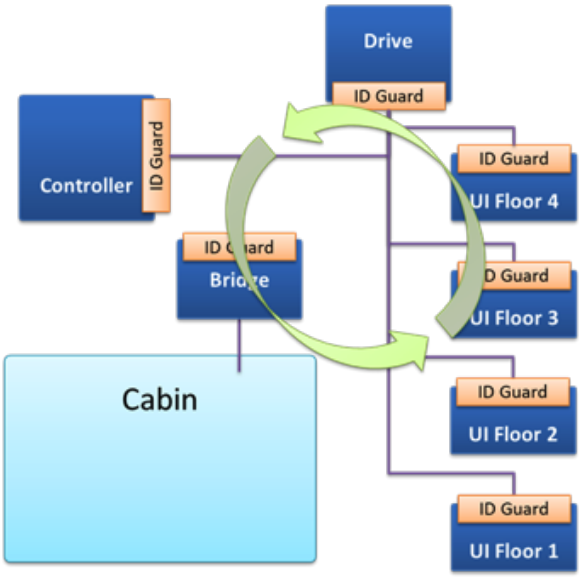

Figure 2: With ID guard protection, all devices protect each other

CAN (FD) cryptographic layer (CAN CL)

The basis for these mechanisms is a shared cryptographic secret. Typically, methods are provided to update a symmetric key continuously based on the shared secret. This can happen “in the background”, for example once per second.
Once a shared secret is available, cryptographic checksums for CAN (FD) frames can be generated. The input to the creation of such checksums typically includes:

- CAN (FD) ID
- CAN (FD) DLC
- CAN (FD) data
- message counter or timestamp (to prohibit replay attacks)

The currently shared key is used on the data above to create the cryptographic checksum.

To authenticate a CAN (FD) frame, a security record including a cryptographic checksum (or a truncated version of it) needs to be added to the CAN (FD) data or unused CAN ID bits (e.g. expanding 11bit ID frames to 29bit ID frames). If there is no room (CAN (FD) data fully used, ID fully used) for the security record in the frame itself, an additional CAN (FD) frame is required to transmit the record.

To authenticate a message, consumers must receive both the CAN (FD) frame and the security record with the matching cryptographic checksum.

The security provided at this level is an authentication of selected messages with a cryptographic checksum and optional message encryption. This mechanism can also be used to implement a secure grouping as illustrated by figure 3.

Depending on MCU performance available and desired protection, such a method could be used in our elevator example to protect the essential devices in the system as shown in figure 3. Here the controllers and drives are securely grouped via the CAN cryptographic layer. One solution for a CAN cryptographic layer was previously introduced by [iCC2017Pf].

A note on the security methods usable here: the methods used to encrypt data or generate cryptographic checksums are typically based on lightweight block ciphers that embedded microcontrollers can handle. Stronger block ciphers are only used, when all devices connected either provide cryptographic engines or have the CPU power to process them.

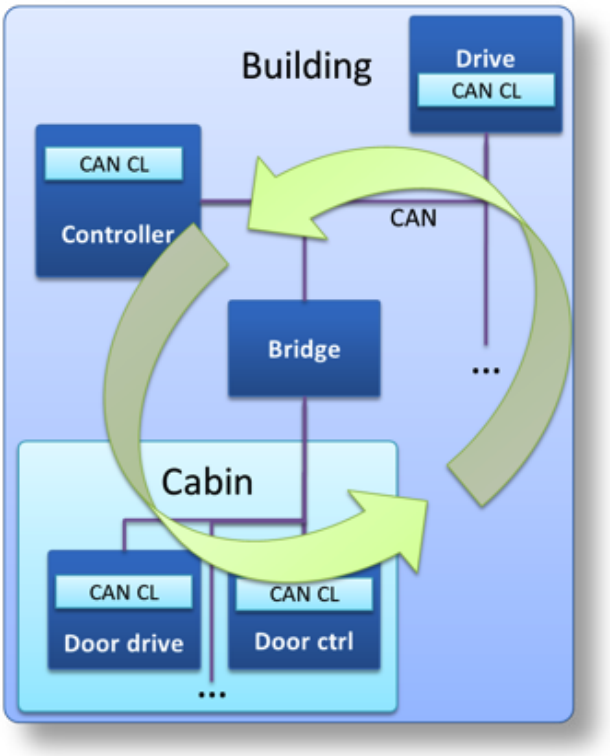

Figure 3: Using a CAN cryptographic layer, devices can be securely grouped and authenticate messages

DTLS-based end-to-end security

End-to-end security in our context refers to a setup where only the endpoints of a communication association have full access to exchanged data. By construction, intermediaries like gateways, bridges, or routers are not in possession of the cryptographic keys used for protection.
In the IT world, Transport Layer Security (TLS) is among the most widely used protocols for end-to-end security between two endpoints [RFC8446]. TLS requires a reliable and stream-oriented transport channel (e.g., TCP) between its client and server. Datagram TLS (DTLS) is a variant of TLS that can cope with unreliable, datagram-like transport (e.g., UDP) to achieve nearly identical security objectives at the cost of slightly higher communication overhead [RFC6347].
During the (D)TLS handshake, client and server can perform mutual authentication using certificates, pre-shared symmetric keys, or passwords, to finally come up with symmetric cryptographic session keys. The following application data exchanges are cryptographically protected using these session keys. Depending on the chosen

cipher suite, the protection operation stretches the length of every application datagram by some dozens of octets.

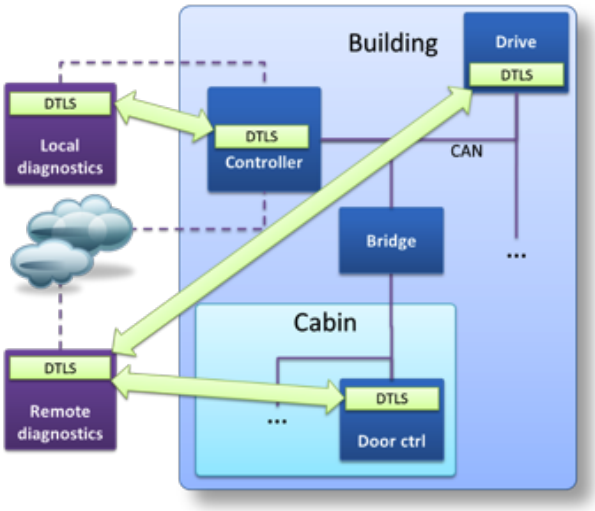

Figure 4: End-to-end security between individual devices using DTLS

Mapping DTLS into the CAN (FD) world is non-trivial, as the characteristics and capabilities of CAN (FD) are quite different from UDP. However, it has already been shown that and how (D)TLS can be used over and within CAN (FD) and CANopen (FD) networks [EWC2019, PDeS2018].In our elevator example, DTLS is best used to protect all remote and temporary system access. At least access to the main controller system should only be granted with DTLS-based authentication. Depending on use cases, one could also envision DTLS-based end-to-end security to individual components in the system. In figure 4 this is illustrated by the DTLS blocks in the drive and door controller. Here an end-to-end connection could also be established directly between a remote diagnostic server and the drive, for example to perform some calibration services.
One solution for DTLS over CAN was previously introduced by [PDeS2018].

Strength & shortcomings

While cryptography-based methods build on the presence of a proof of authenticity, ID guarding relies on the absence of an assertion of non-authenticity. The implication is that the generation, transport, and verification of the cryptographic proof (the cryptographic checksum) imposes overhead on the system's communication and computation resources, which is not

Table 1: Comparison of CAN (FD) security methods

	ID Guard	CAN Security Layer	Remote (D)TLS
Protection provided	Local CAN domain ID ownership and busload limit	Multi domain CAN pairing or grouping	Remote access secure pairing
Relation	1:N	1:N	1:1
Layer	CAN Data Link Layer	Above CAN Data Link Layer	Transport Layer Security
Key usage	None	Symmetric	Asymmetric, Certificate
Root of trust	Not applicable	install at system integration	install at device manufacturing
can leave CAN	No	No	Yes
can cross CAN domains	No	Yes	Yes
MCU processing	None	Lightweight block cipher per transfer	Public key method and block cipher
Overhead, init/background	None	A few messages for key update	About 700 bytes
Overhead, data exchange	None	A few bytes, might result in extra frame	min 128 bit data per exchange
Best for	Frame authentication and flood protect on local CAN domain	App authentication across CAN domains	Remote access authentication

the case for ID guarding solutions. The latter, however, requires that a uniquely defined, legitimate owner for every critical CAN ID exists. The ID Guard of the owner must be permanently online in the network and assert non-authenticity, if it recognizes a frame with a CAN ID, which it owns, but which it did not send.
It is worth noting that cryptography-based methods require the secure distribution and life-cycle management of keys to nodes. The ID guarding configuration and management are typically added to the existing CAN ID assignment/configuration methods.

Comparison

Our comparison in table 1 summarizes essential differences of the three security methods listed in this paper.

Best practice recommendation

There are some general best practice recommendations not directly related to the CAN (FD) security methods.

- When it comes to remote control and service access, do not provide more functionality than needed. If a certain remote-control command is never needed, do not implement it.
- Limit temporary configuration, diagnostic and debug access: Devices not permanently connected to CAN (FD) should not be allowed to just plug-in to any CAN (FD) domain, as shown in figure 5. For temporary devices, use a bridge/gateway as firewall, preferably with DTLS authentication required by the temporary device.

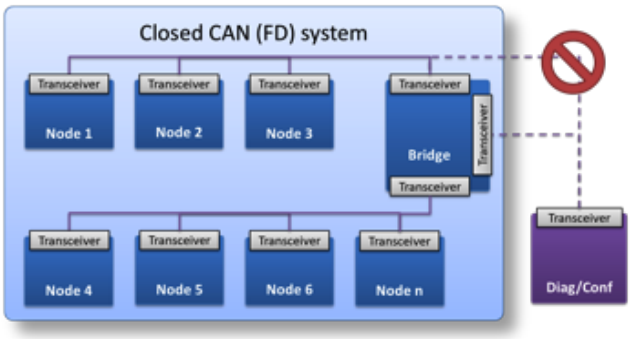

Figure 5: Temporary devices should not be allowed direct access to a CAN domain

- Minimize the number of “access points”: For each CAN (FD) device, check which other “ports” are available into this device and shut these down as far as possible: local debug ports, local bootloader access, other networking interfaces and wireless interfaces. If a device requires other network or wireless interfaces, then these are “gateways” and should use DTLS security.
- For any new development, use a secure MCU: A secure MCU does not only offer hardware support for security algorithms but also features a secure bootloader and a secure code update mechanism. These are essential to fully protect these devices.

Specific CAN (FD) security recommendations and their benefits

Our recommendation for combining the security methods are as follows:

- Protect every CAN domain with minimum ID guarding security in all devices. This ensures that at no additional resource cost (bandwidth, processing, delay) only specified communications are possible and flooding attacks fail.
- For all devices that communicate across multiple domains or have interfaces to other networking technologies, implement a CAN cryptographic layer. Using a secure heartbeat ensures that applications of these devices are authenticated. Using secured messaging ensures secure communication across CAN domains.
- For all “remote access” options including temporary diagnostic devices or Internet connected maintenance ports implement DTLS security. This ensures that remote access is properly secured.

The elevator system shown in figure 6 combines the different security methods introduced.

- DTLS is used to authenticate all remote or temporary access paths to the system.

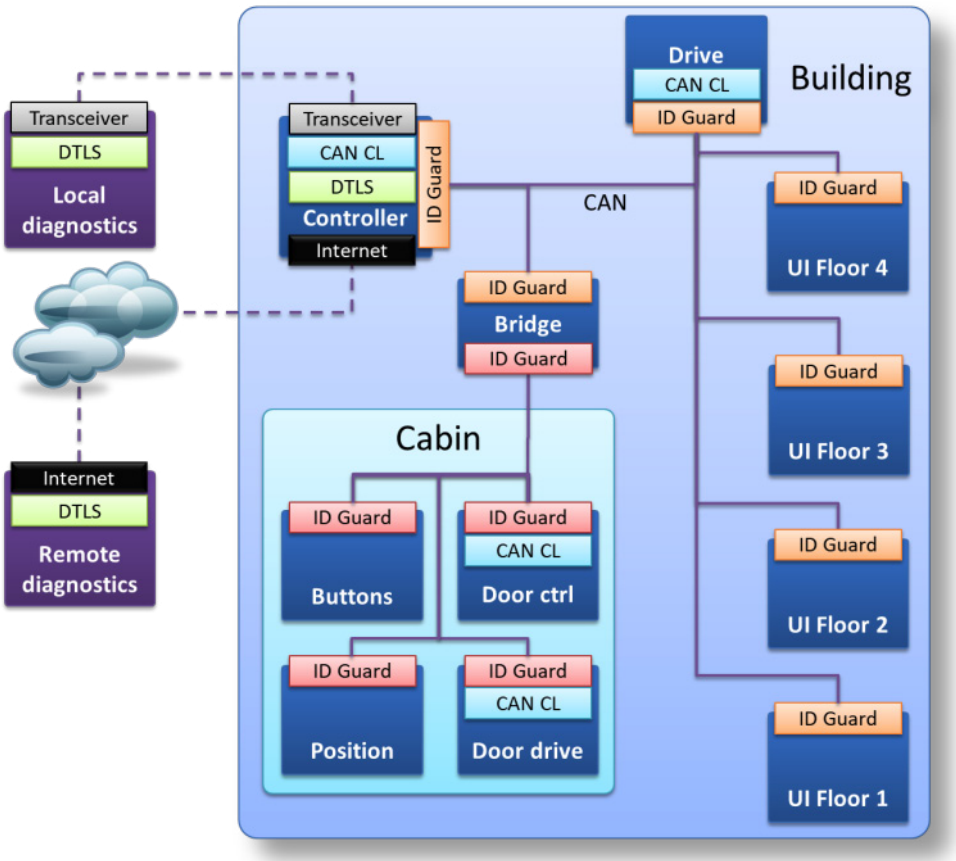

Figure 6: Combining the different security methods available

- A CAN cryptographic layer is used to securely group selected essential devices.
- CAN ID guarding is used to protect all devices of each domain.

Notes on the life cycle of secure CAN (FD) systems:

Development:
Where possible, use secure MCUs for new designs. Code and keys stored in MCUs without specific security features could potentially be extracted, there are numerous commercial extraction services available on the market.
On lowest level, protect your code updates independently from the CAN (FD) communication security mechanisms shown here. Secure MCUs have a root of trust, perform your code updates within that root of trust. Where possible, disable unused code update channels (USB, other serial protocols).

Deployment:
For the deployment of a secure CAN (FD) system you need to ensure that there is a "trusted environment" when private and symmetric keys are installed. Symmetric keys for the CAN cryptographic layer are typically assigned during system integration, upon first power-up of the system. If that does not happen in a trusted environment, consider DTLS as a protection for the initial key distribution.

Maintenance:
Diagnostic ports should not give direct access to all internal networks, but to a bridge/gateway only. We recommend that all diagnostic tools use DTLS to identify themselves to that bridge/gateway, before they get access to the system.

Demonstrator

Our updated demonstrator setup from [EWC2019] now uses two CAN domains as shown in figure 7.
One CAN domain is a CANopen FD network with two I/O modules for buttons and LEDs and a control and visualizer unit with screen and an LED matrix. A bridge separates this CAN domain from a second that has a Bluetooth gateway connected.

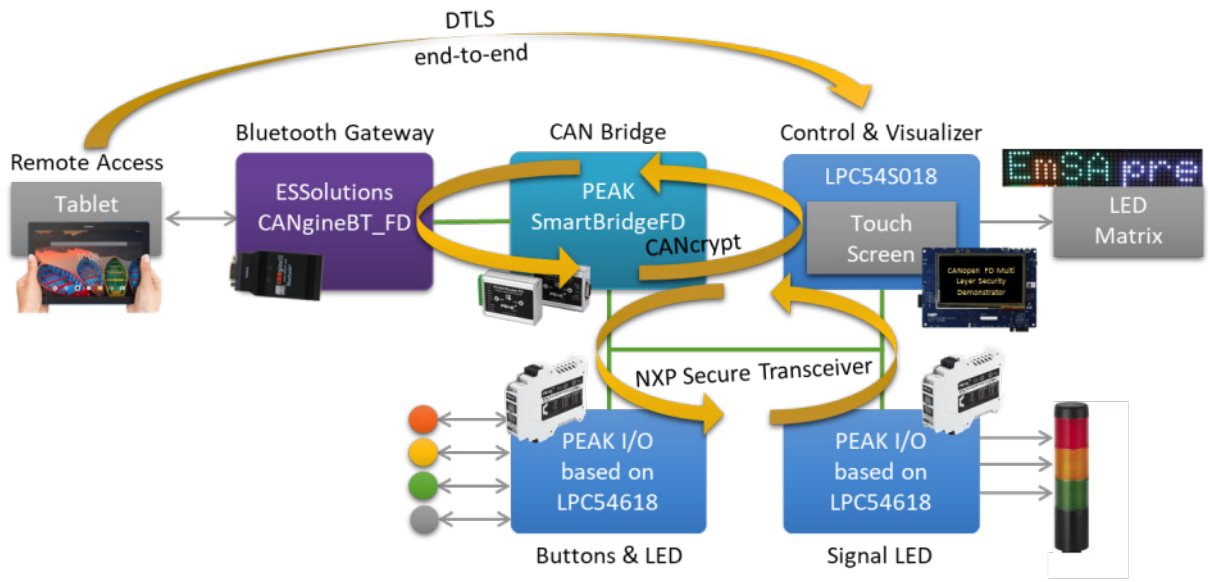

Figure 7: Demonstrator setup

The ID guarding is provided by the NXP TJA115x Secure CAN Transceiver and protects all devices in the main CAN domain with the Controller, the I/O and the bridge. No firmware/software changes are required, only the original transceivers are replaced. The initial, default setup of the ID guarding filters typically happens during manufacturing of a device. The ID guarding filters get securely re-configured during system integration, if CAN (FD) IDs or node IDs have to be (re-) assigned.

The CAN cryptographic layer is implemented using CANcrypt and is used by the Controller, the bridge and the Bluetooth Gateway. In this use case, it ensures that these three devices form a secure group; none of the three could be replaced by an attacker not knowing the shared symmetric key. The initial key assignment happens via CAN on first time system integration. The system specific initial key is negotiated between the participating devices.

DTLS is used to provide end-to-end security between a tablet (BT connection to BT gateway) and the control unit. Active control commands for the screens and displays are only processed after authentication. Upon production, the control unit is equipped with the root keys.

Summary & Outlook

In this contribution, we showed how the three main CAN (FD) security methods can complement each other. Still, the final selection of concrete security methods continues to be largely application specific. Using CAN (FD) nodes with MCUs that do not have dedicated security features built-in is still common practice, so the security level one can reach with such devices will always be limited.

The further evolvement of security features in CAN (FD) systems will strongly depend on the general “security awareness” of manufacturers and developers of CAN (FD) based systems. With more and more systems offering wireless interfaces or Internet connections, the risk of “unauthorized access” is on the rise. The amount of damage such an intrusion might cause can be narrowed effectively using the techniques shown here.

Olaf Pfeiffer
Embedded Systems Academy GmbH
Bahnhofstr. 17
DE-30890 Barsinghausen
Tel.: +49-5105-582-7897
Fax +49-5105-584-0735
opfeiffer@esacademy.de
www.em-sa.com

Thierry Walrant
NXP Semiconductors
Interleuvenlaan 80
BE-3001 Leuven
Tel.: +32 16 390 897
Fax: +32 16 390 855
thierry.walrant@nxp.com
Website: www.nxp.com

Georg Olma
NXP Semiconductors Germany GmbH
Schatzbogen 7
DE-81829 München
www.nxp.com

Andreas Walz
Hochschule Offenburg
Badstr. 24
DE-77652 Offenburg
http://ivesk.hs-offenburg.de

Dr. Axel Sikora
Hochschule Offenburg
Badstr. 24
DE-77652 Offenburg
http://ivesk.hs-offenburg.de

References

[iCC2017El] B.Elend, T. Adamson, “Cyber Security enhancing CAN transceivers”, 16th international CAN Conference 2017, March 2017, Nuremberg, Germany

[iCC2017Pf] O.Pfeiffer, C. Keydel, “Scalable CAN security for CAN, CANopen and other protocols”, 16th international CAN Conference 2017, March 2017, Nuremberg, Germany

[PDeS2018] A. Yushev, M. Barghash, M. P. Nguyen, A. Walz, A. Sikora, „TLS-over-CAN: An Experimental Study of Internet-Grade End-to-End Communication Security for CAN Networks“, in Proceedings of the 15th International Conference on Programmable Devices and Embedded Systems (PDeS 2018), 23 - 25 May 2018, Ostrava, Czech Republic

[EWC2019] O. Pfeiffer, C. Keydel, A. Walz, A. Diehm, and A. Sikora, „Securing All Network Layers of CAN (FD) Communication,“ Embedded World Conference 2019, 26 – 28 February 2019, Nuremberg, Germany