Implementation requirements for secured gateways

The concept of a security gateway is trivial but the implementation is not: a gateway is not just a functional device that filters CAN data frames, but must operate so that the resulting traffic patterns on the trustworthy CAN network meet all real-time requirements and operate securely at all times. The requirements described in this article are designed to ensure that.

(Source: Adobe Stock)

In road vehicles, there are installed multiple CAN-based in-vehicle network segments for real-time control purposes. Between electronic control units (ECUs) CAN data frames containing sensor and actuator data are exchanged using a publish-subscribe paradigm. Some of the data is sent to – and comes from – inherently untrustworthy devices such as ECUs that are cellularly connected or aftermarket, third party provided.

This is clearly a security issue: these devices cannot in general be trusted and being given direct access to a CAN network would allow all kinds of attacks on the CAN network and hence the vehicle. The approach described here is to create a CAN network for untrustworthy devices separated from internal vehicle communications via a security gateway that forwards traffic between the trusted internal CAN networks and the untrustworthy external CAN network.

Rationale

The National Motor Freight Traffic Association [1] (NMFTA) is a nonprofit motor freight carrier organization in the U.S.A. representing over 500 carriers collectively operating over 200 000 commercial road vehicles. One key focus of the organization is protecting their members' commercial vehicles from the ever-evolving cyber-threat landscape. As such, the organization has been a pioneer in conducting and supporting security research in the transportation domain since 2015.

Because CAN enables robust, low-latency communication among many ECUs at once electronic architectures in commercial vehicles, just as for passenger cars, have utilized CAN as the foundational data link layer protocol for inter-ECU communication. In the commercial vehicle space, application messages have historically been standardized by SAE J1939 to allow 'plug and play' of device suppliers' systems (Figure 1). This has allowed for ultimate configuration and customization to optimize fleet operations, with a diverse supplier industry that has encouraged innovation.

For all its benefits, CAN was never designed with security in mind: communication has relied on each node acting in good faith. A plethora of research has demonstrated CAN is vulnerable to attacks, both at the frame level (such as spoofing fake data frames and eavesdropping on sensitive data) ▷

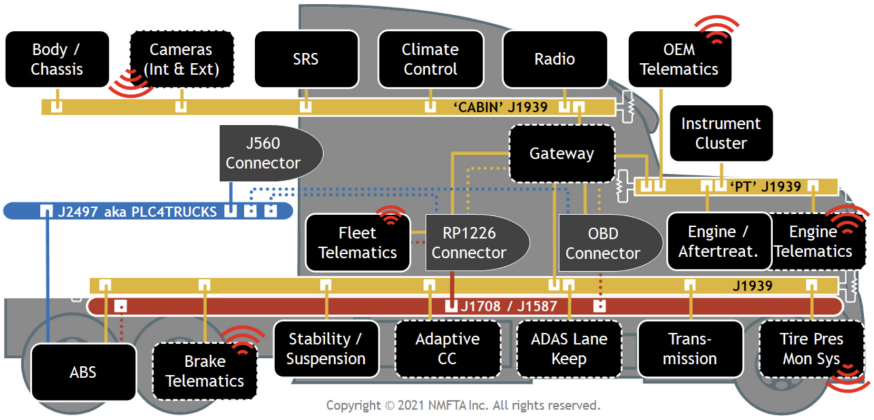

Figure 1: Typical J1939 in-vehicle network architecture (Source: NMFTA)

and at the protocol level itself (such as the Bus Off attack), undermining all three aspects of the Confidentiality/Integrity/Availability (CIA) Triad security objectives. If vehicle electronics were largely isolated and air-gapped systems, these security issues might pose minimal overall security risk. Unfortunately, this is no longer – if it ever was – the case. For one, existing technology such as trailer brake ECUs, which have been a requirement on commercial trailers since the late 1990s, have recently been demonstrated as remote attack vectors [2] and for leaking information via power line communication (PLC) networks. Additionally, third party remote fleet tracking and telematics has proliferated within the industry to track and optimize operations as well as monitor assets for uptime and maintenance. Since 2017, the Federal Motor Carrier Safety Administration (FMCSA) has required connected devices in the form of electronic logging devices (ELDs) to track drivers' hours of service [3].

Clearly, commercial vehicles on the road today have multiple remote vectors that could serve as entry points for an attacker to access and affect the safety critical operations of the vehicle. At the same time, these potential attack vectors provide critical functionality for operators and fleets and cannot be removed entirely. Instead, one method to effectively reduce risk in the case of a remote compromise is to introduce a device to partition any untrusthworthy, connected devices from the safety critical controls of a vehicle: a CAN based security gateway. Gateways have the purpose of physically separating a device with potential risk from the rest of the vehicle system. Messages and data can be defined in a bi-directional manner to ensure no unintended or malicious messages are transported across the gateway boundary.

While CAN-based gateways are already implemented in many vehicle architectures, to date the authors are not aware of public, comprehensive cybersecurity requirements to define such a device. The NMFTA has led a working group to develop such requirements. The main intention of the requirements is for NMFTA member fleets to use as a tool for procurement: to confirm OEM (original equipment manufacturer) vehicles provide protections before purchasing. They could also be used by OEMs and aftermarket suppliers alike as a baseline to develop secure, industry leading gateways.

Top-level requirements

The NMFTA security gateway requirements define two domains: the trustworthy network domain (TND) where the vehicle control systems operate on CAN networks, and the untrustworthy network domain (UND) which inherently cannot be trusted (for example, third-party wirelessly connected devices containing complex software). A security gateway is defined to connect the UND with the TND (normal CAN gateways operating entirely within the TND are not covered by these requirements but obviously nothing prevents a security gateway from being used in that role). There are three top-level requirements for a security gateway connecting the UND with the TND.

1. A security gateway must restrict CAN traffic in each direction to only defined traffic for each operational mode. These modes might include over-the-air downloads taking place and diagnostics sessions (it is not required that modes are mutually exclusive, merely that there is an ability to define what is and is not legal traffic for a given situation). Restricted and defined traffic prevents a compromised, untrustworthy device from both directly spoofing application data frames that originate on TND and tampering with internal diagnostic interfaces.
2. Communication within the TND must not be disrupted by traffic from the UND. This is a less obvious requirement but just as important: the TND forms part of a distributed real-time vehicle control system where the message timing is just as important as message contents. If traffic from the UND exceeds a defined usage there can be serious consequences for the latencies of CAN data frames in the TND, such as timeouts causing false error warnings, buffer overflows and dropped frames and even excessive CPU (central processing unit) load with potential to cause erratic system disruption in receivers.
3. The gateway itself must be secure. Clearly, the security gateway itself needs to be controlled. For example, being instructed to switch between operational modes, or being re-programmed with a new configuration, or even extracting and clearing event logs. This control must be via secure mechanisms to prevent compromising the protections a security gateway seeks to provide.

These top-level requirements are refined into multiple, more specific requirements [4].

Defining traffic patterns

Traffic definitions for a security gateway define for each operating mode specific frames in one domain (TND or UND) that will be forwarded to the other domain. The traffic patterns define not only CAN IDs but also how the CAN data field is handled. The J1939 application layer defines that for a Parameter Group (PG) with a given PGN (PG number) mapped into the CAN ID field, the payload contains known suspect parameters (SP), sometimes called in laboratory slang *signals*. The J1939 traffic definition may restrict the parameters in a payload on a *need-to-know* basis. For example, if an application in the UND requires access to a CAN data frame for a specific parameter then the other parameters in the CAN data frame ▷

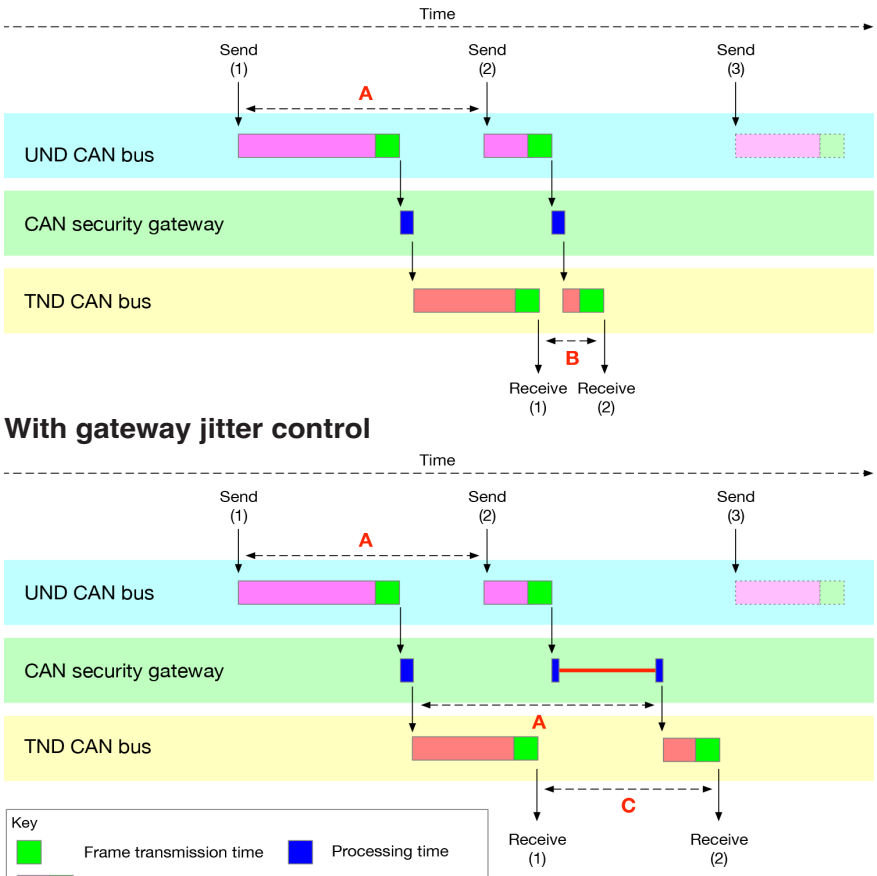

Figure 2: Jitter control in a security gateway (Source: Ken Tindell)

may be zeroed out by the security gateway to avoid inadvertently disclosing proprietary or confidential information.

A traffic definition also specifies a real-time frame rate for each frame to be forwarded so that a CAN data frame is only passed through if that rate is not exceeded. This is the starting point for guaranteeing the timing behavior of the TND: the real-time data frame rates can be used in CAN network schedulability analysis [5] calculations to determine worst-case latencies. This analysis can guarantee all TND CAN data frames will arrive on time no matter how often CAN data frames are sent within the UND. It also allows the buffer space and CPU time dealing with CAN data frames to be bounded and so prevent frame losses and CPU overloads.

CAN frame handling requirements

A crucial requirement for any CAN gateway is that CAN data frames are handled properly: applications built on top of CAN often rely on the network behaving properly (although sometimes unknowingly). For example, multi-frame segments like ISO-TP [6] require frames that form the segment are not dropped and are not transmitted out of order. Another example is with typical cryptographic schemes for CAN networks, like the CryptoCAN [7] scheme of Canis Labs: a chained block cipher mode relies on segments and messages being sent in order. If the security gateway does not handle CAN data frames correctly then the overall system could fail.

One of the key properties of CAN is atomic broadcast/multicast: a CAN data frame that is successfully sent will have been received at all receivers connected to the network. With other protocols like Ethernet, frames that contain errors are just discarded, and atomicity is implemented in software, which is far from easy. Applications using CAN often rely on this property – it's a key feature of the publish/subscribe communications model of CAN – and a security gateway is required to maintain this property. Because the transmission on one side of a gateway has completed before the frame can be sent on the other side, a gateway inherits an obligation (if the frame is legal) to always transmit that frame on the other side. This means that a gateway is required to have sufficient memory for buffers such that no frame will ever be dropped due to a buffer overflowing. Fortunately, it is possible to statically determine the maximum buffer space needed after putting an upper bound on the frame latencies for frames waiting to be sent on the destination CAN network segment.

A security gateway is also required to transmit frames without priority inversion: this is a problem where the latencies of urgent (and hence high priority) activities are normally short but intermittently and unpredictably become very large. This can induce further problems (such as triggering timeouts leading to spurious fault handling) and must be avoided. Priority inversion can occur in any system scheduled by priorities and most famously occurred during NASA's Mars Pathfinder Mission [8]. It can nearly always be avoided by writing CAN driver software correctly so that there is a priority-ordered (i.e. ordered by CAN ID) queue of frames to be transmitted, where the driver software ensures that the highest priority CAN data frame is always entered into CAN arbitration whenever it starts.

There is a requirement for a gateway to transmit CAN data frames in order (for example, so that ISO-TP segments are not corrupted). This at first sight may appear to conflict with the requirement for no priority inversion – in effect it is mandating FIFO frame transmission. But this is not so: the requirement is that FIFO order is required for multiple frames of the same sequence with respect to each other. This cannot be left to the hardware: with most CAN controller hardware, if two frames with same ID are sent at the same time then the hardware will arbitrarily pick a frame to send first, which would violate the requirement to transmit frames in order. So, a typical way to meet this frame ordering requirement is to put related frames (typically those with the same ID) into their own FIFO, and for this FIFO to feed the priority-ordered frame queue.

Another requirement for frame handling is to control the jitter of a CAN data frame: frames can be queued on a CAN network with a precisely regular periodicity (matching the defined rate of the frame) but there will be variations in latency and so the relative arrival time of the frame will vary. If the frame were immediately placed into an outgoing CAN controller then the ▷

frame would inherit jitter in the queueing time, and by the time it was transmitted on the network the variability would be even larger, potentially so large that two CAN data frames with the same CAN ID could arrive back-to-back, and this could cause a receiver to overwrite one frame with another before the first frame could be handled – effectively dropping a frame and violating the requirement not to drop frames. A security gateway is required to limit this jitter: if a CAN data frame arrives on one CAN network sooner than its defined period [9] since the previous arrival time then it must be held back and only queued on the outgoing CAN network segment when this elapsed time has reached its period (Figure 2).

This requirement does mean that the average latency through a security gateway is increased, but average case for real-time control systems is not very important: it is the worst-case latency that matters. This jitter control means that CAN schedulability analysis on a TND can take place without knowledge of the timing properties of the UND and therefore the UND cannot disrupt the timing behavior of the TND.

Secure gateway control

A security gateway itself needs to be secure. Aside from normal protections common to most embedded systems (like firmware secure boot), the control of the gateway must be secure. Control includes programming a configuration, setting operational modes, and clearing event logs. There is a requirement that gateway control must be authenticated. This could be done by cryptographically secure communications. For example, a cryptographically secure message between a driver's touch screen display and the security gateway. An alternative is to use a physical interlock switch that prevents changes until a human has turned a key or flipped a switch.

For cryptographically secure messaging, a security gateway must store keys securely (so that not even software in the gateway can read them) and then execute cryptographic operations with these keys. This can be done by using a hardware security module (HSM) and the automotive industry has defined its own Secure Hardware Extensions (SHE) standard for HSMs. A secure challenge-response protocol using HSM operations can provide mutual authentication between a control tool and a security gateway.

Discussion

There are several ways to instantiate the NMFTA security gateway requirements. One is by using dedicated firmware on a regular microcontroller with multiple CAN controllers (either within an existing ECU with the necessary CAN connections or in a dedicated security ECU). Another is by using a Hardware Security Gateway Module (HGM): dedicated silicon IP (intellectual property) cores (like an HSM) that implements a complete security gateway (including CAN controllers) in silicon, deployed as a stand-alone chip or within a System-on-Chip (SoC) package and designed into an ECU.

The security gateway functionality may be provided by an OEM to protect certain safety critical systems or provided by a dedicated third-party interface for aftermarket installation of telematics services, ELDs, etc. This might be implemented as a dedicated security ECU or by augmenting the functionality of an existing ECU design. A dedicated security gateway may also be provided as a third-party aftermarket option to retrofit into existing vehicles that are not equipped with an OEM-provided security gateway.

The NMFTA security gateway requirements described here are necessary but not necessarily sufficient. It is recognized that there are additional security requirements needed to make a comprehensive set of requirements for a security gateway and the NMFTA will be tackling these in the Truck Matrix security requirements work [10]; Furthermore, they are primarily intended as tool for procurement by fleets – but they may not be sufficient for all product security applications of a security gateway. For example, there may be frames in the TND that use cryptographic authentication and payload encryption to prevent physical access spoofing and snooping attacks, and a security gateway may need to encrypt, decrypt (and potentially re-encrypt) messages to and from this domain. In these cases, the NMFTA security gateway requirements should be seen as a starting point for a more comprehensive functional specification – one perhaps contributing to a wider industry standardization process. Such a standard could offer not just easier procurement but also cost savings by creating a market for standardized security gateway implementations and interoperable tools. ◀

References

[1] More about NMFTA cybersecurity is available at https://nmfta.org/cybersecurity
[2] CVE-2022-25922 and CVE-2022-26131
[3] https://eld.fmcsa.dot.gov
[4] https://nmfta-repo.github.io/vcr-experiment/vcr-experiment/01_gateways.html
[5] Using schedulability analysis on CAN network was pioneered by Volvo in the 1990s. Today, it is used by many manufacturers as part of a design-for-correctness engineering approach to CAN networking
[6] ISO 15765-2
[7] CAN Newsletter issue 4/2022
[8] Blog post at https://kentindell.github.io/2020/06/29/can-priority-inversion
[9] Period is the inverse of rate, so a 100-Hz frame has a period of 10 ms
[10] See the draft here: https://github.com/nmfta-repo/nmfta-vehicle_cybersecurity_requirements



Authors

Ken Tindell, Canis Automotive Labs
ken@canislabs.com
canislabs.com

Ben Gardiner, NMFTA
Ben.Gardiner@nmfta.org
www.nmfta.org

John Maag, Cummins
john.maag@cummins.com
www.cummins.com