CAN security: how small can we go?

What kind of CAN security can still be added to a deployed CAN system if the processors have only medium performance and only adding a few kilobytes of extra code is possible?

Adding security information to a CAN frame (Source: EmSA)

This article originally appeared in the June issue of the CAN Newsletter magazine 2019. This is just an excerpt.

In past articles, the authors have introduced various security methods which all had in common to work for systems and devices of all sizes and hardware capabilities. Along with the needed amount of flexibility, however, typically come higher resource requirements. A product that includes CAN and that has been sold for many years may not have the amount of resources needed for extra security features to spare. In this article we examine what kind of CAN security we can still add to a deployed CAN system if the processors have only medium performance and we can only add a few kilobytes of extra code.

Motivation

Some things appear to have not changed significantly in the past 20 years of Embedded Systems programming. Back then we would start developing minimal solutions for clients that wanted to add CANopen using “as few resources as possible”. Today, clients want to add CAN security to an already deployed system and again, often with only minimal resources available. Same situation, different technology. We introduced the CANcrypt security framework in previous articles. The framework offers enough functionality and flexibility for a wide range of platforms and security needs.

The 18-bit Micro CANcrypt security record (Source: EmSA)

However, especially in applications where authentication for as many CAN frames as possible is the number one requirement but encryption is not needed, an alternative, cut-down Micro CANcrypt implementation targeting low-footprint environment can fit the bill much better. At the same time, the authors thought of better ways to apply CANcrypt methods to classic CANopen and CANopen FD. In its original incarnation, securing CANopen messages with CANcrypt would always need either a second message or multiple reserved bytes in the data payload while Micro CANcrypt will attempt to stay as close to unencrypted CANopen as possible.

Micro CANcrypt optimizations

The biggest change compared to unsecured CAN communications is the added security information, and the question is where in the CAN frames we want to put it. In networks that only use 11-bit-identifier CAN frames, like virtually all CANopen systems do, it is convenient if secure frames use a 29-bit CAN identifier instead, as illustrated in Figure 1. In the available extra 18-bits long “security record” we can then put a 10-bit signature and some control information. This method greatly simplifies mixing non-secure and secure CAN communications – a secure frame then still uses the same lower 11-bit portion of the 29-bit CAN identifier as the unsecured frame would, and the added security record can be easily recognized. Figure 2 shows the security information added to every secure message in more detail.

The extended security record (Source: EmSA)

The record comprises a 2-bit truncated key refresh counter, a 6-bit truncated timer value and the 10-bit Micro CANcrypt signature. As all devices synchronize their refresh counter and timer locally, the truncated information is enough for receivers to internally maintain the full counter and timer values. Figure 3 shows how Micro CANcrypt devices exchange event-specific information. The record uses five bytes which are either transmitted in dedicated CAN frames only for Micro CANcrypt events, or becomes integrated into a higher-layer protocol. In CANopen for example, these five bytes fit nicely into the manufacturer-specific part of the emergency message.

Looking at the keys used for authentication, we also find optimization potential: Out of the full key hierarchy that is part of CANcrypt, what is essential is that the participating devices must support only at least one permanent shared symmetric key and one last-saved session key. The permanent key is only used once in the beginning to generate a new session key which is then used for all further security algorithms, thus minimizing the use and possible exposure of the permanent key. The core security algorithms use a lightweight block cipher with 64-bit blocks and 128-bit keys. Our first demo implementations use XTEA-64 or, alternatively, Speck-64. Finally, Micro CANcrypt introduces a new secure key sync cycle, which is a simplified variation of the CANcrypt secure heartbeat.

Synchronized, shared parameters, and secrets (Source: EmSA)

Micro CANcrypt secure key sync cycle

The original CANcrypt mechanism for the secure heartbeat offers too much flexibility (between 2 and 15 nodes may participate) for an implementation with limited resources. In Micro CANcrypt, four devices actively maintain a dynamic key, each of them using one grouping / key refresh message. If a network has fewer than four devices, a single device can also produce the CANcrypt messages for two. The new secure key sync cycle therefore always has two to four active participants while all others are passive participants. Both active and passive participants become part of a secure group where all parties consume the secure key sync and know the shared secrets (symmetric key, timer, counter), allowing them to receive and generate secured messages.

Each secure key sync cycle produces a random initialization vector which is then used to generate the current rolling dynamic key from the session key. With a new secure key sync cycle happening every second, the maximum lifetime of the dynamic key is reduced to two seconds, still leaving some time to handle errors. To protect from replay attacks, CANcrypt uses a message counter. However, tracking an individual counter for each CAN identifier received or transmitted requires quite a few resources. Therefore, Micro CANcrypt uses a synchronized timer value instead. A 16-bit timer counting five-millisecond-increments is synchronized as part of the secure key sync cycle. Figure 4 summarizes all active synchronized values.

Figure 5 illustrates how four event messages use the extended security record to share information. Here the extended security record contains a 16-bit timer and a 16-bit random value. These synchronized messages are used once per second to share / create an initialization vector (IV) for a dynamic, current key from the session key and to synchronize a 16-bit timer value and an 8-bit key refresh counter. A full block cipher cycle is used to generate the dynamic key from a shared symmetric permanent key using the IV generated in each cycle.

If you want to continue reading this article, you can download the PDF of Mr. Olaf Pfeiffer and Mr. Christian Keydel from EmSA. Or you download the full magazine. This is free-of-charge.

CW