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?

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.

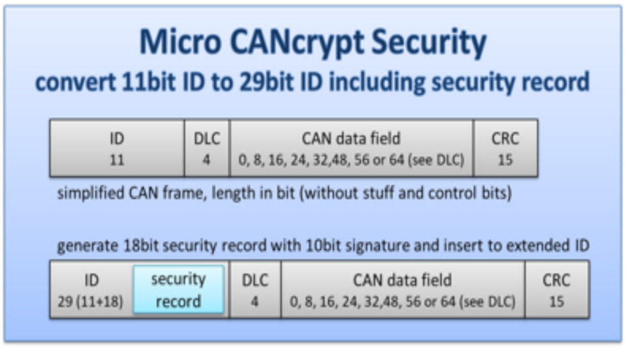
Micro CANcrypt Security
convert 11bit ID to 29bit ID including security record
ID 11
DLC 4
CAN data field 0, 8, 16, 24, 32,48, 56 or 64 (see DLC)
CRC 15
simplified CAN frame, length in bit (without stuff and control bits)
generate 18bit security record with 10bit signature and insert to extended ID
ID 29 (11+18)
security record
DLC 4
CAN data field 0, 8, 16, 24, 32,48, 56 or 64 (see DLC)
CRC 15

Figure 1: Adding security information to a CAN frame (Source: Emsa)

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. 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 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 ▷

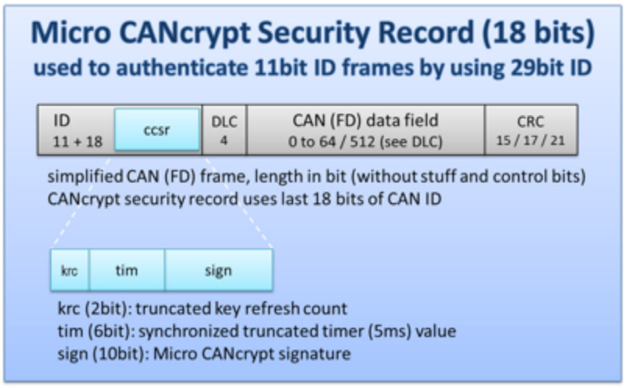
Micro CANcrypt Security Record (18 bits)
used to authenticate 11bit ID frames by using 29bit ID
ID 11 + 18
ccsr
DLC 4
CAN (FD) data field 0 to 64 / 512 (see DLC)
CRC 15 / 17 / 21
simplified CAN (FD) frame, length in bit (without stuff and control bits)
CANcrypt security record uses last 18 bits of CAN ID
krc
tim
sign
krc (2bit): truncated key refresh count
tim (6bit): synchronized truncated timer (5ms) value
sign (10bit): Micro CANcrypt signature

Figure 2: The 18-bit Micro CANcrypt security record (Source: Emsa)

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.

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. ▷

esd electronics

All you CAN plug

CANopen FD   CAN FD

CAN / CAN FD Interfaces

Product Line 402 with Highspeed FPGA

- **Various Form Factors**

  PCI, PCI Express® Mini, PCI Express®, CompactPCI®, CompactPCI® serial, XMC and PMC,USB, etc.

- **Highspeed FPGA Design**

  esdACC: most modern FPGA CAN-Controller for up to 4 channels with DMA

- **Protocol Stacks**

  CANopen©, J1939 and ARINC 825

- **Software Driver Support**

  Windows©, Linux©, optional Realtime OS: QNX©, RTX, VxWorks©, etc.

all about automation essen

June, 5. - 6., 2019
hall 1, booth 625

esd electronics gmbh
Vahrenwalder Straße 207 | D-30165 Hannover
Tel.: +49(0)511 372 98-0
info@esd.eu | www.esd.eu

Quality Products - Made in Germany

esd electronics, Inc.
70 Federal Street - Suite #2
Greenfield, MA 01301
Phone: 413-772-3170
www.esd-electronics.us

www.esd.eu

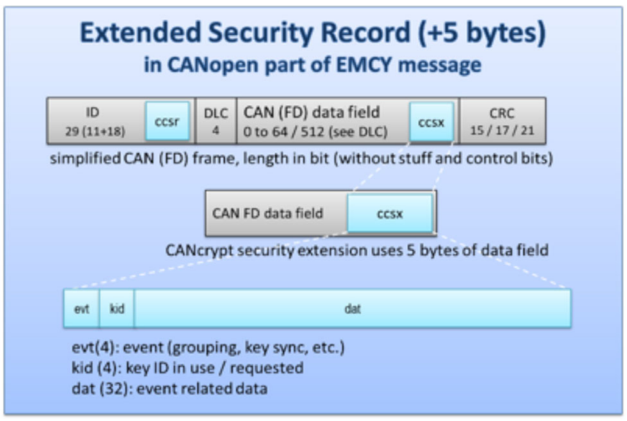
Extended Security Record (+5 bytes)
in CANopen part of EMCY message
ID 29 (11+18)
ccsr
DLC 4
CAN (FD) data field 0 to 64 / 512 (see DLC)
ccsx
CRC 15 / 17 / 21
simplified CAN (FD) frame, length in bit (without stuff and control bits)
CAN FD data field
ccsx
CANcrypt security extension uses 5 bytes of data field
evt
kid
dat
evt(4): event (grouping, key sync, etc.)
kid (4): key ID in use / requested
dat (32): event related data

Figure 3: The extended security record (Source: Emsa)

Signature generation performance requirements

One goal of Micro CANcrypt is to be able to perform a signature generation or verification for every CAN frame processed by a device. At a CAN bitrate of 125 Kbit/s, potential throughput is about one CAN frame per millisecond. At 500 Kbit/s, it can be four frames per millisecond.

A rough estimation: Let's assume that a CPU may use 50 % of its CPU time for CAN processing and that for signature calculation a maximum of 10 % additional CPU time is allowed, then this translates to:

- 125 Kbit/s: 1 ms per CAN frame, CPU time 500 μs, 10 % translates to 50 μs available for signature calculation.
- 500 Kbit/s: 250 μs per frame, CPU time 125 μs, 10 % translates to 12,5 μs available for signature calculation.

These estimates already show that there is not always enough CPU time to execute a full lightweight block cipher with all rounds for every signature in every CAN frame, as not all micro-controllers in use will have the needed performance. However, when keeping in mind, that

- a pseudo one-time pad is used that changes every 5 ms

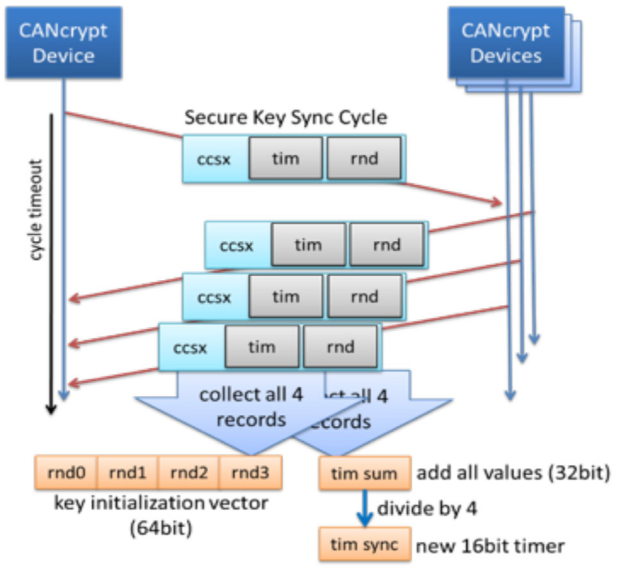

Figure 5: The secure key sync cycle (Source: Emsa)

- the current dynamic key is based on a session key
- the current dynamic key is valid for a maximum of two seconds only, therefore the number of messages communicated in that time frame (= samples to attacker) is limited
- as shown below, attacker will never see all data, only portions of it (see method below)

It means that the method chosen to generate a digital signature does not need to be protected to the highest extent. The Micro CANcrypt method to generate a signature is illustrated in Figure 6. The steps to calculate the transmit-side signature are:

1. Generate a 64-bit checksum of the CAN identifier, DLC and data.
2. Take the current 16-bit timer, counter and dynamic key to create a pseudo one-time pad, using only a portion of the recommended rounds of the block cipher (default: one quarter).

▷

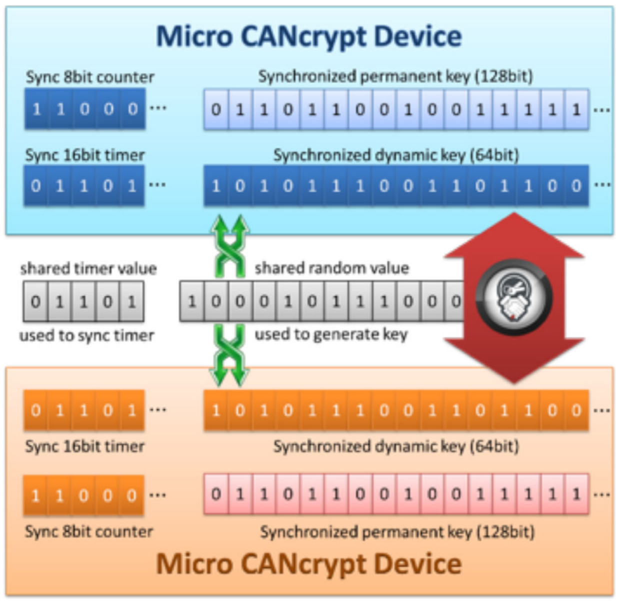

Figure 4: Synchronized, shared parameters, and secrets (Source: Emsa)

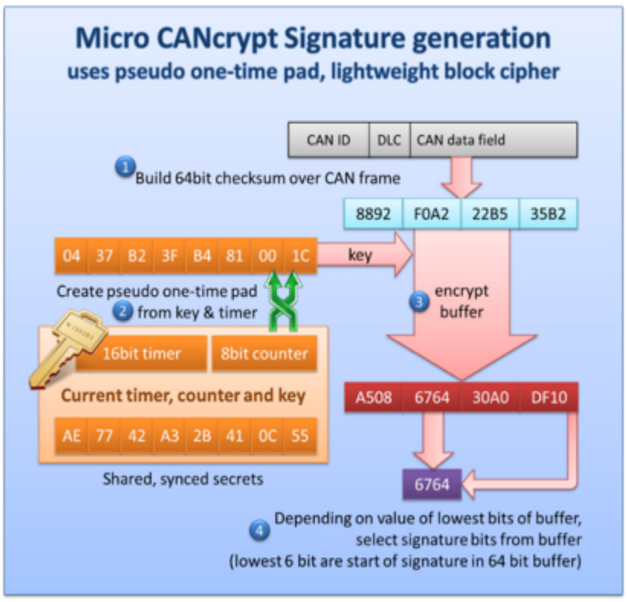
Micro CANcrypt Signature generation
uses pseudo one-time pad, lightweight block cipher
CAN ID
DLC
CAN data field
1 Build 64bit checksum over CAN frame
8892 F0A2 22B5 35B2
04 37 B2 3F B4 81 00 1C
key
Create pseudo one-time pad
2 from key & timer
3 encrypt buffer
16bit timer
8bit counter
Current timer, counter and key
AE 77 42 A3 2B 41 0C 55
Shared, synced secrets
A508 6764 30A0 DF10
6764
4 Depending on value of lowest bits of buffer, select signature bits from buffer
(lowest 6 bit are start of signature in 64 bit buffer)

Figure 6: Signature generation (Source: Emsa)

3. Encrypt the buffer using only a portion of the recommended rounds of the block cipher (default: one quarter).
4. We now have a 64-bit signature buffer which needs to be reduced to 10 bits. The value of the lowest six bits of the buffer is taken as the bit position of the 10-bit slice from the buffer that is used as Micro CANcrypt Signature for this CAN frame.

To verify the signature during secure receive:

- First, verify that the key refresh counter received matches the local key counter or is from previous cycle.
- Second, verify that the received timestamp is not older than 25 ms compared to the local timer.

Then perform the same steps as above to generate the receive-side Micro CANcrypt signature. For the generation of the one-time pad use a full timer value, comprising the lower-16-bit timer value received with the frame and the upper bits of the local timer. If it matches with the received transmit-side signature, the frame is authentic.

Overview of resources used

First prototype implementations of Micro CANcrypt are being done on an NXP LPC11Cxx (ARM Cortex-M0, 48 Mhz) for Classical CAN and an NXP LPC54xxx for CAN FD. For a full integration demo, we use the multilayer security demonstrator, adding Micro CANopen security not only to evaluation boards, but also to commercial CANopen (FD) modules from Peak-System Technik and Embedded Systems Solutions.

Additional CAN traffic and bandwidth

The secure key sync cycle uses four extended security records, in CANopen integrated into the manufacturer-specific field of the emergency message and using an EMCY error code below 100_h to indicate no error. These are used once per second normally or twice per second on failure / recovery. At 125 Kbit/s, the generated bus load for this mechanism is less than 0,01 %.

All existing 11-bit CAN identifier communication that requires security now uses a 29-bit CAN identifier which generates an overhead of about 20 bits per CAN frame, assuming an average of 2 stuffing bits. With CAN frames being some 60 bits to 125 bits long (incl. stuffing bits, assuming DLC between 1 and 8), the overhead calculates to between 16 % and 33 % for all secured messages.

Memory usage

Code size of the Micro CANcrypt specific secure grouping mechanism is below 2000 bytes for the Cortex-M0, using a Keil/ARM Realview compiler at its highest optimization level. Added to this is some “glue” code to interface with the driver level, the size of which highly depends on driver specifics. RAM and stack usage

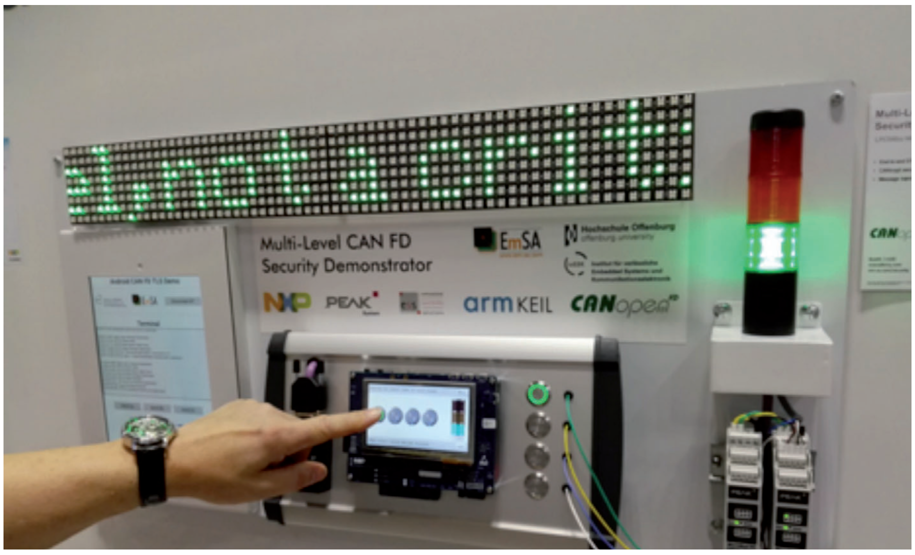

Figure 7: The CANopen multi-level security demonstrator by NXP (Source: Emsa)

depend a lot on the buffering scheme used for keys, cipher blocks and CAN frames. An additional 1000 to 2000 bytes can be expected.

Computational resources

An ARM Cortex-M0 at 48 Mhz can execute a full Speck block cipher in less than 30 *μ*s, a full XTEA cipher in less than 40 *μ*s. A full block cipher (all rounds) is executed twice for initial grouping, then once per second or twice per second on failure or recovery. Each message transmitted or received requires the digital signature generation. Using Speck and one quarter of recommended rounds for the one-time pad and one quarter for the checksum encryption, the CPU time required on the ARM Coretex-M0 comes to about 15 *μ*s. This is already close to the 12,5 *μ*s desired in the estimation above.

CAN and security

- Olaf Pfeiffer (EmSA): CAN security with hidden key generation (CAN Newsletter magazine 2/2016)
- Olaf Pfeiffer (EmSA): Scalable CAN security (CAN Newsletter magazine 2/2017)
- Olaf Pfeiffer, Christian Keydel (EmSA): Security expectations vs. limitations, part 1 (CAN Newsletter magazine 1/2018)
- Olaf Pfeiffer, Christian Keydel (EmSA): Security expectations vs. limitations, part 2 (CAN Newsletter magazine 2/2018)
- Olaf Pfeiffer, Christian Keydel (EmSA): Self-confi guring CANopen controller (CAN Newsletter magazine 2/2018)
- Olaf Pfeiffer, Christian Keydel (EmSA): No excuses for not securing your CAN FD communication (CAN Newsletter magazine 3/2018)
- Olaf Pfeiffer, Christian Keydel (EmSA): CANopen FD multi-level security demonstrator (CAN Newsletter magazine 1/2019)

CAN receive filtering

It is important to do any CAN receive filtering before authentication. The 29-bit CAN identifier still contains the original 11-bit one from its unsecured frame counterpart, so current receive filters need to be adapted accordingly. An authentication cycle should start not before a filter is set to receive this.

Outlook

In the next issue of the CAN Newsletter magazine, you can expect to read about first real-system integrations with Micro CANcrypt. At that point we will be able to give you even more specifics on memory sizes and CPU performance required. In addition, we will review how well this is suited not only to classical CAN but also to CAN FD communications and review possible attack vectors. ◀

Authors

Olaf Pfeiffer, Christian Keydel
Emsa (Embedded Systems Academy)
info@esacademy.com
www.esacademy.de

Products for mobile automation

Maximum reliability for extreme conditions

If there is one thing we know after many years of experience with sensors and control systems:
Products used in mobile machines must be extremely robust. Exposed to heat, cold, moisture, dust and vibrations, they must guarantee maximum reliability – even if the going gets tough. This is why we offer corresponding solutions for operation, communication and remote maintenance. The result: increased uptime and maximum reliability of your machines. ifm – close to you!

WARRANTY 5 years on ifm products

Go ifmonline
ifm.com/gb/mobile