**Challenge CAN trace**

In CAN, normal data frame messages consists of 8 bits.

Error messages consist of 2 and 4 bits, so I went over 2 and 4 bit messages to try and spot one that had suddenly appeared in the middle of the log.

Additionally, all CAN commands seem to constantly repeat themselves, so finding an anomalous command not repeating, or stopping repeating, or starting, might be something of interest.

Constantly repeating commands:

2 41 00 to 2 30 00 descending slowly

0d 0f 07 00 repeats constantly

87 0b 07 00 repeats constantly

32 0a 07 00 repeats constantly

bc 03 07 00 repeats constantly

seeing 07 00 ending as usual, I moved to searching other endings.

**2 3f 00 – suddenly stops at 5.463906 seconds into the log. Beforehand reappears each 0.20 seconds.**

It means something happened **between 5.463906 and 5.663906** that stopped the 2 3f 00 message from appearing until the end of the log.

8 00 00 00 00 00 00 00 00 – appears constantly

4 fd 0c 07 00 – appears constantly slowly descending

1 0 8 8 41 00 38 00 00 01 00 00 – appears constantly

The 3f 00 gets replaced with 3e 00, and then 3d 00, similarly to 2 41 00 that goes down to 2 30 00. **Which means it’s legitimate CAN traffic.**

Going over the commands again, I noticed a single oddly looking command **8 b0 07 00 00 00 FF FF FF.** the command **8 b0 07 00 00 00 00 00 00** repeats all over the file, **each 0.50 seconds.** However, after the command at 37.59, in the following command at 38.09 the last 3 hex bytes turned to FF FF FF, which might indicate a **buffer overflow** somewhere between **37.598429 and 38.098323** so I started looking for new or changed commands in that period.

The ID of the command in question was relatively small, so it hints it may be something important.

msg5 and id481, but it all repeated the same 8 b0 07 00 00 00 00 00 00.

I checked if the written number of bytes matches the actual number of bytes, and it all seems to match.

I was unable to spot the cause of the overflow in this timeframe, so the best I know is that something happened, presumably in this timeframe, that caused the bytes to overflow (the rest of the message is completely identical to the numerous repeats of it all over the file each 0.50 seconds).

I started trying to find other anomalies in the data, going over each msg[n] number, and looking for changes. Statistics:

463 **Msg1**  1 0 8 8 00 00 00 00 00 00 00 00 constant

12dd5471x **Msg2** 1 0 8 8 00 00 00 00 00 00 fe fe constant

5f2 **Msg3** 1 0 2 2 41 00 overall, goes to 2 30 00 over the file. **Not lineary.**

Sometimes jumps between adjacent values, for example **16.46 value is 2 32 00, 16.55 value is 2 31 00, and 16.60 value is 2 32 00 again**. Could be a clue for malicious tampering, could be something legitimate like speed or RPM.

464 **Msg4**  1 0 8 8 00 00 00 00 00 00 00 00 constant

481 **Msg5**  1 0 8 8 b0 07 00 00 00 00 00 00 constant **aside last time with FF’s**

3c0 **Msg6** 1 0 4 4 xx xx 07 00 x’s constantly changing across all file **aside very brief moment between 18.300254 and 18.330254** when the command is 1 0 4 4 fd 0c **09** 00 and it comes **4 times in a row**. Definitely an anomaly and a strong clue that **something happened between 18.300254 and 18.330254**

462 **Msg7** 1 0 8 8 41 00 38 00 00 01 00 00 constant

465 **Msg8** 1 0 8 8 00 00 00 00 00 00 00 00 constant

16a954b1x **Msg9** 1 0 8 8 00 00 00 00 00 00 00 00 constant

16a954bax **Msg10**  1 0 8 8 00 00 00 00 00 00 00 00 constant

33e **Msg11** 1 0 8 8 00 00 00 00 00 00 00 20 constant

650 **Msg12**  1 0 8 8 00 00 00 00 00 00 00 00 constant

000 **Msg13** 1 0 8 8 00 00 00 00 00 7c fd 0c **Appears once at 31.148090.** has very strange ID of **000.** It’s either some extremely important message that comes seldomly, or it is another clue for tampering and possible attack.

**Summing up, based on the statistics and patterns of the trace mentioned above, I can make the following statements regarding anomalies in the CAN Trace:**

1. **Msg5** which takes the value of b0 07 00 00 00 00 00 00 the whole file at **precise intervals of 0.50 seconds**, suddenly changes to b0 07 00 00 00 **FF FF FF** at its last recording at **38.098 seconds**, closely to the end of the trace.

This might point at a buffer overflow attack/bug that caused the last bytes of the msg to overflow to their max possible value.

This might also point at the work of an encryption/destruction malware running in the background, and writing junk data to actual memory locations.

2. **Msg3** goes gradually from 2 41 00 to 2 30 00 over the course of the file, but does so **not in a linear matter**. This could be a perfectly normal behavior as well as sign of tampering or attack.

3. **Msg6** is of constant form xx xx **07** 00, when X’s are constantly changing. However, for a very brief period of time, **between 18.300254 and 18.330254**, it takes the form of fd 0c **09** 00, and not only the byte that is constant in any other place changed, but it did so **for 4 times in a row**, **filling the log only with this Msg6**. This is highly irregular and anomalous behavior and if not a sign of an attack, it’s clearly a bug, probably in the asynchronous part of the programming when a thread takes shared resources and doesn’t release. (Can also point on an attack from that vector).

4. **Msg13** a strange message, **appearing only once at 31.148090** and under an even stranger **ID of 000**. Not any other message has an ID even close to 000, or appearing only once. Therefore, combining those factors, this is an irregularity for the Trace, and either some very important and seldom message, or a sign of attack/bug that happened and it shouldn’t be there/shouldn’t have 000 id.

All those 4 events are suspicious, especially Msg5 and Msg6 cases, and might strongly indicate that an attack or bug took place, thus they require further investigation by professionals.

Daniel Levin.