The CAN Injection attack

The attack has been reported widely in the media. This article focuses on the special properties of the CAN Injector and explains them for "CAN insiders". Methods for defeating the attack are discussed as well.

A few weeks ago, I published on my blog a detective story [1]. It describes in detail how Ian Tabor, an automotive cybersecurity researcher, had his Toyota RAV4 stolen. It was clearly a sophisticated crime: the thieves managed to override the engine immobilizer without using the keys and drive the vehicle away. A few weeks earlier they had tried to steal the car and failed. Ian tweeted pictures at the time of the damage they had caused.

Figure 1: A tweet from Ian Tabor a few weeks before the car was stolen, showing how the 'vandalism' was actually an attempted CAN Injection attack (Source: Ian Tabor, Twitter)

Figure 2: CAN Injector hidden inside a JBL Bluetooth speaker case (the device is powered by the speaker's battery and hidden in resin) (Source: Ian Tabor)

After the car was stolen, Ian used the Toyota 'MyT' telematics service to examine vehicle diagnostics remotely and very quickly focused on diagnostic trouble codes (DTCs) related to the CAN network. He suspected the thieves had accessed the car's CAN network to override the immobilizer and open the doors. After some research on the dark web, he found that devices were being sold to thieves to inject CAN frames for specific brands and models of cars. He bought one of these devices for Toyota and Lexus cars – hidden inside a JBL Bluetooth speaker case (Figure 2) – and asked me to help reverse engineer the device.

The device has CAN_H and CAN_L wires that are attached to a vehicle's CAN network and then CAN frames are injected on to the network. To steal a RAV4, thieves remove a panel and access CAN_H and CAN_L lines in the headlight connector – just as the damage to Ian's car showed (from the earlier failed attempt to steal the car). After some investigation we named this the CAN Injection attack and notified the Automotive Security Research Group (ASRG). It now has an official common vulnerabilities and exposures (CVE) identifier: CVE-2023-29389. This vulnerability applies not just to Toyota or Lexus models: the dark web sites selling these theft devices list many models of cars from many manufacturers.

The core of the CAN Injection attack is Classical CAN frame spoofing, exploiting the way the vehicle is architected according to the perimeter defense concept (i.e. only the outer perimeter of a system is protected on the assumption that nothing can get to the unprotected part). Figure 3 is a simplified schematic of the RAV4's CAN networks.

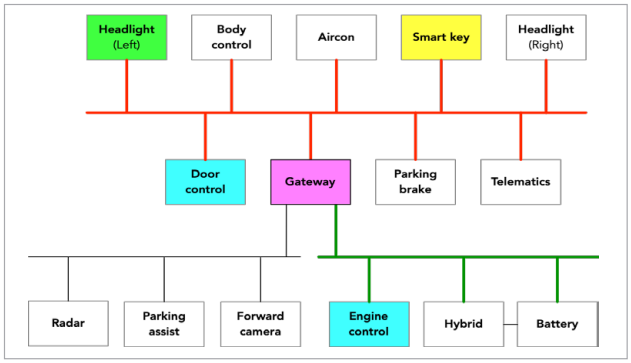
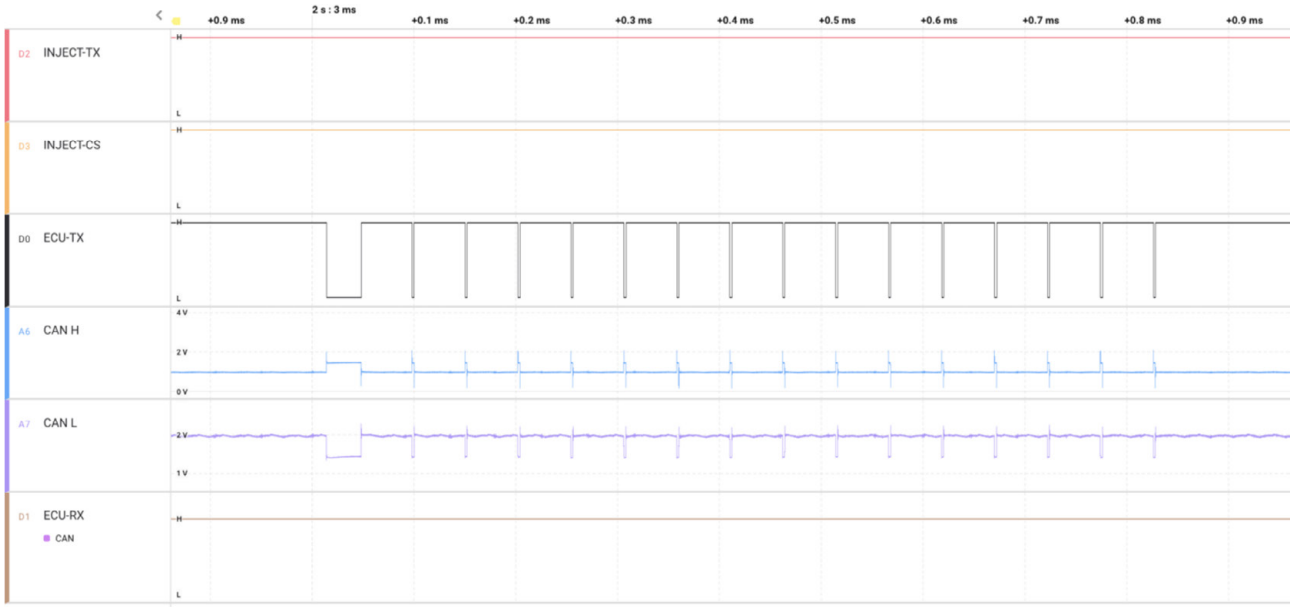


Figure 3: A simplified schematic of the Toyota RAV4 CAN networks (Source: Canis Automotive Labs)

Three CAN networks are shown. The network marked in red connects multiple electronic control units (ECUs) together (there are many others on these networks that are not shown). The thieves broke into the connector near the left headlight ECU. The injected CAN frames spoof the frames that normally come from the smart key ECUs. This ECU has very sophisticated cryptographic messaging over a wireless link to the owner's key, but the CAN messaging from the smart key ECU to the engine ECU (via a gateway) and the door ECU is unprotected.

CAN details

The attack has been reported widely in the media but very few reports focus on the most important property ▷



INJECT-TX	**The line representing CAN TX of the CAN Injector device**
INJECT-CS	**The circuit select for the dominant-override**
ECU-TX	**CAN TX of the CAN controller in the ECU**
ECU-RX	**CAN RX of the CAN controller in the ECU**
CAN H	**CAN-High line**
CAN L	**CAN-Low line**

Figure 4: A logic analyzer trace showing an ECU trying to send a CAN frame when the dominant-state-override function is triggered (Source: Canis Automotive Labs)

of the CAN Injector: *it contains a modified CAN transceiver*. When enabled, this transceiver can actively drive the recessive state on the CAN network, overriding other controllers that try to assert a dominant state. This means that other ECUs cannot transmit frames, leaving the CAN Injector as the only transmitter. Figure 4 shows a logic analyzer trace of a CAN controller trying to send a CAN frame.

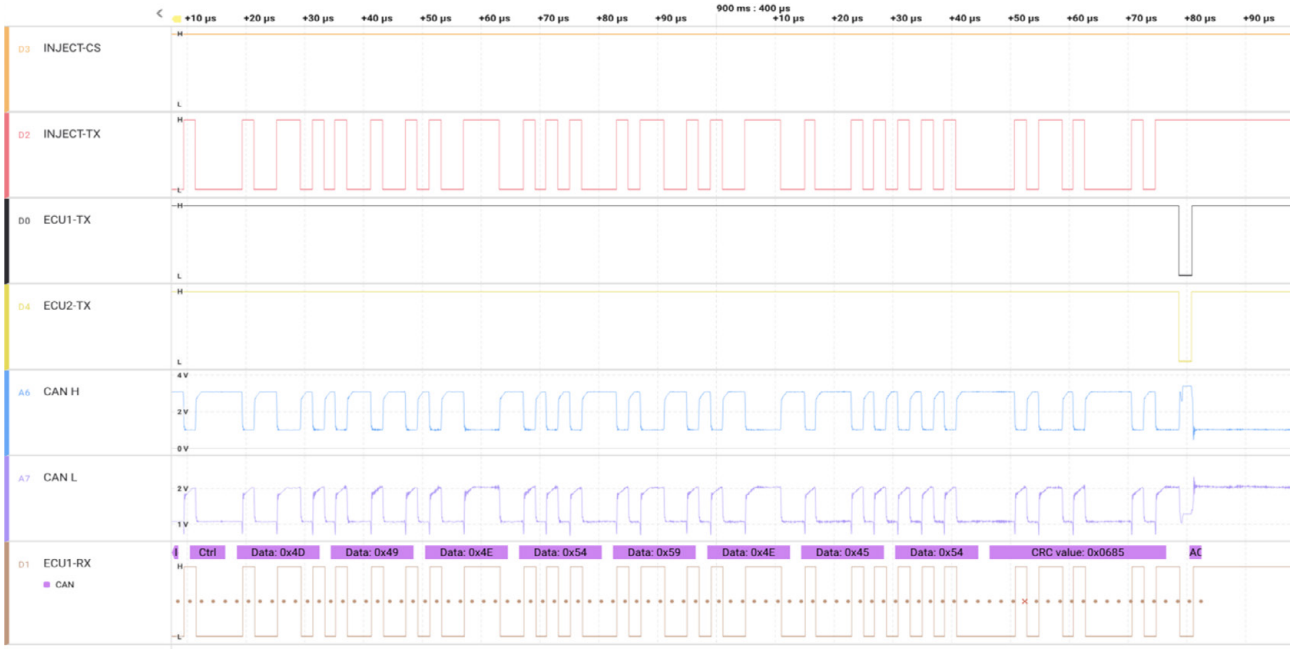


Figure 5: Multiple CAN controllers asserting a dominant state when the CAN Injector dominant-override transceiver circuit is engaged (Source: Canis Automotive Labs)

Figure 4 shows how an ECU CAN controller attempts to assert a dominant state on the CAN network. But the CAN Injector device holds the voltages below the thresholds needed for any CAN transceiver to recognize a dominant state. The transmitting CAN controller goes through the error rules of the CAN protocol (since sending a 0 but reading a 1 is a CAN bit error) and then goes "Bus Off". The specific pattern seen for CAN TX from the ECU CAN controller is explained in the answer to a recent CAN Quiz Question [6].

The specific CAN Injector uses a Microchip PIC18F with an on-chip CAN controller. To transmit a CAN frame correctly, the acknowledge field must be read back as 0: receivers must assert a dominant bit in this field. If a CAN controller cannot assert a dominant state, then this would cause the spoof frames to fail to be received. However, the transceiver circuit in the CAN injector is designed so that when *multiple* CAN controllers assert a dominant state *at the same time*, then the combination does force a dominant state and the receivers all accept the spoof CAN frame. Figure 5 shows this happening.

This is important because of how it affects state-of-the-art CAN security hardware that is used to detect and destroy spoof CAN frames. For example, CAN-HG augmentation [2] can automatically identify spoof frames (by using out-of-band data added to Classical CAN frames, containing physical address information) and destroy them by asserting a CAN error (i.e. sending six dominant bits). The 'Stinger' secure CAN transceiver from NXP [3] contains Block/Pass lists of CAN-IDs and any frames with CAN-IDs on the Pass list that are sent from elsewhere on the network are deemed spoof frames and are destroyed. But dominant override transceivers can neutralize anti-spoofing hardware.

Defeating the CAN Injection attack

There are in practice only two ways to defeat a CAN Injection device with a dominant override transceiver: (1) Partition a CAN network into trusted and untrusted segments with a security gateway [4] between, or (2) use cryptographic protection for CAN frames. The problem with the security gateway approach is that it relies on there being no physical access to the trusted network. This leaves just cryptographic protection to defeat CAN Injection. The Autosar SecOC framework for Classical CAN uses four payload bytes to contain authentication information and four bytes of application payload. The CryptoCAN [5] scheme from Canis Labs uses a pair of CAN frames to carry an encrypted and authenticated version of the original CAN frame. Both schemes rely on the cryptographic primitives provided by the Secure Hardware Extensions (SHE) Hardware Security Module (HSM) standard. It is possible to emulate an SHE HSM in software for micro-controllers without HSM hardware – and this provides a way to defeat CAN Injection for existing vehicles by updating ECU firmware.

There is an adage that says "Cryptography is a machine for turning any problem into a key management problem" and this is certainly true for defeating the CAN Injection attack: there must be tools and infrastructure for the secure creation, distribution, re-programming, and storage of keys. Fortunately, the SHE HSM standard ▷

References

[1] https://kentindell.github.io/2023/04/03/can-injection/
[2] https://canislabs.com/canhg/
[3] https://www.nxp.com/products/interfaces/can-transceivers/secure-can-transceivers:SECURE-CAN
[4] https://can-newsletter.org/hardware/gateways/230308_implementation-requirements-for-secured-gateways_canis-labs_cnlm/
[5] https://canislabs.com/cryptocan/
[6] https://kentindell.github.io/2023/03/29/can-quiz-2-answer/

Inferring the sender of a CAN frame

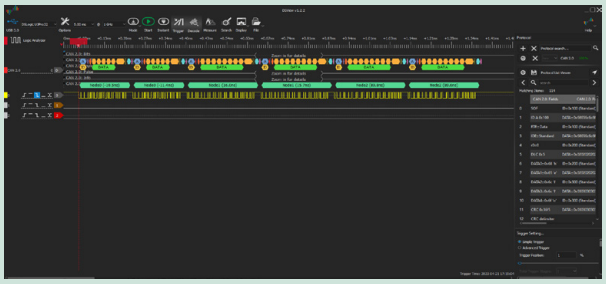


Figure: The decoder uses tiny variations in timing of CAN recessive pulses to automatically infer which CAN frames come from which nodes on the CAN (Source: Canis Automotive Labs)

The latest update of the open-source can2 protocol decoder by Canis Automotive Labs is able to automatically infer the sender of a CAN frame. It uses the method of deterministic distortion of CAN signals that result in frames from a given node on the network having consistently shortened or lengthened recessive pulses. The differences can be quite small - just 10 ns or 15 ns - but they can be picked up by a suitably accurate logic analyzer.

The decoder shows more information about what's happening on the CAN than the usual protocol decoders in logic analyzers. It already warns about unusual CAN events (such as error frames, overload frames, or a double-receive), which might be low-level CAN protocol attacks. Upgrading it to automatically infer the sending node for each frame means that the decoder can passively analyze a CAN network: there is no need to unplug nodes to see which frames no longer appear (which anyway disrupts the behavior of a running system). As it maps CAN-IDs to nodes, it can help to build up a detailed picture of a CAN system. This is useful for debugging (e.g. to see which node sends an unexpected CAN frame), for reverse engineering an unknown system, and even for detecting spoof frames. A spoof frame is one with a CAN-ID normally sent from another node, and is a common technique for hacking the CAN network. The CAN Injection attack used to steal cars is an example of a spoofing attack. Read here {1} how to use the protocol decoder.

{1} https://kentindell.github.io/2023/04/21/can2-decoder-update/#fn:4

defines not only cryptographic operations but also a secure key distribution protocol, and this at least allows standardized tools and processes to be used to address the problem.

Author

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