Karamba Security

A New Model for
In-Vehicle Network Security
with Zero Performance Delays

Autonomous Security

US: +1 734 794 4745
2723 South State Street
Ann Arbor, Michigan, 48104

Tel: +972 9 88 66 113
29 HaCharash St.
Hod Hasharon, Israel

The Conundrum of Connected Cars

Today's vehicles rely on multiple interconnected networks of Electronic Control Units (ECUs) that govern almost every function—from engine timing and traction control to side mirror adjustment and GPS. As you may know, ECUs send messages through a serial data bus, typically a Controller Area Network (CAN). CAN bus networks enable sensors, circuits and motors to share data instantly. Real-time data flow is vital for optimal driving performance and safety. Unfortunately, however, the industry's focus on safety falls short when it comes to cybersecurity.

Networks Left Exposed

The CAN bus is designed with very limited, if any, protections. Adding to the risks, cars are now equipped with Wi-Fi, Bluetooth and wireless sensors, which open many doors to the outside world. Outbound connectivity, over-the-air (OTA) updates and V2X communications expand the attack surface. Even dongles, used by insurance companies and fleet managers to track driver behavior, create new vulnerabilities. External connection points allow hackers to infiltrate ECUs and access bus networks.

White hat hackers have thoroughly demonstrated this with the renowned hacks on Chevrolet Impala, Toyota Tundra, Jeep Cherokee and Tesla Models S and X.[1,2] In every case, hackers were able to exploit a vulnerable network to take remote control of the car.

When hijacking Jeep Cherokee, researchers used a zero-day exploit in Sprint's cellular connection to gain access to the car's infotainment ECU. Once inside, they discovered a D-Bus that did not have authentication enabled.[3] This allowed them to gain root privileges and access a CAN bus, giving them the controls to steer, brake and accelerate any Jeep Cherokee within range of the Sprint network.

Securing Networks on the Move

Given the weakness of in-vehicle networks and ECU vulnerabilities, a multi-layered cyber defense is imperative. In this whitepaper, we focus on securing the network layer with encryption, authentication and tampering protections. Together these defenses can validate the source and data integrity of every message.

The challenge? In-vehicle network security must address the following issues:

1. In-car networks are already overloaded, so security cannot add traffic overhead.
2. Cryptography requires heavy computation, and yet ECUs have limited CPU power and memory.
3. Hackers can connect fraudulent ECUs to the system by using leaked IDs for legitimate ECUs.
4. Attackers can eavesdrop on communications, including the exchange of cryptographic keys.
5. Automaker profit margins are razor thin, so cybersecurity must be cost effective.

In this whitepaper, we explain how a team of determined engineers redesigned cybersecurity to overcome each of these barriers. You'll discover an innovative solution that can immediately improve security on the road. It is now possible to fully deliver the dream of the autonomous and connected car.

Reinventing Network Security

Existing CAN bus encryption solutions fail to address the challenges of securing resource-constrained systems. Carmakers have been asking for better security that can shield bus vulnerabilities without placing a burden on saturated car networks.

To meet these needs, engineers have developed new methods for exchanging encryption keys and authenticating messages so there are no added payloads. The result is Karamba Security SafeCAN, an ultralight yet powerful in-vehicle network security solution that encrypts and authenticates ECU communications—with zero network overhead.

This is a giant leap over existing CAN bus security solutions that can slow performance and often limit security deployment to the vehicle's critical safety systems. By contrast, SafeCAN secures any type and any number of in-vehicle networks. And to minimize cost and effort, it is backward compatible and requires no changes to the application code.

In this whitepaper, we will detail how SafeCAN is designed to secure all communications between ECUs. We will also explain how it defeats threats like ECU impersonation, replay attacks, bit flipping and message tampering.

Exchanging Keys Safely

The first two challenges are: 1) to exchange cryptographic keys without exposing them; and 2) to swap keys without adding network traffic. To accomplish both, crypto keys are deployed during the bring-up phase in the vehicle assembly line, a safe, isolated environment. This means no keys are exchanged while the car is running—when communications are vulnerable to eavesdropping.

Not only does this enhance security, but it improves performance. SafeCAN adds no overhead that could induce latency or impact vehicle safety. Other solutions like vatiCAN, CANCrypt, AUTOSAR secOC, and Trillium SecureCAR exchange keys several times per second during vehicle operation, adding traffic overhead that can cause message collisions and delays.[4]

Keeping Keys Secure

It is equally important to keep the encryption keys safe when stored or in use. Every time the car is started, SafeCAN uses a master key to regenerate keys that were first made at the factory. This all happens in the protected, sealed environment of a hardware security module (HSM). By storing a master key and regenerating all other keys in the trusted world of an HSM ensures keys remain secure.

By securing the key exchange and storage, SafeCAN presents a significant breakthrough. In the following sections, we will dive into greater detail to explain how it generates and swaps keys, then encrypts, decrypts and authenticates ECU communications. We will also explain how our tampering protections combat real-world threats—keeping the car secure.

Phase One: The Bring Up

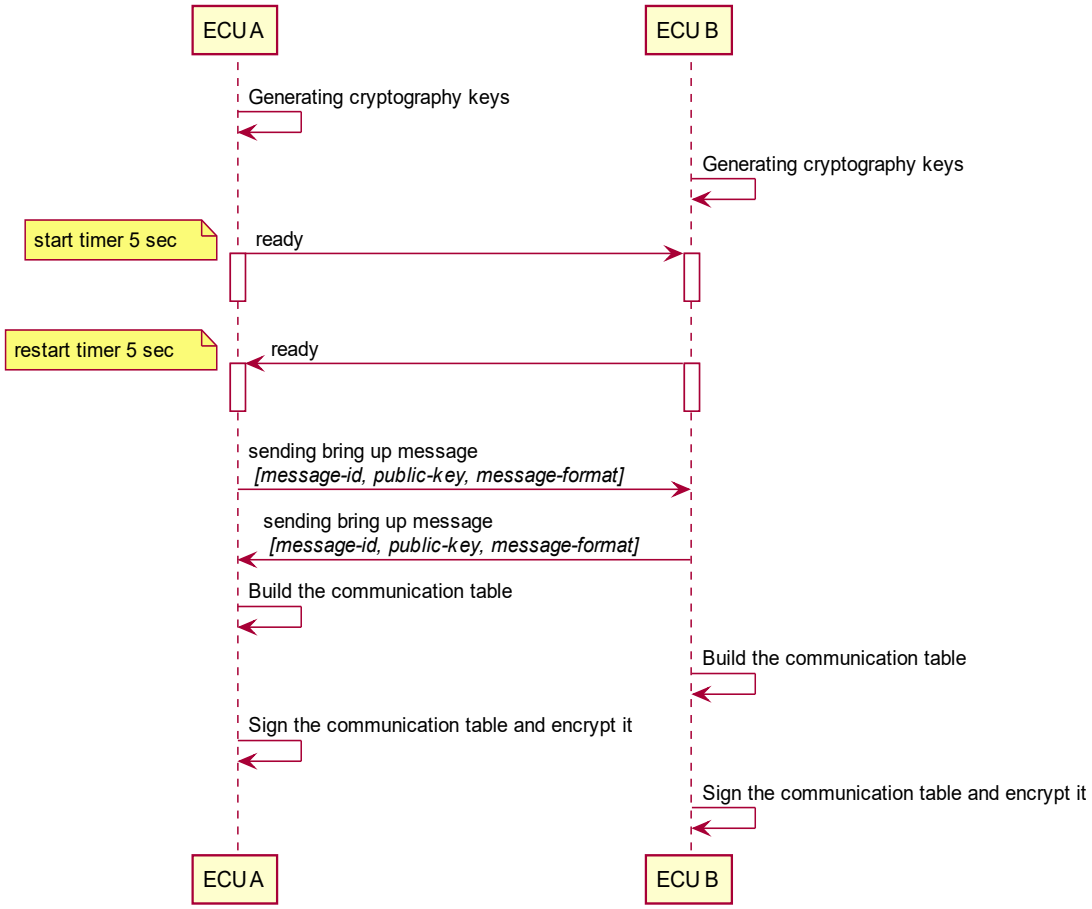
In the manufacturing bring-up phase, SafeCAN is tightly integrated, configured and customized. At this stage, there is no external connectivity, like Wi-Fi and Bluetooth, for hackers to infiltrate and eavesdrop. The factory provides is a safe environment where SafeCAN can generate and exchange keys freely—without any risk of exposure.

To generate keys, SafeCAN extracts each ECU's unique identifier (ID) to use as a master key. SafeCAN then uses the ECU's master key to build four types of cryptographic keys:

1. A public key (K_{pub}) that is openly shared, then stored on the receiving end in an encrypted communication table
2. A private key (K_{pri}) that is *not* stored but is regenerated each time the ECU is started
3. A symmetric key (K_{sym}) to encrypt and decrypt the communication table
4. A MAC secret (K_{mac}) to authenticate the communication table

After generating four keys for every ECU secured by SafeCAN, the ECUs negotiate to share information about the messages they will be sending and receiving. In this process (shown below), ECU A and ECU B send each other bring-up messages that include:

1. Message IDs – a unique identifier for each message
2. Public keys (K_{pub}) – later paired with matching private keys to secure communications
3. Message formats – indicating the bits in the message that will be used when sending data



Building Communication Tables

After ECUs collect the message data, SafeCAN uses it to build communication tables. These tables are used later as a set of instructions to encrypt, decrypt and authenticate each individual message. Every ECU has its own communication table, and each line contains a message that the ECU will either send or receive.

Understanding the Table

If an ECU attempting to negotiate does not receive a response from other ECUs, the messages on that channel do not need to be encrypted. If there is more than one response per message, it means multiple ECUs receive this message.

Flexible Security Policy

As an alternative, when there are two or more receiving ECUs, the system manager may decide not to encrypt the message based on its customized security policy. This customization gives developers the flexibility to adjust for their unique system configuration. For example, SafeCAN may be deployed on two of the receiving ECUs but not the third. The system manager can override any encrypting decisions through the SafeCAN management portal.

Signing and Encrypting the Table

After a communication table is built for every secured ECU, each table is signed and encrypted, using the ECU's MAC secret and symmetric key. These same two keys are used later in the operating phase to decrypt and authenticate the communication table, the most efficient, accepted standard for single-party encryption.

Encrypting the communication table prevents an attacker from downloading the image, reading it, or altering the data. By the end of the bring-up phase, all communication between the ECUs is secure. And because each step took place in an isolated environment with no external connections, there is no chance the encryption keys were exposed.

Phase 2: Vehicle Operation

This phase begins every time the car is started. When the car is running, and an ECU is first activated, SafeCAN repeats several steps in the bring-up phase to regenerate three of the encryption keys and perform a string of rapid functions:

1. A symmetric key (K_{sym}) is used to decrypt the communication table
2. A MAC secret (K_{mac}) is used to authenticate the communication table
3. A private key (K_{pri}) is paired with a matching public key stored in the communication table

Once these keys are regenerated, the ECU uses the symmetric key to decrypt its communication table, unlocking the public keys and message data. SafeCAN then authenticates the communication table, using a hash-based message authentication code (HMAC-SHA256), which requires two passes of cryptographic hash computation and the MAC secret. By authenticating the sender, HMAC verifies data integrity—ensuring the communication table has not been altered or sent by a hacker.

High-Performance Block Cipher

For encryption and decryption, SafeCAN is using a block cipher with variable cryptography methods, depending on the message data length (shown below). The preferred cipher for eight bytes of data is SPECK 64/128 with 27 rounds. This is the heaviest of our computation tasks, and yet the cipher performance is 170 cycles per byte (cpb), requiring only 200 bytes of ROM and 120 bytes of RAM. On a 100MHz CPU, encryption requiring 1000 cycles would take only 10 microseconds (µs).

In comparison, AES128 encryption, runs 765 cpb, consuming 4500 bytes of ROM (over 20 times as much), and 1800 bytes of RAM (15 times as much). This would introduce a level of latency that would most certainly impact vehicle performance.

Length-Dependent Cypher Alternatives

Data Length	Cipher	PRF	Cycles
1 Byte	Lookup table	N/A	~2 cycles
2 Bytes	Feistel	Lookup table	~10 cycles
3 Bytes	Feistel	Lookup table	~20 cycles
4 Bytes	SPECK32/64	N/A	~340 cycles
5 Bytes	Feistel	SPECK32/64	~1020 cycles
6 Bytes	SPECK 48/96	N/A	~690 cycles
7 Bytes	Feistel	SPECK32/64	~1020 cycles
8 Bytes	SPECK64/128	N/A	~800 cycles

As you can see in the lookup table, the Feistel cipher is used when the data length is 2, 3, 5 or 7 bytes. The symmetric Feistel structure has the advantage of encryption and decryption operations that are very similar, requiring only a reversal of the key schedule. As a result, the size of the code and the resources required are nearly cut in half.

Hack-Proof Communication

In addition to encryption and authentication, SafeCAN fortifies security with multiple layers of tampering protections. To avoid adding traffic overhead, the developers created trip wires using the redundant bits, a pseudo-random counter and the hamming distance. These protections do not change the meaning nor size of the message—so receiving ECUs can validate data integrity. This means it is accomplished without adding data to the message payload. Other CAN bus security solutions authenticate data by adding a counter, MAC and extra messages, which can double the payload and slow performance.

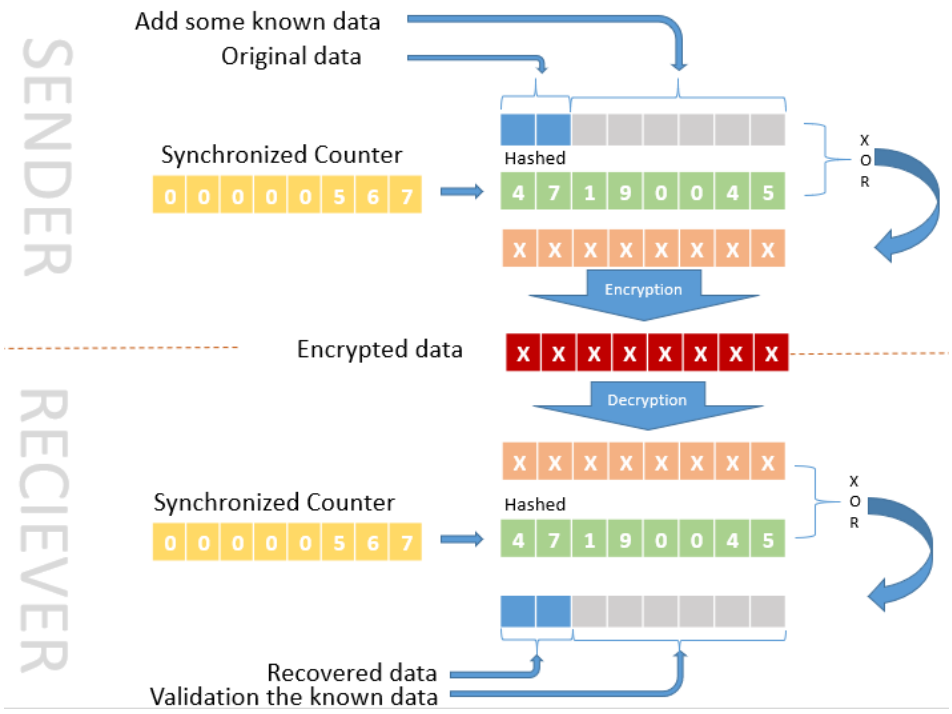
Verifying the Redundant Bits

The first layer of defense authenticates messages by using their redundant bits, the extra unused binary digits moved along with the significant bits in a data transfer. As illustrated below, the maximum payload of a CAN message is eight bytes, but the length of the actual message might be shorter (shown in blue). Knowing the expected value of the redundant bits allows SafeCAN to authenticate a message after the decryption, if all the redundant bits are the same. If any of the redundant bits are altered, we know someone has tampered with it, and the message is denied.

Using a Pseudo-Random Counter

The second layer of protection does even more to prevent message tampering, message injection and replay attacks. SafeCAN uses a pseudo-random counter, similar to a time stamp, but all of the significant and redundant bits are scrambled to make it unpredictable. SafeCAN does this by running a counter through a pseudo-random function. It then applies an “exclusive or” (XOR) operation with the original data. The output (shown in orange) is then encrypted.

Hacking Detections: Redundant Bits and Pseudo-Random Counter



After the receiving ECU decrypts the message, it reverses the pseudo-random counter and checks the redundant bits. If the redundant bits didn't change and the hamming distance between messages is in range, the message is authenticated. If any redundant bits changed, SafeCAN blocks the message. For example, if a hacker records a session and attempts a replay attack, the pseudo random counter, which validates freshness, would not be correct, plus the redundant bits and hamming distance would change. All three factors sound the alarm, indicating an attempted attack.

Blocking Real Threats

The types of threats that target in-vehicle networks are similar to other network attacks. There are a few exceptions, however, that are unique to vehicle networks derived from CAN Bus specifications. The bus broadcast topology allows any connected device, including a fraudulent or compromised ECU, to send malicious control messages. Without the right security, the receiving ECUs have no way of verifying the authenticity of the sender or the data.

In this section, we describe various attack methods that hackers use to gain remote control of vehicles. More importantly, you will learn how SafeCAN combats these threats to prevent attacks.

Replay Attacks

To initiate a replay attack, hackers eavesdrop on the bus communications, record a session and replay it later. Since the message is exactly the same, the replayed data easily tricks the receiving ECU into false authentication or a duplicate transmission. Even if the messages are encrypted, and the attacker does not know what it says, the retransmission of a message is sufficient for the hacker to gain access to the in-vehicle network. This means encryption and HMAC authentication are not enough.

SafeCAN is able to prevent replay attacks because the pseudo-random counter ensures freshness and scrambles the message bits so a hacker cannot generate a valid counter. Any attempt to replay a message using an old counter value will result in redundant bits that are not zeroed. SafeCAN will then invalidate and reject the replayed message.

Message Tampering

Before launching a message tampering attack, also known as message spoofing, an attacker listens to the vehicle's transmitted data and analyzes it to collect useful information. The hacker can then modify messages or inject fake messages using valid message IDs. The hacker's goal is to fool an ECU into processing false data, causing it to perform unauthorized, possibly dangerous operations. Even if the data is encrypted, an attacker can make the communication appear legitimate.

SafeCAN blocks tampered and spoofed messages by testing the redundant bits and the hamming distance between messages that are using the same message ID. If someone has spoofed or modified a message, the redundant bits will not be zeroed and the hamming distance will be greater than epsilon. SafeCAN prevents the ECU from executing the message.

Bit Flipping

A bit flipping attack is specific to vehicles due to a vulnerability of the CAN bus network. An attacker can simply alter the serial data by pulling the serial line down during transmission. This method enables the hacker to change the ciphertext in a way that results in a predictable change of the plaintext, even though the attacker is not able to see the plaintext message.

SafeCAN makes bit flipping communication errors impossible because the block cipher is not linear. As described earlier, the block cipher is variable and involves a lookup table and a pseudo-random function. This complexity prevents any bit flipping attempt to pull the serial line down without raising a red flag. By checking the redundant bits and the hamming distance between the messages, SafeCAN would detect changes to the messages and block the attack.

ECU Impersonation

As mentioned in the introduction, ECU impersonation is one of the four primary challenges that security must address. Without preventative measures, hackers can connect a fraudulent ECU to the system by impersonating a legitimate ECU. This threat exploits a weakness specific to the CAN bus architecture.

Attackers can program malicious messages in a fake ECU and trick the system, simply by using the ID of a real ECU. Attackers can also impersonate an OTA update and deploy malicious software on ECUs. In a network that does not have sender authentication, the system will execute unauthorized messages or malware, putting the hacker in control of car functions.

SafeCAN prevents hackers from impersonating ECUs by authenticating the sender of every message. And without access to the cryptography engine, the fake ECU cannot send encrypted messages, so any message it transmits will be blocked.

Comparing Solutions

Other solutions like vatiCAN, CANCrypt, Trillium SecureCAR and AUTOSAR secOC only secure CAN bus communications. They also add traffic overhead, which means these solutions are typically deployed on a limited number of critical safety systems. In many cases, the induced latency for authenticated messages is not fast enough for timing-critical functions, like braking and steering.

SafeCAN, on the other hand, secures any type and any number of in-vehicle networks without a performance penalty. SafeCAN is also backward compatible, offering cost savings and flexibility advantages over high-end NXP solutions that require all ECUs to use the same specialized transceivers.

Performance Impact

In every way possible, SafeCAN is designed to minimize the burden on the car's resource-constrained systems. Even the closest competitor, Trillium SecureCAR has the potential to slow performance. SecureCAR, vatiCAN, CANCrypt, and AUTOSAR secOC all exchange encryption keys several times a second, while the car is running. This adds network overhead and increases the risk of exposing keys.

SafeCAN, on the other hand, only exchanges keys at the factory, a safe, isolated environment. And by eliminating the need to exchange keys during vehicle operation, SafeCAN adds zero network overhead—a significant advantage for securing in-vehicle networks where performance is paramount.

Added Message Payloads

Other solutions also slow performance by adding message traffic while the car is operating. SecureCAR, vatiCAN, AUTOSAR secOC, and CANCrypt all implement some degree of message authentication by sending extra messages that contain validation data. These messages are typically the same size as the authenticated message, adding 100% overhead and slowing performance. In contrast, SafeCAN does not add any extra messages or additional payloads.

CPU and Memory Requirements

Another factor to consider, the cryptography algorithms used by some solutions require extra computation power and memory. For example, CANCrypt uses AES128 encryption, which requires 4,500 bytes of ROM and 1,800 bytes of RAM due to the cipher performance of 765 cpb. For this reason,

CANCrypt and other solutions are typically limited to vehicle safety systems—to avoid latency.

In comparison, SafeCAN can be deployed across all ECUs without creating a performance penalty. For the heaviest computation task of encrypting eight bytes of data, SafeCAN uses 170 CPU cycles per byte (cpb), requiring only 200 bytes of ROM and 120 bytes RAM.

Flexible Deployment Options

SafeCAN also works in a mesh network system where security is deployed on some, but not all, ECUs. As explained in the bring-up section, if a message is sent to multiple receiving ECUs, and one or more do not have SafeCAN installed, the system manager decides whether to encrypt the message based on security policy. This is an example of how tier one developers can customize policies to dictate message handling. The system manager can be used to override any encrypting decisions.

Reducing Costs

Cost savings and quality are a delicate balance in the highly competitive auto industry, challenged by low profit margins. For high-end cars, NXP solutions offer advantages with in-vehicle transceivers and microcontrollers that include network security. The integrated NXP security is fast and powerful because it uses an in-transceiver cryptography engine to encrypt and decrypt traffic.[5] However, that engine requires all ECU's to use the same NXP transceivers or microcontrollers across the entire system.

For lack of other options, carmakers have started adding NXP solutions to midrange vehicles too. This practice should be reevaluated. SafeCAN is highly effective and cost efficient—a true game changer that lowers costs without lowering performance standards. Unlike NXP solutions, SafeCAN is hardware agnostic, and the message format is unchanged—so it is backward compatible. Most notably, SafeCAN is more affordable and more flexible than NXP solutions. Carmakers no longer need to overspend to secure networks in midrange cars.

Conclusion

Karamba SafeCAN is the auto industry's first ultra-light network security software to encrypt and authenticate communications between ECUs—without slowing performance. From the key exchange to message authentication, SafeCAN is designed to maximize protection while minimizing the burden on the vehicle's limited-resource systems. This integrated security solution detects and stops threats to ensure ECUs only execute legitimate commands.

Advantages of Karamba SafeCAN

- Authenticates the sender and data integrity of every message—without adding payloads
- Eliminates the risk and overhead of exchanging keys while the car is running
- Blocks malicious messages from fake ECUs and unauthorized senders
- Authenticates cloud-to-vehicle communications to prevent OTA malware downloads
- Minimizes cost and effort, requiring no changes to the application code or chipset
- Secures any type of serial data bus so mixed ECUs can work together in a mesh system
- Delivers real-time protection, enabling sensors, circuits and motors to share data instantly

End-to-End Security

When deployed together, Karamba Carwall and SafeCAN deliver end-to-end protection—from the external connection points where hackers gain entry—to the networks where hackers can take control.

As the industry's first autonomous cybersecurity for vehicles, Carwall hardens the ECU runtime environment based on factory settings. During the build process, Carwall autonomously maps all acceptable call sequences and creates a customized security policy. Using lightweight control flow integrity, application whitelisting, and an ECU network firewall, Carwall blocks any illegitimate function calls and malware downloads—before any damage can occur.

About Karamba Security

Autonomous and connected cars, IoT and smart factories are changing our lives. But with progress comes a growing risk of cyberattacks. Karamba Security delivers award-winning cybersecurity solutions for connected systems. Karamba's software is designed and implemented to safeguard resource-constrained systems.

Product manufacturers in the automotive industry, industry 4.0, IoT, and enterprise edge rely on Karamba's automated runtime integrity software. Using Karamba software, the devices are self-protected against cyberattacks without requiring any development change or security updates. Vendors leverage Karamba Security to increase their product value, while protecting their brand image and customers against cyberattacks.

Learn more online at Karamba Security or contact us.

[1] *Wired*, "Tesla Responds to Chinese Hack with a Major Security Upgrade," Sept. 27, 2016.

[2] *Forbes*, "Hacker Says Attacks On 'Insecure' Progressive Insurance Dongle In 2 Million US Cars Could Spawn Road Carnage," Jan. 15, 2015.

[3] *IEEE Spectrum*, "Jeep Hacking 101," Aug. 6, 2015.

[4] *EE Times*, "Karamba Says It Can Protect CAN," Jan. 13, 2017.

[5] NXP, "C29x: Crypto Coprocessor" accessed Feb. 2018.