2022-01-0126 Published 29 Mar 2022

Mechanism for Runtime Kernel Integrity Check without Additional IP and without TEE for Low/Mid Automotive Segments

Shyju Thekkumbadan, Sreedhar Reddy Pacharla, and Jipin Jose Harman International India Pvt Ltd.

Citation: Thekkumbadan, S., Pacharla, S.R., and Jose, J., "Mechanism for Runtime Kernel Integrity Check without Additional IP and without TEE for Low/Mid Automotive Segments," SAE Technical Paper 2022-01-0126, 2022, doi:10.4271/2022-01-0126.

Received: 24 Jan 2022 Revised: 24 Jan 2022 Accepted: 18 Jan 2022

Abstract

Vehicles have more connectivity options now-a-days and these increasing connection options are giving more chances for an intruder to exploit the system. So, the vehicle manufacturers need to make the ECU in the vehicle more secure. To make the system secure, the embedded system must secure all the assets in the system. Examples of assets are Software, Kernel or Operating system, cryptographic Keys, Passwords, user data, etc. In this, securing the Kernel is extremely important as an intruder can even exploit the operating system characteristics just by changing the kernel code without introducing a trojan in the system. Also, the Kernel is the one entity that manages all permissions, so, if the kernel is hacked, these permissions also get compromised. The proposed approach is to make the kernel secure by doing the integrity check periodically of the kernel code loaded into the main memory of the system. This method uses ARM TrustZone technology which reduces the risk of attacks by hardware partition and separating the critical assets related to security. The Arm TrustZone technology protects the security-critical operations by executing them in a trusted execution environment (TEE). The idea to use the ARM TrustZone for the approach is, Kernel security check needs to be done at high privilege level than the Kernel. The kernel privilege level is Exception Level 1 (EL1), and the monitor code has the highest privilege level that is Exception Level 3 (EL3). This proposal can be used in all mid/low automotive ECUs where neither a dedicated hardware controller nor Trusted execution environment (TEE) is available.

Introduction

The new generation cars are providing many features like Internet connectivity, vehicle-to-vehicle communication, application to vehicle connectivity. These types of cars are also called connected cars. Connected cars can provide the facility of the internet to the IoT devices inside the car and can also transfer the information to external devices. With the help of connectivity, the vehicle OEM can know the behavior of the car and be able to foresee the issues faced by the customer and the issue is being fixed through OTA update. The next-generation cars will have an autonomous driving facility and it depends upon vehicle-to-vehicle connectivity and connectivity with road network facilities. Connected cars also enhance the customer experience who can connect to the outside world and can also use applications which they use on mobile or laptop. Through this connectivity, Customers can install third-party applications which provide entertainment in the car alone. Customers can use the navigation system of the car without opening the mobile and will get advanced safety features that can decrease the chances of an accident. In case of an accident, the car can contact emergency services to prevent death. Emergency call (eCall) has been made mandatory in all cars sold within the EU from April 2018 [1].

The Connected Car is being useful for both OEM and customers, but it is becoming vulnerable and increasing the chances of cyber-attack [2, 3]. Now the attacker can attack the system of the car without being near to the car and control the vehicle system like braking system, electrical system, etc. It is one of the major requirements to safeguard the car and customer data from unauthorized persons to manipulate the vehicle system. So, the OEM of vehicles needs to provide security features to the ECU to avoid or detect attacks by intruders. The basic protection from cyber-attacks is to protect the kernel.

Kernel is an important code of operating system which has control over hardware directly and provides the communication between the software applications and hardware components. The kernel includes the services like interrupt handling to process the requests of hardware or software components, scheduling of process to decide which process need to run on the system. It has access to memory where the customer can store personal information or diagnosis of vehicle system will be present. Therefore, the Kernel exploit will give full access over the system to the attacker which is being depicted in Figure 1. For the Kernel Protection, Developers are reducing the chances of attack by reducing attack surface, strict kernel memory partitions, by making executable code not writable. Still, there is a chance of exploiting the kernel which is running on DRAM. Thus, in such scenarios, it is crucial to check and protect the kernel while the system is running. Traditional Approaches are

[ATF](0)[0.823687]NOTICE: BL31: Built : 15:13:42, Oct 12 2021

[ATF](0)[0.824376]VERBOSE: bl31_platform_setup()

[ATF](0)[0.824917]INFO: SPM_POWERON_CONFIG_SET_0x10006000=0x1

[ATF](0)[0.825626]INFO: BYPASS_SPMC_0x10006c5c=0x0

[ATF](0)[0.826218]INFO: [spmc_init]change to SPMC mode !!!

[ATF](0)[0.826918]WARNING: Using deprecated integer interrupt array in gicv2_driver_data_t

[ATF](0)[0.827891]WARNING: Please migrate to using an interrupt_prop_t array

[ATF](0)[0.828732]INFO: ARM GICv2 driver initialized

[ATF](0)[0.829372]INFO: ptr = 0x4300ead0, len = 2021

[ATF](0)[0.829986]INFO: BL31: Initializing runtime services

[ATF](0)[0.830686]INFO: BL31: Initializing BL32

D/HC:4 init_secondary_helper:957 Secondary CPU Switching to normal world boot

[0.285558] unable to find node gpio_setup

[OK] Started nostromo.service.

[OK] Reached target multi-user.

mtk2712: root

[ATF](8)[181.005866]INFO: runtime calculated hash:351475601c696e795282036000076684a547e317836304a49696c591851

[ATF](8)[181.005869]INFO: Hash matches with stored value. Kernel is intact

[ATF](8)[191.009702]INFO: runtime calculated hash:351475601c696e795282036000076684a547e317836304a49696c591851

[ATF](8)[191.011628]INFO: Hash matches with stored value. Kernel is intact

[ATF](8)[201.014732]INFO: runtime calculated hash:351475601c696e795282036000076684a547e317836304a49696c591851

[ATF](8)[201.017037]INFO: Hash matches with stored value. Kernel is intact

[ATF](8)[211.021853]INFO: runtime calculated hash:351475601c696e795282036000076684a547e317836304a49696c591851

[ATF](8)[211.025852]INFO: Hash matches with stored value. Kernel is intact

In the experimental setup, the Initial hash is being calculated during the process and stored as a reference hash. The reference hash can be precalculated to reduce initial hash calculation operations. Then after that SMC call triggers the hash calculation service in ATF and will be compared with the reference hash. It can be observed from logs that run time calculated hash is same as reference hash. This shows that Kernel is intact during the run time. The hash algorithm used in the experimental setup is Secure Hash Algorithm 2 (SHA2).

The proposed solution does not suggest any crypto algorithm to calculate the hash. The algorithm can be decided by the OEM based on the capabilities of the processor and the strength of security required.

Summary/Conclusions

The connected car is integrated with many electronic devices which increased connectivity, which is useful for both OEM and customers. But the increase in connectivity also increased the chances of cyber-attacks. To make ECU secure, Kernel is one of the important software that needs to be protected as Kernel has direct access to the hardware. In this paper, the integrity of the kernel is being monitored using ARM TrustZone architecture. The idea to use the ARM TrustZone here for the Kernel security check is to be done at a more privilege level than the Kernel. The Kernel privilege level is EL1, and the monitor code has the highest privilege level that is EL3. Reduction of cost can be achieved through this method as the integrity check is done without using TEE and external hardware to carry out the crypto operations. The efforts to make a trusted application on TEE are also reduced. This proposal can be used for all mid/low automotive ECUs where neither a dedicated hardware controller nor Trusted execution environment (TEE) is available.

References

1. eCall, https://en.wikipedia.org/wiki/ECall.
2. Pese, M., Shin, K., Bruner, J., and Chu, A., "Security Analysis of Android Automotive," *SAE Int. J. Adv. & Curr. Prac. in Mobility* 2, no. 4 (2020): 2337-2346, doi:https://doi.org/10.4271/2020-01-1295.
3. Dadam, S.R., Zhu, D., Kumar, V., Ravi, V. et al., "Onboard Cybersecurity Diagnostic System for Connected Vehicles," SAE Technical Paper 2021-01-1249, 2021, doi:https://doi.org/10.4271/2021-01-1249.
4. "Security in an ARMv8—A System," Documentation - Arm Developer.
5. "Platform Security Boot Guide," https://developer.arm.com/documentation/den0072/latest/.
6. Azab, A., Ning, P., Shah, J., Chen, Q. et al., "Hypervision Across Worlds: 'Real-time Kernel Protection from the ARM TrustZone Secure World'," in *Proceedings of the ACM Conference on Computer and Communications Security*, 90-102, 2014, https://doi.org/10.1145/2660267.2660350.
7. "ARM Power State Coordination Interface Platform Design Document," https://developer.arm.com/documentation/den0022/d/.
8. "Arm-Trusted-Firmware," https://github.com/ARM-software/arm-trusted-firmware.
9. Hofmann, O.S., Dunn, A.M., Kim, S., Roy, I. et al., "Ensuring Operating System Kernel Integrity with OSck," *ACM SIGARCH Computer Architecture News* 39 2011, (2011): 279-290, doi:https://doi.org/10.1145/1961296.1950398.

Contact Information

Shyju Thekkumbadan
Harman International India Pvt. Ltd,
Bangalore, India - 560103,
Contact No: +91 7829400211,
Shyju.Thekkumbadan@harman.com

Definitions/Abbreviations

BL1 - Boot Loader 1

TEE - Trusted Execution Environment

SMC - Secure Monitor Call

OS - Operating System

PSCI - Power state coordination interface

EL - Exception Level

BL2 - Boot Loader 2

ATF - Arm Trusted Firmware

OEM - Original Equipment Manufacturer

ECU - Electronic Control Unit

SoC - System on Chip

ISSN 0148-7191