HW/SW Security Mechanisms for Future Automotive Society

Daisuke Moriyama, Automotive System Security Department, High Performance Computing, Analog Power Core Technology Development Division, High Performance Computing, Analog Power Solution Group, Renesas Electronics Corporation

Introduction

The demand of information security for automotive has substantially increased in recent years. The In-vehicle network is digitalized, and its connectivity is becoming very popular. Similar to the typical network security developed with the internet around 2000, the strong demand of cybersecurity within automotive is rapidly changing. Automotive hacking and theft caused by the flaw of security mechanisms or vulnerabilities are reported from the middle of the 2010's.

Nowadays, automobiles cannot be considered as just a transportation method. It should be treated as electronic devices (similar to laptop computers) and we need to take care that the information security is sufficiently guaranteed. A laptop computer only has one CPU, one vehicle contains about 50-100 ECUs (Electronic Control Units). Furthermore, the total number of ECUs is expected to increase with technological advances like full self-driving system. In many cases, ECU controls a specific safety related function such as powertrain or ADAS (Advanced Driver-Assistance Systems), and these communications must be secure to defend against malicious attacks. Based on the above background, various information security technologies are requested for the semiconductor solutions in the automotive industry

In this white paper, we introduce overall security architecture for the latest automotive semiconductor solutions and argument related to future technologies. We explain what security mechanism is provided in which component, and how to protect the security for each module.

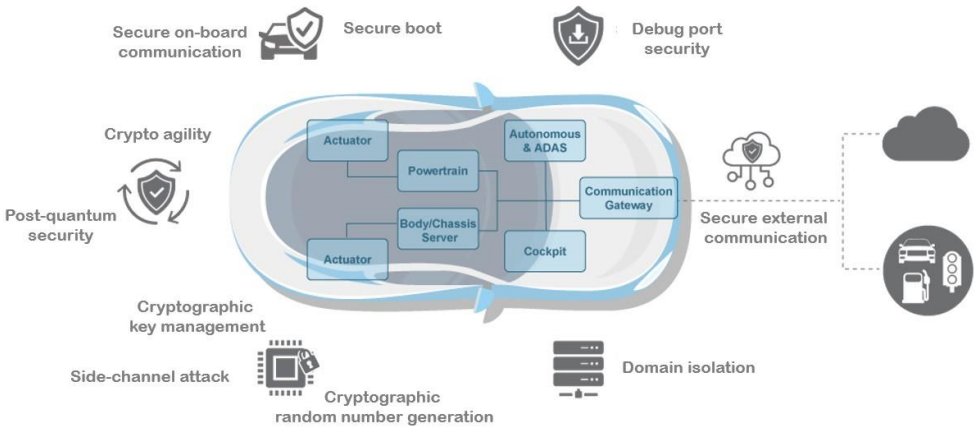


Figure 1 Automotive security technologies

© 2023 Renesas Electronics

Automotive MCU/SoC Security Components

The latest automotive MCU/SoC provides various security functions to cover the digitalization of in-vehicle communication. These products support multiple hardware and software countermeasures against the security threats like the integrity check for boot program, data integrity verification for communication message to defend impersonation attack, data encryption for eavesdropping, etc.

There is usually an HSM (Hardware Security Module) in the automotive MCU/SoC products from semiconductor companies including Renesas Electronics. Especially, high-performance MCU/SoC products contain a dedicated CPU inside the HSM which is independent from the generic application CPU, and it also holds cryptographic hardware including block cipher, hash function, public key cryptography, random number generator, etc. This is one of the criteria to satisfy the highest security level EVITA (E-safety vehicle intrusion protected applications) full, defined by European Union [1]. The HSM is treated as the “Root of Trust” and is responsible for asset protection and access management to provide the security function securely for the whole chip.

On the other hand, the amount of data transactions for each MCU/SoC is gradually increasing as the number of semiconductors inside one vehicle is also growing. Not only traditional communication interfaces as LIN, CAN or CAN-FD, the industry is is implementing Gbps-class broadband ethernet for in-vehicle communication interface nowadays. Furthermore, integrity support is recommended for streaming data from sensors and cameras for automated driving and video file recorded from a dash cam. Accordingly, it is quite hard for only one HSM, per ECU, to handle whole cryptographic processing. Therefore, the latest semiconductor products sometimes contain several security processing modules outside HSM to cover the high security demand.

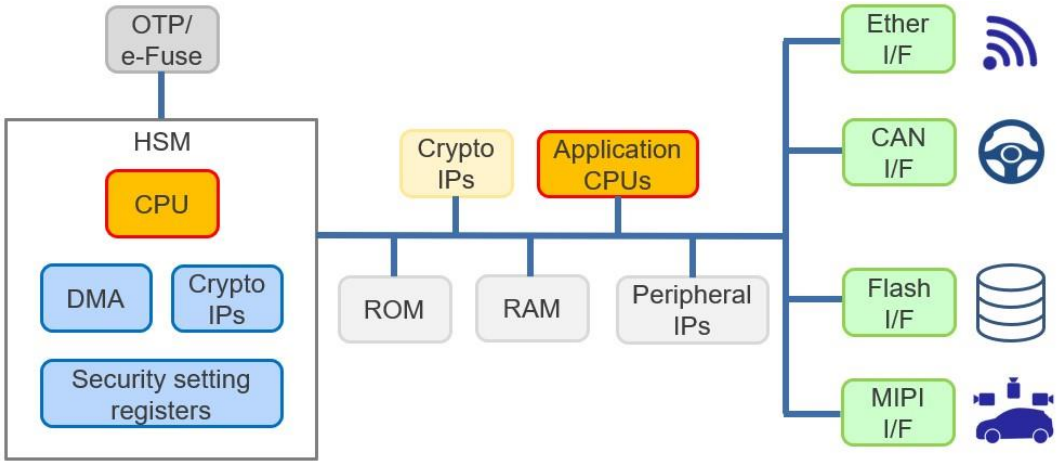


Figure 2 Security related modules and communication interfaces

If we want to keep the robust security measures for the automotive, it is necessary to determine where and how the attacks are attempted and what their goal is. For each of them, we must implement an individual countermeasure (or common technologies). In the next section, we explain several use-cases where security is critical and what countermeasures are implemented or required for the future from the MCU/SoC perspective.

Security Issues to be Considered in Present and Future

[Security inside MCU/SoC]

- **Secure Boot**

  The first step for executing a program in a secure and robust way is to check the authenticity and integrity against the initial boot firmware and underlying software. It is important to verify that all programs in startup timing are legitimate in order to create a secure environment where malware cannot be easily installed. To provide rigorous secure boot, a specific hardware mechanism is quite important. The first boot code should be stored in an area that cannot be modified such as MaskROM or OTP (One Time Program) memory. Starting from the HSM control program, each program to be activated is sequentially verified so that the boot chain covers the legitimacy of the next program. If a boot data must be stored in an external area, it should be encrypted in advance to keep the confidentiality and it is decrypted whenever the boot sequence is launched. The validity check is ensured with the cryptographic technology as the digital signature or message authentication code, so it is better to verify the boot program which manages the execution of HSM and its cryptographic hardware in an early stage. See [2] for more details about the secure boot behavior.

  The latest automotive MCU/SoC products usually equips multi-core CPUs for applications, and it is expected that each CPU core has a different role and will execute its own software. Therefore, it is desirable to build a trust chain for each CPU or launched program and execute a secure boot sequence so that the program with the higher priority can be activated in the minimum latency.

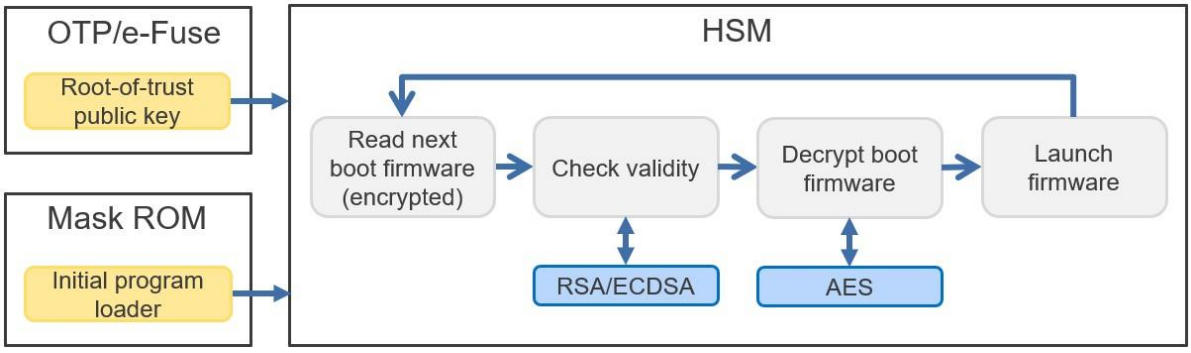


Figure 3 Secure boot

- **Domain Isolation**

  In general, it is necessary to individually configure the access area (non-volatile memory, SRAM, external storage, internal registers in the peripheral modules) between HSM and application CPUs to securely control all data. In addition, if the access authorization must be flexibly configured for each application CPU or cluster to isolate individual data, each of them requires a unique identity to distinguish each domain.

  The access attributes (no access, read only, read-write) are assigned for each access address area and its configuration is judged in hardware whenever the bus master tries to access. Even when SRAM or DRAM is connected to the chip, it is usual to divide its area and set the access attribute for each of them. This control is typically handled by the MPU

(Memory Protection Unit). It provides a mechanism to configure the attribute for a specific unit (e.g., 4KB boundary). It is better to handle a series of these configurations by HSM only so that malicious users cannot intentionally change the access rule.

- **Cryptographic Key Management**

The secure on-board communication in automotive society typically adopts the CMAC mode with AES, and we expect the digital signature is used in the V2x communication to authenticate the unspecified number of devices. The cryptographic secret keys must be carefully stored and managed not only in the above situations but also in any use-cases. When a MCU/SoC fulfills EVITA Medium or higher, there is a dedicated CPU in the HSM. Thus the CPU for the HSM can handle arbitrary secret keys.

How to securely keep the secret key is dependent on the type of microcontroller such that the non-volatile memory (e.g., Flash) is integrated in the chip or equipped outside the chip. If a chip internally contains the non-volatile memory, one easy way to solve this issue is to use the MPU (memory protection unit) so that only the HSM can access to a certain range of the storage area. If there is only an external non-volatile memory (it is technically difficult to mix Flash in advanced processes), then secret keys should be retained even in the power off are encrypted with authenticated encryption and transferred to the external storage. For example, a root-of-trust secret key to encrypt them cannot be encrypted, so it must be stored in the OTP memory so that the HSM can read it during secure boot.

As we described in Figure 2, several recent automotive semiconductor products include multiple cryptographic hardware as an accelerator outside the HSM. We expect the application CPUs to launch them for high-speed or low-latency security operations. We must cover the additional security issues by adding a security function. If we allow multiple application CPUs to access the same cryptographic accelerator, the internal state of the module must be reset whenever a CPU finishes the usage to prevent one secret key is reused for a different purpose.

- **Cryptographic Random Number Generation**

Random number is one of the key elements in cryptography. For example, random numbers are used as the seed of the key generation procedure in public key cryptography or directly treated as the secret key in symmetric key cryptography. If this random number is easily predictable, no security is guaranteed unless a theoretically secure cryptographic algorithm is launched. How to generate a cryptographically secure random number is roughly categorized in two classes: TRNG (Truly Random Number Generator) and DRNG (Deterministic Random Number Generator). TRNG generates randomness with physical phenomenon, such as power supply noise or intentionally metastable circuits. The bias of the TRNG output may be caused by the variance in the manufacturing process or operating environment as temperature. DRNG takes as input a seed which has enough entropy (generally the output from TRNG) and derives longer random sequence with a specific computational algorithm (cryptographic algorithm as block cipher or hash function).

We introduce two famous international standards for random number generation. American governmental organization NIST (National Institute of Standardization

Technology) publishes several publication documents and SP800-90A and SP800-90B describe the requirements to be satisfied or how to construct TRNG/DRNG [4][5]. Germany governmental organization BSI (Bundesamt für Sicherheit in der Informationstechnik) describes several levels for TRNG/DRNG depending on the required properties [6]. Apart from the above documents, NIST SP800-22 describes a statistical test method to evaluate the validity of the output from a random number generator [7].

While it is desirable to implement a secure random number generator especially compliant with the standard specified by NIST or BSI, there are many vulnerability reports that the root of problem is caused by the insufficient randomness in actual IT systems. Therefore we recommend that the randomness generator has an architecture to assess the sufficient randomness on-demand or it is certified by the third party evaluation company.

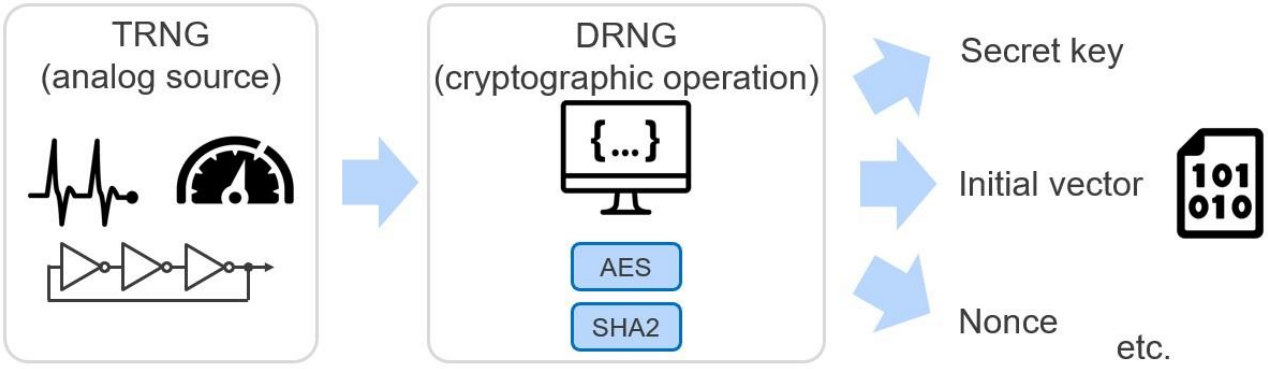


Figure 4 Construction of a Secure Random Number Generator

- **Side-Channel Attack/Physical Attack**

The security functions are provided by a specific procedure or configuration such as access management, encryption, authentication, etc. In contrast, independent from the intrinsic vulnerability of the algorithm or bugs caused during the implementation, there are several attacks which exploit secret information by observing external behavior or inserting a malicious signal. This is challenging to determine which technique can successfully mount the attack to which target, and what mitigation is the suitable countermeasure. We introduce several well-known side-channel attacks and physical attacks, and general countermeasures in the following.

The simplest attack is the timing attack. It exploits secret information by measuring the processing time of the cryptographic algorithm which depends on the secret key value or an access time of the CPU cache to check which confidential information remains in the cache. The timing attack can be launched even from the remote computer, so it is a relatively easy attack method. To defend above attacks, it is usual to add dummy operation in the cryptographic procedure to be executed in a constant time, and erase the cache whenever an essential operation is finished not to cause the latency gap.

If physical access to the chip is available, there are different attack paths. The side-channel attacks with power consumption or electromagnetic radiation are examples of non-invasive passive attacks. An attacker tries to calculate a secret key by observing the transition of power consumption or radiated electromagnetic wave with multiple input-output data. The most effective countermeasure against these attacks is considered as

the masking technique. This method mixes a random number with the secret key and launches a special modified cryptographic algorithm so that the final output is consistent with the normal operation.

Another physical attack categorized as a non-invasive active attack is a fault injection attack. This attack injects a glitch signal to clock or voltage which is connected from an external pin of the chip, and consequently causes an unintended behavior. An attacker can target a branch operation such as a selector circuit in hardware or “IF” clause in software. The fault injection attack tries to proceed an opposite sequence from the actual decision result. For example, a signature verification or challenge-response authentication may be bypassed even if those algorithms are actually computed. To defend the fault injection attack, a dedicated fault detection circuit must be installed to the influential hardware modules or double check should be launched in hardware or software.

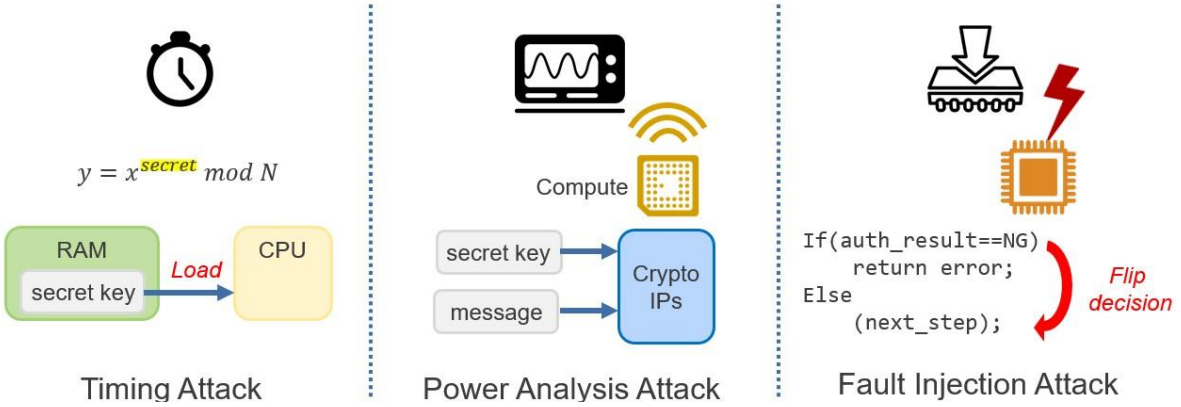


Figure 5 Side-Channel Attack/Fault Injection Attack

[Security for interaction with MCU/SoC]

Secure External Communication

To support system updates for vehicle software, in-vehicle entertainments or V2x (Vehicle-to-Everything) communications for automated driving, wireless communication systems are required such as 5G network, Wi-Fi or NFC (Near Field Communication). The communication channels must be sufficiently secure against eavesdropping, tampering and spoofing interaction. By following the existing security mechanism used by the internet, it is desirable to establish a standard secure communication channel using TLS (Transport Layer Security) or VPN (Virtual Private Network) if necessary. If DoS (Denial of Service) attack or malformed packet is also considered as a critical threat such that it may interfere the fundamental vehicle control system, it will be necessary to equip Firewall, IDS (Intrusion Detection System) or IPS (Intrusion Prevention System) in a specific interface to filter malicious communication packets.

- ### Secure On-Board Communication

Even if the external communication path is secure, it is not enough to establish a whole secure environment. For example, CAN Invader, one of the recent car theft methods, directly connects a dedicated hacking device to the communication interface with a wired

connection and attempts to send commands to open the door. To prevent unintended instructions which are not transmitted from the actual device, it is necessary to constantly check the integrity of the sender of the message in the in-vehicle network.

AUTOSAR (Automotive Open System Architecture) defines that the CMAC mode of AES should be applied to on-board communication [3]. Many semiconductor companies including Renesas Electronics provide AES hardware inside the HSM for automotive products. The in-vehicle network is static and is not changed by the car owners, the secret key used in the CMAC mode is typically stored in the factory in advance based on the network topology. Therefore, it is necessary to keep N secret keys for N communication nodes. If the number of chips in the car is extremely increased in the future, we expect the key exchange may be required instead of pre-shared keys.

While we simply state that integrity is the key point for on-board communication in security, the required throughput and latency fully depends on the usage. For example, driving control information such as steering and braking clearly requires minimum latency, but it is easy to imagine that the amount of data itself is not so large. On the other hand, video data from cameras, necessary for automated driving, is continuously transferred in a real time. It requires sufficient bandwidth and throughput for the cryptographic engine to add integrity, even though the data compression technology has been developed. These examples show that it is desirable to equip various types of cryptographic engines which cover the sufficient processing for each usage in the vehicle.

- **Debug Port Security**

  Semiconductor products typically have a debug port and it allows to access to the various internal resources (non-volatile memory, SRAM and registers) inside the chip. There is no expected situations that car owners use this function. In contrast, it is essential for OEMS, Tier 1s, and software developers in automotive industry to use the debug function to analyze defects and bugs during the development phase. However, if the debug port is opened even after shipping and anyone can access it, a malicious hacker can easily access to the arbitrary information. They can steal communication logs and secret keys contained in the secure zone. Therefore, it is necessary to entirely disable the debug port before shipping or provide a rigorous access protection mechanism so that only trusted users (design and manufacturing companies) can access to the debug port with the identification authentication or challenge-response authentication.

  In the recent automotive hacking news, the self-diagnosis port is usually targeted to analyze in-vehicle communication message and send spoofing commands. To defend these attacks, the access protection mechanism as we noted above so that unauthorized device cannot pass the authentication.

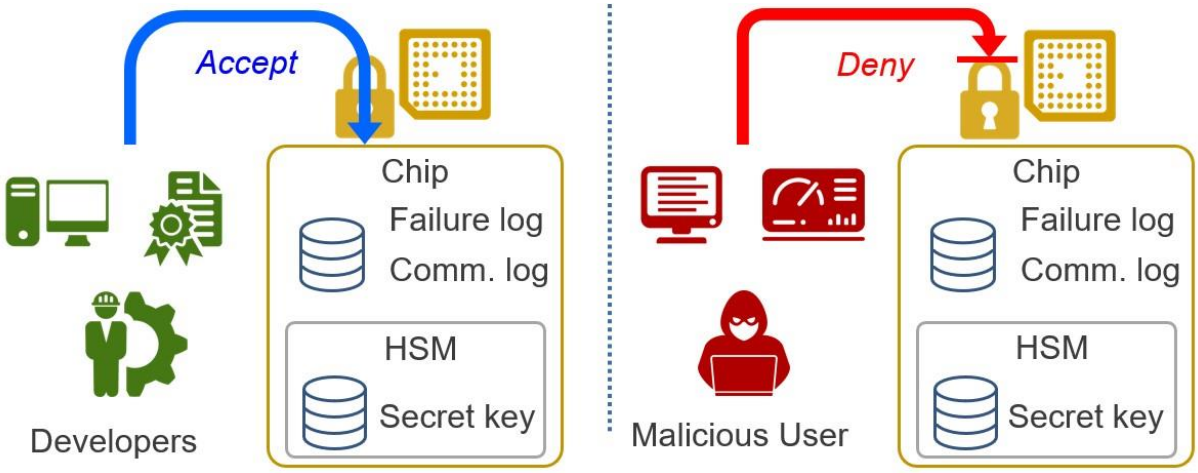


Figure 6 Debug Port Security

[Security for future technological transition]

- **Crypto Agility**

  When we use cryptographic algorithms, we must estimate that the security is sufficiently maintained even if a best solving algorithm or an exhaustive search is launched within a realistic time frame. On the other hand, the computational power of our computers is increasing every year. So it is inevitable to move to stronger key length or cryptographic algorithm after a certain period of time. NIST publishes a guideline that 112-bit security (e.g., 2,048-bit RSA and SHA-224) should not be used after 2030. CRYPTREC publishes more precise guideline as shown in Table 1 [8]. If such an algorithm or key length is used in the current system, it should prepare a migration plan to change cryptographic algorithms to 3,072-bit RSA cryptography, elliptic curve cryptography with P-256 curve or SHA-256 to satisfy 128-bit security near future.

  Even if a cryptographic algorithm is implemented in hardware, when it supports multiple security levels including 128-bit security or higher, it is easy to switch the algorithm to be used with a firmware update. The more important point is that if any portion of the security architecture still works with weak security, this can be an entry point of the intrusion. Therefore the entire security network must be updated in a timely manner to keep a sufficient security level.

  When we focus on the automotive society, it is not a rare case that car drivers own their vehicles for more than 10 years. We need to care about the transition to higher level in security and insecure algorithms should not be utilized if upgrading is possible. Unless the hardware component is FPGA (Field Programmable Logic Array), it is impossible to change the manufactured circuit. One reasonable approach is to support higher security level (than the current demand) when a cryptographic algorithm is implemented in hardware. Then the software can be configured to change the security level according to the world trend.

Table 1 Crypto Agility (from CRYPTREC [8])

		-2030	2031-2040	2041-2050	2051-2060	2061-2070
112-bit security	Encrypt/Sign	Transition	Unavailable	Unavailable	Unavailable	Unavailable
	Decrypt/Verify		Permit			
128-bit security	Encrypt/Sign	Available	Available	Transition	Unavailable	Unavailable
	Decrypt/Verify				Permit	
192-bit security	Encrypt/Sign	Available	Available	Available	Available	Available
	Decrypt/Verify					
256-bit security	Encrypt/Sign	Available	Available	Available	Available	Available
	Decrypt/Verify					

- **Post Quantum Security**

  The cryptographic security for quantum computers which is rapidly developing now is independently evaluated from the traditional computers like PC or cloud server. Public key cryptography such as RSA cryptography and elliptic curve cryptography can be easily broken when a quantum computer has sufficient computational power. NIST has been proceeding with their post-quantum cryptography project which formally selects post-quantum cryptographic algorithms sufficient to meet industry and national needs. Currently, Kyber is selected as a public key encryption scheme, and Dilithium, FALCON, SPHINCS+ are selected as the digital signature algorithms [9]. NIST plans to publish the related FIPS documents in the future after the final specifications of these algorithms are fixed. Apart from the above “Crypto Agility” issue, we recommend planning to replace the public key cryptography schemes for post-quantum security. See [10] for an additional document about post-quantum cryptography.

  Symmetric key cryptography is not immediately broken with quantum computers, different from the public key cryptography. Instead, the current security level is reduced by half compared to the security evaluated with the classical computer. For example, it is better to select AES-256 rather than AES-128 for the block cipher and SHA-512 rather than SHA-256 in the case of the hash function. We estimate the impact with this change is lower than that of public key cryptography. Nonetheless, we still need a certain software update to select a higher security level.

Hardware Countermeasure v.s. Software Countermeasure

This is an arguable issue in security whether hardware countermeasures should be implemented rather than software countermeasures. We discuss this argument in this section since this is a generic issue and applicable to all the above security countermeasures. When a device is used in general-purpose such as PC and server, it is convenient to implement hardware security at minimum, such as root-of trust for secure boot, and the other general security issues are handled by software so that the security measures are flexibly selected for each usage. It is a quite general issue that it is quite difficult to write a bug-free program especially for high level function (both hardware and software). The possibility of a hardware bug should be minimized in general-purpose devices because hardware cannot be fixed after development. Even if a vulnerability is found in the software side, the update program can mitigate the risk.

On the other hand, hardware implementation is inevitable in several situations such that the required performance or latency cannot be satisfied by software. For example, a naïve software implementation of AES with 32-bit CPU requires approximately 1,500 to 2,000 cycles to process 1 block (128-bit) data, while a typical hardware implementation takes only 11 to 12 cycles. When we assume that a cryptographic algorithm is applied to the data transfer like braking or steering control, no critical delay should be caused from the security measures to minimize the possibility of vehicle accidents. Of course, security cannot be completed only from hardware components. These are actually configured by software to start the operation. Therefore, the underlying software must be run without any vulnerabilities.

The hardware-based security countermeasures are definitely beneficial in a specific scenario. But it is not realistic to recall the device and redesign the chip whenever a critical bug caused by a hardware component is found. Semiconductor companies spend a huge amount of time verifying the function including the combination. Furthermore, it is impossible to estimate unexpected threats that will be exploitable in the future from technological progress.

Conclusion

Nowadays, various security technologies are required in automotive semiconductor solutions with the recent rapid technological changes. It is desirable to select and implement sufficient level security technologies including boot procedure, inside and outside communication, physical attack countermeasure, and next generation of cryptography. We always need to keep the security awareness and make an effort so that end users always enjoy secure environment.

There is no perfect security countermeasure in general. The attack method is daily evolving, and a technology which is considered as secure today may not be secure after tomorrow. In the future, we may find ransomware targeting IVI (In-Vehicle Infotainment) or malware with cryptocurrency mining to consume battery charge for electric vehicles. Nonetheless, it is important to assess required security in the future based on the currently known threats from world trends. We hope the security technologies implemented in the current and future semiconductor solutions are useful to minimize the total security risks.

[References]

[1] https://www.evita-project.org/
[2] https://www.renesas.com/us/en/blogs/introduction-about-secure-boot-automotive-mcu-rh850-and-soc-r-car-achieve-root-trust-1
[3] https://www.autosar.org/fileadmin/standards/R22-11/FO/AUTOSAR_TR_SecureHardwareExtensions.pdf
[4] https://csrc.nist.gov/publications/detail/sp/800-90a/rev-1/final
[5] https://csrc.nist.gov/publications/detail/sp/800-90b/final
[6] https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Kryptografie/Zufallszahlengenerator/zufallszahlengenerator_node.html
[7] https://csrc.nist.gov/publications/detail/sp/800-22/rev-1a/final
[8] https://www.cryptrec.go.jp/report/cryptrec-gl-3004-1.0.pdf (in Japanese)

[9] https://csrc.nist.gov/projects/post-quantum-cryptography
[10] https://www.renesas.com/us/en/document/whp/latest-trends-post-quantum-cryptography?r=1601456

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Rev.1.0 Dune 2023)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan
https://www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
https://www.renesas.com/contact-us

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

© 2023 Renesas Electronics Corporation. All rights reserved.